UCLouvain

École polytechnique de Louvain

Stochastic Thermodynamics of Machine Learning

The inevitable cost of neural networks

Author: **Niels Beuseling**
Supervisor: **Jean-Charles Delvenne**
Readers: **John Lee, Alexandre Lazarescu**
Academic year 2020–2021
Master [120] in Data Science: Information technology

Contents

Chapter 1

Introduction

As the digital age continues to unfold itself, the accumulation of data is exploding and its treatment requires evolving methods that can keep up with the increasing quantity of information to analyse [1][2][3]. While the data science market is thriving, the ecological and environmental aspect of newer technologies is becoming ever more important [4], as climate change arises a plethora of issues that need to be addressed [5][6][7][8].

Although data collection and analysis regarding these concerns is an expanding and promising topic [9], ecological improvement could come from technologies already being used. Whether renewable energy is being used or not, one must do its best to avoid wasting energy in vain. This motto is also valid when computing. While this has been heavily discussed by David. H. Wolpert [10] [11] and others [12][13], the motivation of these works all come from the same physical principle, namely the Landauer Principle.

This fundamental result states that "*any logically irreversible manipulation of information, such as the erasure of a bit or the merging of two computation paths, must be accompanied by a corresponding entropy increase in non-information-bearing degrees of freedom of the information-processing apparatus or its environment*" [14]. This statement takes its origin as an attempt to answer Maxwell's infamous demon [15], a thought experiment that challenges the second law of thermodynamics. It stated that if an intelligent being had information about both velocities and positions of the particles in a box, it could split and transfer the hot particles from a cold reservoir to a hot one, without applying external work [16]. This would therefore break the second law of thermodynamics, assuming that the information earned by the demon is a non-physical quantity.

Fortunately, studies have shown that information is physical [15] as it must be recorded in a physical device of some kind, and extensive work has been done in information theory to exploit that knowledge [17]. Landauer's principle gives the first brick to construct a proper thermodynamical framework for computations.

While some researches are performed towards building a better framework and definition of the Landauer's principle for quantum computing [18][19][20], the specific type of computations that are of interest in this thesis are artificial neural networks (ANN), as their use is becoming more and more popular. Evaluating the possible energy loss associated to their use would be beneficial and a good starting point in improving their energetic consumption as some deep-learning applications are quite energetically greedy [21].

Although the training phase of an ANN is the one consuming the most energetic resources, the evaluation done by an already trained model could also theoretically be improved, by configuring a model losing a minimum of energy, given the weights and the input distribution law. This will be the focus of this work, as it constitutes the first step into evaluating the thermodynamic consqequences of running a neural network.

This work will extend and apply the results of [11] to a greater scope, where circuits aren't binary but take real values, or discrete approximations thereof corresponding to a discrete approximation of the real-valued case. Once the framework and its appropriate tools are defined, simple cases will first be approached, to get an insight on how the *mismatch cost*, the cost yielded when giving inputs that do not match the best distribution associated to the ANN, behaves based on the distribution parameters. This will serve as a building block to extend the study upon more complex networks, to learn how such loss of energy behaves through the different layers of a neural network. Using weights of an already trained model, an estimation of the loss due to this non-optimality of the inputs in comparison to what the inevitable cost (*Landauer cost*) is will be computed for a simple classification problem. The approach to estimating the same quantity for the MNIST classification problem will be depicted, although exact computations won't be done as this specific problem is computationally heavy but also challenging in terms of memory usage.

Chapter 2

Thermodynamics of computing with circuits

2.1 A brief history of Landauer's principle

Maxwell's demon

In 1867, James C. Maxwell suggested a thought experiment that could apparently violate the **second law of thermodynamics**. The second law states that the entropy of an isolated system that was at thermodynamic equilibrium can only increase when an irreversible process occurs (*e.g.* when the walls of an isolated system at internal thermodynamic equilibrium become more permeable, heat is transferred from a partition to the other, creating a natural but irreversible process, increasing the entropy of the system).

The experiment is the following : consider a being that is capable of knowing the exact positions and exact velocities of each particle in a closed system. This closed system is split into two partitions, A and B. The being is able to open the wall between the two parts in such manner that only the swift particles of B go in A, and only the slow particles of A go in B[1]. The wall that the demon opens is massless, meaning that it doesn't spend any work by opening it; however by doing so, it rose the temperature of the subsystem A and lowered the temperature of the subsystem B, as the average speed of the particles in A and B was increased and decreased respectively. One could then possibly use a heat engine to extract work from the system, as the entropy of the system seems to have decreased, although no physical work was put into it; only a "sorting" of the particles was performed.

[1]This intelligent being can do as such as it knows the velocities of each particle, whereas only the mean velocity of the particle in a system is usually known, although some particles are a lot faster than others

The approach to tackle this paradox is to take into account the fact that the demon had the information about the system. In this theoretical framework where the demon is omniscient and the opening requires no work, it can be seen as a problem ; however in the real world the acquisition of such information must have required work that corresponds to the rise of temperature [22][23]. The total entropy of both the system and the demon should therefore be considered.

Landauer's principle

To properly grasp how thermodynamic entropy and Shannon entropy are linked, it is wise to consider a simplified example of the thought experiment, where only one particle lies in the box, but in which the demon can push the exterior walls using a piston. In that scenario, if the demon knows in which side of the box the particle is, it can decide to close the wall, use a piston such that the volume of this side of the box decreases, and extract work by opening the wall dividing both sides. The demon therefore could extract work from the system because it had 1 bit of information, since only two states were possible : either the particle was in the left side, or in the other. This two-state point of view can only be seen in the thermodynamic standpoint as having two possible physical states for the system, one where the particle is in one side, one where it's anywhere else. When the demon uses the piston, and then opens up the wall dividing the two parts of the box, it resets the information it had about the particle; this reset was accompanied by the work he extracted. This work in particle can be computed for this single-particle case, as it corresponds to one bit of information, from a 2-states system. The amount of worked extracted is therefore proportional to the entropy of the system, namely $k_B T \ln(2)$ where k_B is the Boltzmann constant which has units of heat capacity ($J \cdot K^{-1}$) and T is the temperature of the heat bath. The $\ln(2)$ comes from the fact that only two states are possible; it corresponds to the Shannon entropy of this system. The thermodynamic entropy is then simply the Shannon entropy multiplied by the Boltzmann constant. This relationship takes its origin to how the system is described, through the different states the system can take; the probability in information theory can be chosen arbitrarily close to the probability of being in a certain thermodynamic state. To each physical state (thermodynamic point of view) corresponds a logical state (information point of view); since the demon knows the current state, it has to memorize it in terms of logical state. While this is a simple example, it applies for more general cases and is not dependent on how information is stored.

Landauer argued that acquiring information may not increase thermodynamic entropy if the way it is acquired is thermodynamically reversible; however this information would therefore have to be stored, such that there is no loss due to

the relationship between Shannon and thermodynamic entropy. Once its trickery has been performed, it needs to clean up its memory (assuming the demon has a finite amount of storage capacity) in order to gather information on the newly obtained states so that it can iterate the malicious transfer. It does so by erasing all stored information, which is an irreversible logical process, and thus needs to emit an amount of heat corresponding to the change in entropy of the system in order to satisfy the second law. This precisely corresponds to the work extracted in the simple example introduced before.

This reset of information, performed through the erasing of the written down information, must therefore be accompanied by an entropy change proportional to the amount of information acquired. In explicit terms, Landauer established a limit to the amount of energy released by the erasure of a single bit, called the *Landauer limit*:

> The minimum possible amount of energy required to erase one bit of information is $k_B T \ln 2$,

where k_B is the Boltzmann constant, T is the temperature of the thermal bath in kelvins. At room temperature $T = 20$ °C, this limit is equal to 0.0175 eV, or $2.805 \times 10^{-21} J$. This serves as a lower bound on how much heat is emitted through the erasure of 1 bit of information within any storage devices.

The fundamental meaning of Landauer's thinking, yielding his principle, can be summarized in terms of the link between irreversible logical operations and increase in entropy. In fact, any irreversible logical operation on information is accompanied by an entropy increase, in terms of physical degrees of freedom (not the information ones). This serves as a basis to express the possible cost of energy in terms of the entropy of the system, when dealing with circuits; further details about this notion will be mentioned in the following sections.

2.2 Information theory and general notation

As information theory is at the center of this work, a non-exhaustive reminder of its functions and relationships will be helpful to grasp all technical aspects of the upcoming results. This also allows for the reader to familiarize with the notations[2].

Random variables are written with an uppercase letter (*e.g.* X), a particular outcome of a random variable by a lower case letter (*e.g.* x) and the set of outcomes by a calligraphed letter (*e.g.* $\mathcal{X}$). The value of a probability distribution for some output x is written $p(x)$ such that $p(X = x) \equiv p(x)$ as it will allow for more

[2]Most of them are taken from [11] and [10]; the reader is encouraged to have a look at these references for further details on how the upcoming results are obtained, as some technical subtleties about mathematical theorems will be omitted.

readability in the following sections. Given a distribution p over the set of outcomes $\mathcal{X}$, for any subset $\mathcal{Z} \subseteq \mathcal{X}$, the probability that the outcome of X is in $\mathcal{Z}$ is given by $p(\mathcal{Z}) = \sum_{x \in \mathcal{Z}} p(x)$. Note that probability distributions are discrete, as logical (binary) states are at the center of this chapter.

An important notion used through Wolpert's work as well as in this thesis is the induced probability Pp, or mapping of a conditional distribution, obtained by applying a conditional distribution $P(y|x)$ of $y \in \mathcal{Y}$ given $x \in \mathcal{X}$ to $p(x)$ over $\mathcal{Y}$:

$$Pp(y) \equiv \sum_{x \in \mathcal{X}} P(y|x)p(x). \tag{2.1}$$

Logical operations are depicted through conditional distributions; a conditional distribution P is *logically reversible* if it is deterministic (for all $x \in \mathcal{X}, y \in \mathcal{Y}$,$P(y|x)$ is binary-valued, at least for this chapter[3] and if $\nexists\, x, x' \in \mathcal{X}$ and $y \in \mathcal{Y}$ such that $P(y|x) > 0$ and $P(y|x') > 0$ ([11]).

Definition 2.1. The **Shannon entropy** of a distribution $p(X)$ over a set of outcomes $\mathcal{X}$ such that $X \in \mathcal{X}$ is given by the sum over all possible values taken by X :

$$S(p(X)) = -\sum_{x \in \mathcal{X}} p(x) \ln p(x). \tag{2.2}$$

Shannon entropy can be used as a measure of "uncertainty", giving an indication on the randomness of the possible outcomes of the random variable. In that sense, a uniform probability (*e.g.* a probability of 1/2 of having one of 2 possible outcomes) yields the greatest entropy as none of the possible outcomes is likelier than another; maximum randomness is assured.

Another useful quantity used to compare distributions is the *Kullback-Leibler* (KL) *divergence* (sometimes called *relative entropy*). It measures how much a distribution p is different from another distribution q, used as reference. This quantity is particularly used when defining the mutual information, a quantity measuring the mutual dependence between two random variables.

Definition 2.2. The **Kullback-Leibler** divergence between two probability distributions p and q over the set of outcomes $\mathcal{X}$ is given by :

$$D(p(X)||q(X)) = \sum_{x \in \mathcal{X}} p(x) \ln \frac{p(x)}{q(x)}. \tag{2.3}$$

While the following quantities are not directly involved in the upcoming results, it would be a shame to not include the definition of mutual information in a proper

[3]This binary aspect is a consequence of dealing with logical gates; in that manner, a AND and a XOR are deterministic, as their value (the value of the output gate) depends deterministically on the values of the input gates.

information theory reminder. This definition requires the conditional entropy, which is defined as follows :

Definition 2.3. The **conditional entropy** of a random variable X conditioned on a variable Y under joint distribution p is defined as

$$S(p(X|Y)) = \sum_{y\in\mathcal{Y}} p(y)S(p(X|y)) \tag{2.4}$$

$$= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(y)p(x|y)\ln p(x|y). \tag{2.5}$$

It is now time for mutual information to be explicitly introduced. This measure, as mentioned before, gives an indication on how much two random variables are mutually dependent. This mutual dependency allows for example to detect redundancy when dealing with a lot of random variables, as a high mutual dependency between a random variable X and a random variable Y means that knowing X gives a lot of information about Y and thus knowing both doesn't bring a lot to the table in terms of information gain.

Definition 2.4. The **mutual information** between two random variables X and Y according to a distribution p is, assuming they can be jointly distributed :

$$I_p(X;Y) \equiv S(p(X)) + S(p(Y)) - S(p(X,Y)) \tag{2.6}$$

$$= S(p(X)) - S(p(X|Y)). \tag{2.7}$$

One can also consider the mutual information as being a measure of the dependence of X and Y when jointly distributed, relative to their marginal distribution; in fact $I_p(X,Y) = 0$ if and only if X and Y are independent, meaning $p(X,Y) = p(X)p(Y)$.

The last notions to be introduced are extensions of the mutual information and KL-divergence when dealing with distributions over a multi-dimensional space.

Definition 2.5. The **multi-information** of a vector of random variables $\mathbf{X} = (X_1; X_2; ...)$ distributed according to a distribution p is given by

$$\mathcal{I}(p(X_1; X_2; ...)) \equiv \left(\sum_i S(p(X_i))\right) - S(p(\mathbf{X})). \tag{2.8}$$

Definition 2.6. The **multi-divergence** between two probability distributions o and q over $\mathbf{X} \in \mathcal{X}$ where $\mathcal{X}$ is multi-dimensional is defined as :

$$\mathcal{D}(p(X_1; X_2; ...)||q(X_1; X_2; ...)) \equiv D(p||q) - \sum_i D(p(X_i)||q(X_i)) \tag{2.9}$$

As logical reversibility is a key element to determine thermodynamic behavior of the circuit, defining it in terms of sets will allow for easier use and interpretation in the upcoming theorems and their proofs. The *island* definition therefore comes handy as it characterizes equivalence classes on the set of outcomes for which the conditional distribution P is not logically reversible.

Definition 2.7. Consider the following equivalence relation

$$x \sim x' \leftrightarrow \exists y : P(y|x) > 0, \qquad P(y|x') > 0, \tag{2.10}$$

with $x, x' \in \mathcal{X}$. An **island** of the conditional distribution $P(y|x)$ is defined as any connected subset given by the transitive closure of the above equivalence relation. The set of islands of the conditional distribution P is written $L(P)$, which actually forms a partition of $\mathcal{X}$. When dealing with subsets of $\mathcal{X}$, the partition of $\mathcal{Z} \subseteq \mathcal{X}$ generated is written $L_{\mathcal{Z}}(P)$.

Example 2.8 (AND gate)**.** Given $\mathcal{X} = \{(0,0),(0,1),(1,0),(1,1)\}$, and the conditional distribution P implementing the logical AND operation of two binary variables such that

$$P(y|(x_1,x_2)) = \delta(y, x_1 \wedge x_2), \tag{2.11}$$

there are two resulting islands, the first one corresponding to $y = 0$ given by $(x_1,x_2) \in \{(0,0),(0,1),(1,0)\}$, and the other corresponding to $y = 1$ given by $(x_1,x_2) \in \{(1,1)\}$.

The reader is now armed to face the multitude of information theory and probability concepts they will face. However, this quite mathematical reminder lacks thermodynamic notions to properly describe how logical gates and entropy production are linked.

2.3 Stochastic thermodynamics

Stochastic thermodynamics is an emerging field of statistical mechanics that aims at describing non-equilibrium dynamics through the help of stochastic variables. While its applications are various, going from quantum systems [24] [25] to biopolymers and enzymes [26], it allows for a good understanding on entropy dynamics occurring within computational machines.

To account for thermodynamic properties, an assumption is made on the circuits to be physical systems in contact with one or more thermal reservoirs [11]. The fundamental notion that allows for the study of entropy dynamics is that as the physical system evolves, entropy not only flows, but is also produced. In that sense, entropy changes will be evaluated on a simple time-interval $t \in [0,1]$ to ease the description of the upcoming concepts.

To the physical system is associated a countable state space X. This system thus evolves over the above-mentioned time interval, while in contact with a thermal reservoir and possibly connected to a work reservoir. Note that the term "reservoir" depicts an exchange between the system an the reservoir, according to its adjective : heat for thermal reservoirs, and *work* for work reservoir[4], in the sense that they exchange work, entropy or heat, playing with the energy spectrum[5] associated to the system in the sense of statistical physics, therefore inducing or extracting work [27] . The system's dynamics are modelled by *continuous-time Markov chain* (CTMC), or *Markov Processes*.

Definition 2.9. A continuous-time Markov chain $(X_t)_{t\geq 0}$, or Markov Process, is a stochastic process whose arrival times (time between two state changes) are exponentially distributed and depend only on the current state and the state of arrival. It is defined through several parameters [28][29] :

- a countable state space $\mathcal{X}$,
- a transition rate matrix K_t with the same number of rows and columns as dimensions of $\mathcal{X}$ and evolving according to t,
- an initial state n such that $X_0 = n$.

Off-diagonal elements ($i \neq j$) of K are non-negative, and correspond to the rate of transition between state i and j. As the elements of the rows of the rate matrix are expected to sum up to zero for each rows[6], diagonal elements are negative, in particular :

$$K_t(i,j) = -\sum_{i\neq j} K_t(i,j) \tag{2.12}$$

CTMC's are useful in order to model a system with countable state space as it offers a convenient mathematical framework for the dynamics at work. The probability distribution p^t of the system at time t over $\mathcal{X}$, considering that the system is coupled to multiple thermodynamic reservoirs indexed by α, behaves

[4]See [10] p21-22.

[5]In equilibrium statistical physics, this refers to the spectrum associated to the partition function of the system. For the reader not accustomed to statistical physics, the partition function constitutes a toolbox of the thermodynamic properties of the system based on a probabilistic formalism, as quantities such as the entropy or the free energy can be easily derived from the partition function.

[6]This convention is quite different to the one when dealing with transition matrix in the discrete case (where the rows sum up to 1) and is the convention used throughout [11] and [10]

according to the following equation :

$$\frac{\mathrm{d}}{\mathrm{d}t}p^t(x') = \sum_x p^t(x)K_t(x \to x') \tag{2.13}$$

$$= \sum_{x,\alpha} p^t(x)K_t^\alpha(x \to x'), \tag{2.14}$$

where $K_t^\alpha(x \to x')$ is the rate matrix at time t for reservoir α, for which the off-diagonal entry $K_t^\alpha(x \to x')$ corresponds to the rate at which probability flows (or, in other words, evolves monotonically as time goes on) from state x to x' for $x \neq x'$. Conservation of probabilities is guaranteed thanks to the condition imposed on the rows for K_t, the global rate matrix, that satisfies the equation $K_t(x \to x') = \sum_\alpha K_t^\alpha(x \to x')$.

Considering this flow of probability, the *entropy flow* (EF) can be defined as follows :

Definition 2.10. The instantaneous rate of **entropy flow** out of the system at time t is :

$$\dot{Q}(p^t) = \sum_{\alpha,x,x'} p^t(x)K_t^\alpha(x \to x') \ln \frac{K_t^\alpha(x \to x')}{K_t^\alpha(x' \to x)}. \tag{2.15}$$

The overall entropy flow incurred over the course of the entire process is $Q = \int_0^1 \dot{Q}\mathrm{d}t$.

It corresponds to the increase of entropy in all coupled reservoirs; through the definition of its instantaneous rate, one can see how the flow of probabilities characterizes the evolution of the entropy flow : if it is more likely for the probability to flow from x to x', the logarithmic factor will be great, thus bringing a positive contribution to the instantaneous entropy flow rate; while the converse will yield a negative value for the logarithm (as the argument will be lower than 1), therefore contributing negatively to the flow.

To highlight the physical interpretation of such formalism, the probability of any particular trajectory $\mathbf{x} = (x_0, x_1, ..., x_n)$, the vector of all states or positions visited by the trajectory at corresponding times $\tau = (\tau_0 = 0, \tau_1, ..., \tau_{N-1}, \tau_N = 1)$, having its initial position at x_0 is given by[7] :

$$p(\mathbf{x}|x_0) = \left(\Pi_{i=1}^{N-1} S_{\tau_{i-1}}^{\tau_i}(x_{i-1})K_{\tau_i}(x_i \to x_{i-1})\right) S_{\tau_{N-1}}^1(x_N). \tag{2.16}$$

where $S_\tau^{\tau'}(x) = \exp(\int_\tau^{\tau'} K_t(x \to x')\mathrm{d}t)$ corresponds to the probability of staying in the current state x throughout the interval $t \in [\tau, \tau']$. The reader might observe some similarities with the formalism used in path integrals; it is no coincidence

[7] See [10] for further details.

since the same thinking is involved here, both topics use a stochastic approach to model physical systems[8]. The total *entropy production* (EP) can be deduced from the entropy flow, using the differential expression of the latter to evaluate the instantaneous rate of EP.

Definition 2.11. The instantaneous rate of **entropy production** at time t is defined as :

$$\dot{\sigma}(p^t) = \frac{\mathrm{d}}{\mathrm{d}t} S(p^t) + \dot{Q}(p^t). \tag{2.17}$$

As for the entropy flow, the overall EP induced over the whole duration of the process can be obtained as follows : $\sigma = \int_0^1 \dot{\sigma} \mathrm{d}t$.

The intuition behind this definition is that the total increase of entropy in the universe when the system is run is equal to the change of entropy in the system, in addition to the entropy flowing into the heat bath.

When considering a system with some initial distribution p on which a conditional distribution P is induced (meaning, the conditional distribution acts as an update of the state distribution of the system) , the relationship between the drop of entropy, entropy flow and entropy production is given by [32] :

$$Q(p) = [S(p) - S(Pp)] + \sigma(p). \tag{2.18}$$

Both terms are non-negative quantities, whose contribution will be discussed separetely. A convenient decomposition of the entropy production will help to find a lower bound to its contribution to the total entropy flow.

Note that while the entropy flow can take both negative and positive values, it can be shown that the entropy production is non-negative, or in other words

$$Q(p) \geq S(p) - S(Pp). \tag{2.19}$$

In this framework where it is assumed that the system evolves forward in time like a continuous-time Markov chain, this result can be percieved as a derivation of the

[8]While path integral formalism is used to account for the contribution of all possible paths through a random-walk approach, based on the action of the system, stochastic thermodynamics depend on the rate matrix of the system, which is linked to the thermal reservoirs; the rate matrix therefore constitutes the physical quantity at the center of this formalism. In particular, off-diagonal entries of the rate matrix must be such that for any reservoir α, K_t^α must obey the *local detailed balance*, a concept that allows for small changes in entropy over small time intervals; a more popular but similar result is the fluctuation theorem [30]. To obey such condition, for all $x, x' \in \mathcal{X}$ K_t^α must be such that either $K_t^\alpha(x \to x') = K_t^\alpha(x' \to x) = 0$, or

$$\frac{K_t^\alpha(x \to x')}{K_t^\alpha(x' \to x)} = e^{\beta_\alpha (H_t(x) - H_t(x'))},$$

given the Hamiltonian $H_t(\cdot)$) of the system[10][31][26].

second law of thermodynamics [33].

Precautions must be taken when using CTMC (continuous-time Markov chains) to implement dynamics of circuits; would it be a bit-flip or a erasure, some actions cannot be directly and precisely implemented using CTMC. Furthermore, some processes would imply some transition rates to be exactly zero, as possible errors aren't considered in the logical operations considered in [11]. However, it is possible to design CTMC that apply to the specific operation, given any conditional distribution to some arbitrary precision by extending the dynamics through "hidden states"; an elegant approach at tackling recurring issues that are encountered in non-equilibrium statistical physics that is explained in [34]. Since this goes beyond the scope of this work, it is assumed that such complying implementations will be possible through *serial-reinitialized circuits*, that will be introduced further below. A last section about the convention and notations used when dealing with circuits is however needed beforehand.

2.4 Decomposition of the entropy production

As entropy is unavoidably produced when running a circuit, it is useful to have an appropriate decomposition of its production based on the initial state distribution as well as the conditional distribution that describes the dynamics of the system. The upcoming decomposition evaluates how the input distribution alters the thermodynamic cost of a gate, and how it would change if the input distribution would be different.

The first step to this decomposition is to consider the lower bound of possible entropy flow, given by Eq. 2.19. This drop in entropy is referred to as the Landauer cost, that is defined as :

$$\mathcal{L}(p) = S(p) - S(Pp). \tag{2.20}$$

By Eq. 2.18, the relationship between the entropy production and the entropy flow therefore depends on the initial distribution p of the process. The upcoming decomposition of the entropy production will involve a quantity describing how much entropy is produced due to the fact that p is not the optimal distribution while the other part of the decomposition will correspond to the unavoidable EP that would occur, even if p was optimal. The decomposition is obtained as a result of the following theorem, as explained in [11] (the proof of this theorem can be found in the same article). This theorem applies for functions that can be written as the sum of the Landauer cost and an expected quantity that depends on p :

Theorem 2.12. *For any function $F : \Delta_{\mathcal{X}} \to \mathbb{R}$ of the form $F(p) \equiv S(Pp) - S(p) + \mathbb{E}_p(f)$ where $P(y|x)$ is some conditional distribution of $y \in \mathcal{Y}$ given $x \in \mathcal{X}$ and*

$f : \mathcal{X} \to \mathbb{R}\{\infty\}$ is some function; let $\mathcal{Z}$ be any subset of $\mathcal{X}$ such that $f(x) < \infty$ for $x \in \mathcal{Z}$, and let $q \in \Delta_{\mathcal{Z}}$ be any distribution that obeys

$$q^c \in \underset{r:supp\ r \subseteq c}{\operatorname{argmin}} F(r) \qquad \forall c \in L_{\mathcal{Z}}(P)$$

.

Then, each q^c will be unique, and for any p with supp $p \subseteq \mathcal{Z}$,

$$F(p) = D(p||q) - D(Pp||Pq) + \sum_{c \in L_{\mathcal{Z}}} p(c) F(q^c). \tag{2.21}$$

This theorem can be directly applied to the entropy production, with $\mathcal{Y} = \mathcal{X}$ and $\mathbb{E}_p(f) = \mathcal{Q}$, with the EP of running a conditional distribution $P(y|x)$ corresponding to F in the theorem. It therefore follows that

$$\sigma(p) = D(p||q) - D(Pp||Pq) + \sum_{c \in L(P)} p(c) \sigma(q^c). \tag{2.22}$$

In the theorem, it is assumed that q^c is unique, so that the optimal distribution is unique for each island, but there is no assumption that q is. The optimal distribution q is unique only if there is a single island, as $L(P)$ would be constituted of only one element.

The drop in KL-divergence between in the first two terms of Eq. 2.22 is referred to as **mismatch cost**. The mismatch cost is non-negative by construction, and equals zero when the optimal distribution of each island is the one run by the circuit. Such an optimal distribution yielding zero mismatch cost is also referred to as prior distribution of the physical process, that has to be taken in the Bayesian interpretation of the term; that is, the distribution that best describes the distribution of states of the system, based on the knowledge acquired about the system (would it be a first guess or obtained after several experiments). The last term in the decomposition corresponds to the unavoidable entropy production, also referred to as **residual EP**.

If the conditional distribution P that runs the dynamics of the system is not logically reversible over $\mathcal{Z} \subseteq \mathcal{X}$, then not every initial distributions p with supp $p \subseteq \mathcal{Z}$ will yield a zero mismatch cost, as there will be such a distribution p that behaves similarly under the mapping P as the optimal distribution q, with the KL divergence between Pp and Pq being lower than for other initial distributions. This implies that some unavoidable entropy production is yielded by some initial distributions when P is not logically reversible. When it is, with supp $p \subseteq \mathcal{Z}$, then the mismatch and Landauer costs are zero and the entropy flow is equal to the entropy production.

2.5 Circuits notations

Since conditional distributions implement the logical operations, logical circuits are considered as a special type of Bayes networks [11]. A circuit Φ is defined as a tuple $(V, E, F, \mathcal{X}_V)$, where the pair (V, E) specifies the vertices and edges of a directed acyclic graph, corresponding to the *wiring diagram* of the circuit. For each node g of the circuit, the set of possible states associated to this node is $\mathcal{X}_g$, and $\mathcal{X}_V = \Pi_g \mathcal{X}_g$ is the cartesian product of all set of states for each node. The remaining parameter F corresponds to the set of conditional distributions corresponding to the logical operations at the non-root nodes.

In this section and as in [11], the term *gate* is used to refer to any non-root node, while **input nodes** denotes non-root nodes as the circuit is considered to be a directed acyclic graph (DAG), and the **output nodes** or **output gates** are the leaf nodes. The input nodes are denoted by IN, and their joint sate by x_{IN}. Output nodes are denoted in the same fashion, replacing the IN by OUT. The set of all gates is denoted $G \subseteq V$, and a particular gate is indicated by $g \in G$ with its parent nodes (the nodes that point towards g) as $\text{pa}(g)$. As the conditional distributions in the circuit correspond to the logical maps that are implemented in the circuit, each element of F corresponding to a gate g is written as $\pi_g(x_g|x_{\text{pa(g)}})$.

Definition 2.13. Given the set of output nodes x_{OUT} and the set of input nodes x_{IN} , the **overall conditional distribution** implemented by a circuit Φ is defined as

$$\pi_\Phi(x_{\text{OUT}}|x_{\text{IN}}) = \Sigma_{x_{G\setminus\text{OUT}}} \Pi_{g\in G} \pi_g(x_g|x_{\text{pa}(g)}), \tag{2.23}$$

where G is the set of all gates in the circuit Φ.

To illustrate this definition, the simple XOR operation implemented by a gate is therefore given by $\pi_g(x_g|x_{\text{pa(g)}}) = \delta(x_g, \text{XOR}(x_{\text{pa(g)}}))$, thus applying the result of the XOR onto x_g.

To account for the physical connection between gates, in particular some kind of wire, this link is described by a **wire gate** such that for each edges in the circuit, it can be split such that the wire gate w has a single parent and a single child; each edge in the circuit will therefore connect a non-wire node to a wire node. The set of wire gates is described by $W \subset C$, and each wire gate therefore implements the identity map $\pi_w(x_w|x_{\text{pa}(w)}) = \delta(x_w, x_{\text{pa}(w)})$[9].

[9]When using the results of this section with artificial neural networks, the wire gates will not be explicitly mentioned as the quantities that will be of interest do not depend on wire gates explicitly. While wire gates provide a useful interpretation of physical connection between gates, they will not be useful in the upcoming sections as the quantities that will be derived in this section will be directly appliable with the artificial neural network case.

2.6 Solitary processes and serial-reinitialized circuits

An important feature in physical circuits lies in their structure, which limits the physical coupling of each gates to only their direct inputs and its output. It means that when a gate is run, only the subsystem corresponding to the gate and its direct inputs and output is subject to thermodynamic change; the other gates are invariant under the running of this gate as they are not physically coupled to it (or more precisely, their coupling is neglected).

This property implies a possible decomposition of the entropy production and entropy flow of the system, which will be introduced in a bit. This decomposition will consider the impact of the wiring, the impact of the physical circuit implementation on the entropy flow and entropy production. First, let's see how this modular aspect of the physical circuit can be expressed in terms of the system's conditional probability.

As mentioned earlier, one can consider the gate that is running as a particular subsystem of the subsystem, while all the other gates to which it is not physically coupled constitute the complementary subsystem. One can therefore describe the system's overall state space $\mathcal{X}$ as the combination of two major subsystems, such that $\mathcal{X} = \mathcal{X}_A \times \mathcal{X}_B$, where the corresponding states are written (x_A, x_B). The term **solitary process** [11] is used to refer to any physical process over a state $\mathcal{X}_A \times \mathcal{X}_B$ during a time interval $t \in [0, 1]$ such that :

1. The subsystem described by A evolves independently of B, while B doesn't change :
$$P(x'_A, x'_B|x_A, x_B) = P_A(x'_A|x_A)\delta(x'_B|x_B); \tag{2.24}$$

2. The entropy flow of the process depends only on the initial distribution of the subsystem which is currently running, namely
$$\mathcal{Q}(p) = \mathcal{Q}_A(p_A). \tag{2.25}$$
Consequently, based on the lower bound already expressed in Eq. 2.19, the entropy flow is bounded by the change in entropy[10] of subsystem A, based on its marginal distribution p_A :
$$\mathcal{Q}_A(p_A) \geq S(p_A) - S(P_A p_A). \tag{2.26}$$

[10]The author refers to such entropy as "marginal entropy", as it depends on the marginal distribution of the inputs, corresponding to the subsystem A's inputs. While the overall entropy is also expected to change, the bound is expressed in respect of the subsystem's entropy flow.

Note that the conditional distribution implemented in the subsystem A is described as P_A, such that it only applies onto its corresponding states in Eq. 2.24.

The **subsystem EP** for this solitary process is expressed as

$$\sigma_A(p_A) \equiv \mathcal{Q}_A(p_A) - [S(p_A) - S(P_A p_A)]; \tag{2.27}$$

if P_A is logically reversible, then the term between brackets (the subsystem Landauer cost) vanishes, resulting in an entropy production in the subsystem equal to the entropy flow due to this gate running alone – the solitary process. One can also highlight that the subsystem entropy can be decomposed as in theorem 2.12.

The Landauer cost of the system is usually not equal to the entropy of the joint system over $\mathcal{X}_A \times \mathcal{X}_B$; the difference between the two is referred to as **Landauer loss** :

$$\mathcal{L}^{\text{loss}}(p) \equiv [S(p_A) - S(P_A p_A)] - [S(p_{AB}) - S(P p_{AB})]. \tag{2.28}$$

It can also be described as the difference between the joint entropy and the marginal entropy, namely $\mathcal{L}^{\text{loss}}(p) = \sigma(p_{AB}) - \sigma_A(p_A)$.

Keeping this modular aspect of the circuit in mind, it is important to consider how such property can be implemented. The type of circuits considered in [11] are called **serial reinitialized** circuits, or **SR** circuits, in which the gates are run one after the other – *serially*. In particular, such a gate has its output value set to $\emptyset$ (a special value to which every gate is initialized), then set to the value obtained by running the inputs through the gate (under the conditional distribution implemented); eventually the gate then reinitializes the states of the parent gates once the gate finishes to run. This allows for repetition in the circuit as all nodes are set to $\emptyset$, *reinitialized*, which will yield the same expected thermodynamic cost.

As just mentioned, all nodes $v \in V$ are set to a special value $x_v = \emptyset$ at time $t = 0$, before the circuit starts to run. As the inputs nodes are set to some distribution p_{IN}, the cascading sequence of solitary processes starts; each solitary process constitutes a subset A, and the rest of the system constitutes a subset B, as described before. Naming the set of both gate g and its parent $\text{pa}(g)$ as $n(g) = \{g \cup \text{pa}(g)$, this set of nodes evolve as follows :

$$P_g(x'_{n(g)}|x_{n(g)}) = \pi_g(x'_g|x_{\text{pa}}(g))\Pi_{v\in\text{pa}(g)}\delta(x'_v, \emptyset). \tag{2.29}$$

The other nodes in the circuit do not change their state.

Of course, when the whole circuit has finished to run, the final values for the outputs are registered on some external memory device, after which they are reinitialized such that the circuit can be run another time. The state of the ouptut gates follows the distribution $\pi_\Phi(x_{OUT}|x_{IN})$.

2.7 Thermodynamic costs of SR circuits

Again, the results of [11] are described below. Only one decomposition of the entropy flow will be considered, as its definition of the mismatch cost is the one that will be used.

As a reminder, the total entropy flow of a circuit implementing a mapping P can be decomposed in to the sum of the Landauer cost $\mathcal{L}(p) = S(p) - S(Pp)$ and the entropy production $\sigma(p)$, such that $\mathcal{Q}(p) = \mathcal{L}(p) + \sigma(p)$.

When considering a particular part of the circuit, namely a solitary process within a gate and its parents, the total entropy flow can be obtained as follows :

Lemma 2.14. The total EF incurred by running the sequence of two solitary processes over the parents of gate g given some initial distribution $p_{\mathrm{pa}(g)}$ can be decomposed as

$$\begin{aligned}\mathcal{Q}(p_{\mathrm{pa}(g)}) =& S(p_{\mathrm{pa}(g)}) - S(\pi_g p_{\mathrm{pa}(g)}) + D(p_{\mathrm{pa}(g)}||q_{\mathrm{pa}(g)}) - D(\pi_g p_{\mathrm{pa}(g)}||\pi_g q_{\mathrm{pa}(g)}) \\ &+ \sum_{c \in L(\pi_g)} p_{\mathrm{pa}(g)}(c)\hat{\sigma}_{n(g)}(q_{\mathrm{pa}(g)}).\end{aligned} \tag{2.30}$$

The drop in KL divergence is the **mismatch cost**, which illustrates the disparity between the optimal (or prior) distribution and the actual one. Since the decomposition of the entropy flow is based on the circuits and their implementation, it is only normal to find a decomposition fitting to the structure of the system. Hence, the mismatch cost, which contributes to the entropy production, is defined as follows :

Definition 2.15. The mismatch cost of the circuit is given by the sum of the differences between the KL-divergence of the parent nodes distribution to the optimal one and the KL-divergence of the child node distribution :

$$\mathcal{M}(p) = \sum_{g \in G \setminus W} \Big[D(p_{\mathrm{pa}(g)}||q_{\mathrm{pa}(g)}) - D(\pi_g p_{\mathrm{pa}(g)}||\pi_g q_{\mathrm{pa}(g)}) \Big] \tag{2.31}$$

where G is the set of gate nodes, and $>$ the set of wire nodes.

As said before, the mismatch cost constitutes an element of the entropy production, which itself can be considered as a part of the total entropy flow. This total EF can actually be decomposed as follows :

Theorem 2.16. *The total EF incurred by running an SR circuit where p is the initial distribution over the joint state of all nodes in the circuit is*

$$\mathcal{Q}(p) = \mathcal{L}(p) + \mathcal{L}^{loss}(p) + \mathcal{M}(p) + \mathcal{R}(p) \tag{2.32}$$

where

- $\mathcal{L}(p)$ *is the Landauer cost;*
- $\mathcal{L}^{loss}(p)$ *is the circuit Landauer loss, the inevitable entropy flow generated due to the SR implementation of the circuit when gate* g *is run, defined by*

$$\mathcal{L}^{loss}(p) = \sum_{g \in G \setminus W} \mathcal{L}_g^{loss}(p)$$

  where $\mathcal{L}_g^{loss}(p) = S(p_{pa(g)}) - S(\pi_g p_{pa(g)}) - \mathcal{L}_g(p)$ *as described in the previous section and* $\mathcal{L}_g(p)$ *is the drop of entropy of the entire circuit, considering the joint distribution;*

- $\mathcal{R}(p)$ *is the circuit residual EP, which is equal to the subsystem EP that would be incurred even if each island of each gate was run optimally.*

Note that the joint distribution over all the nodes, if the wiring is not known, is given by

$$p(x_V) = p_{IN}(x_{IN}) \Pi_{v \in V \setminus IN} \delta(x_V, \emptyset) \tag{2.33}$$

while the ending joint distribution is

$$Pp(x_v) = p_{OUT}(x_{OUT}) \Pi_{v \in V \setminus OUT} \delta(x_v, \emptyset). \tag{2.34}$$

Since $S(Pp) = S(p_{OUT}) = S(\pi_\Phi p_{IN})$, the Landauer cost is then given by

$$\mathcal{L}(p) = S(p_{IN}) - S(\pi_\Phi p_{IN}). \tag{2.35}$$

In particular, this helps redefine the circuit Landauer loss, which can be obtained as

$$\mathcal{L}^{\text{loss}}(p) = \mathcal{I}(p_{IN}) - \mathcal{I}(\pi_\Phi p_{IN}) - \Sigma_{g \in G \setminus W} \mathcal{I}(p_{\text{pa}}(g)). \tag{2.36}$$

where $\mathcal{I}(\cdot)$ refers to the multi-information defined in Eq. 2.8. Since $\mathcal{I}(p_{IN})$ and $\mathcal{I}(\pi_\Phi p_{IN})$ do not depend on the wiring specifically, minimizing the Landauer loss means minimizing $\Sigma_{g \in G \setminus W} \mathcal{I}(p_{\text{pa}}(g))$. This results, as Wolpert indicates in [11], to choose a wiring such that the parents of each gate are as strongly correlated among themselves as possible. While this applies for wirings that are not symmetric, this consequence loses its usefulness when dealing with artificial neural networks since they are mainly highly symmetric networks. This special choice of wiring to minimize the loss of correlations is still interesting as the mismatch cost will be minimum when the inputs are totally uncorrelated. This result will be developed in the next chapter, but one can already think about the link between the solution of the optimization problem with the Landauer loss proposed by Wolpert and the mismatch cost : as the most correlated inputs are considered together, they "merge" into one input that is therefore less correlated with the other ones, hence possibly minimizing the mismatch cost. This will be discussed in the next chapter.

In this same mindset, the entropy production can also be decomposed as

$$\sigma(p) = D(p_{IN}||q_{IN}) - D(\pi_\Phi p_{IN}||\pi_\Phi p_{IN}) + \Sigma_{c\in L(\pi_\Phi)} p(c) \sum (q^c_{IN}). \tag{2.37}$$

Such formalism ignores the possible wiring of the circuit, therefore depending only on the initial distribution of the input nodes. Since the SR implementation involves some kind of wiring, one can estimate the possible difference in mismatch cost between the two situations.

Definition 2.17. The **circuit mismatch loss** is defined as the difference in mismatch costs between the SR circuit implementation and a process ignoring the impact of the wiring of the system, called an 'all-at-once' (AO) process, in which there is no Landauer loss term as this AO process avoids variable issues induced by SR circuits. This AO process actually corresponds to the theoretical process skipping all the details of the SR implementation, therefore propagating in a straight forward manner the initial distribution to the ending distribution. In particular, the entropy flow of the AO process answers the following equation :

$$\mathcal{Q}_{AO}(p) = \mathcal{L}(p) + D(p_{IN}||q_{IN}) - D(\pi_\Phi p_{IN}||\pi_\Phi q_{IN}). \tag{2.38}$$

The difference in mismatch costs behind the idealized process and its physical implementation therefore corresponds to the following equation :

$$\mathcal{M}^{\text{loss}}(p_{IN}||q_{IN}) = \mathcal{D}(\pi_\Phi p_{IN}||\pi_\Phi q_{IN}) - \mathcal{D}(p_{IN}||q_{IN}) + \sum_{g\in G\setminus W} \mathcal{D}(p_{\text{pa}(g)}||q_{\text{pa}(g)}). \tag{2.39}$$

Note that the mismatch loss can actually be negative, implying that the physical implementation through the serial-reinitialized circuits may incur less entropy flow than the actual idealized process.

As the preceding sections has been heavy in new concepts, notations and functions to understand, their application to a simple example (taken from [11]) will help to familiarize with the concepts. As most of the earlier notions were introduced to provide a complete understanding of the theory behind the decomposition of the entropy flow, only the Landauer cost as well as the mismatch cost will be computed.

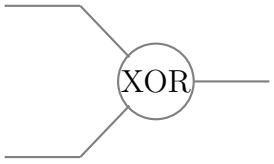


Figure 2.1: XOR gate

Example 2.18. Let's consider a XOR gate, where the parents are designed by $\mathrm{pa}(g)$. Say that the initial distribution is uniform over the set $\mathcal{X}_{IN} = \{00, 01, 10\}$ such that

$$p_{\mathrm{pa}(g)}(x_{\mathrm{pa(g)}}) = (1 - \delta(x_{\mathrm{pa(g)}}, 11))/3.$$

This therefore means that any state in $\mathcal{X}_{IN}$ as the same probability to occur, meaning that 11 has a zero probability of happening. Now consider the optimal distribution to be uniform over $\{00, 01, 10, 11\}$ such that

$$q_{\mathrm{pa}(g)}(x_{\mathrm{pa(g)}}) = \frac{1}{4}.$$

The conditional distribution implemented by the XOR gate is given by

$$\pi_g(x_g|x_{\mathrm{pa}(g)}) = \delta(x_g, \mathsf{XOR}(x_{\mathrm{pa(g)}})),$$

and with this the Landauer cost as well as the mismatch cost can already be computed :

$$\begin{aligned}
\mathcal{L}(p) =& S(p_{\mathrm{pa}(g)}) - S(\pi_g p_{\mathrm{pa}(g)}) = \ln 3 + \left(\frac{1}{3}\ln\frac{1}{3} + \frac{2}{3}\ln\frac{2}{3}\right) && (2.40)\\
\mathcal{M}(p) =& D_{KL}(p_{\mathrm{pa}(g)}||q_{\mathrm{pa}(g)}) - D(\pi_g p_{\mathrm{pa}(g)}||\pi_g q_{\mathrm{pa}(g)}) \\
=& (\ln 4 - \ln 2) - (S(p_{\mathrm{pa}(g)}) - S(\pi_g p_{\mathrm{pa}(g)})). && (2.41)
\end{aligned}$$

Note how the Landauer cost appeared in the computation of the mismatch cost; this is due to the definition of the Kullback Leibler divergence, as it can be rewritten

$$D_{KL}(p||q) = \sum_i p_i \ln\frac{p_i}{q_i} = \sum_i p_i \ln p_i - \sum_i p_i \ln q_i.$$

Second term in Eq. 2.40 refers to the 1/3 probability to have 00, meaning that the output would be 0, and the 2/3 corresponds to the 2/3 probability of having either 01 or 10 as input values. As the sum is over all states, the sum $\sum_i p_i \ln q_i$ is then only equal to $\ln q_i$ as the optimal distribution is uniform. Assuming a heat bath of 20 °C, the corresponding energy emitted would therefore be $k_B \times 293,15K \times (0.46+0.23) = 2.79 \times 10^{-21}$J. Note that the Landauer cost account for the change in entropy of the system as involved in Eq. 2.17, whereas the mismatch cost contributes to the entropy production incurred by the system. This example only covered a single gate; but for a greater circuit involving more gates, the quantities derived are still useful as they describe the behavior of the subsystem, the solitary process run by this gate.

These final results complete the current understanding of how entropy flows and is produced within a circuit; it depicts the consequences of computations in

thermodynamic terms, therefore constituting a solid background in estimating the thermodynamic cost of computations within artificial neural networks. While the Landauer cost provides a lower bound to the energy yielded by running the circuit based on the input distribution, without considering the impact of the serial-reinitialized implementation or what the best distribution could be for this circuit, next chapter will dedicate its vast majority to the understanding of the mismatch cost when applied to neurons in an artificial neural network. Based on the results from Wolpert and Kolchinsky developed and explained in this chapter, the results displayed in the next chapter are new as only binary logical circuits have been covered.

Chapter 3

Thermodynamic cost of computing with artificial neural network

3.1 Artificial neural networks

3.1.1 Artificial neural networks : a reminder

While the reader may be familiar with artificial neural networks (ANN), it is always useful to have a specific idea of what kind of artificial neural networks are considered, as well as their mathematical implementations. This will avoid confusion, as this thesis is only a first step into the adaptation of Kolchinsky and Wolpert's work on thermodynamics of computation applied to ANN's.

Artificial neural networks are computing systems that are similar to biologicial neural networks, as it was the inspiration to create such computing systems. In fact, *neuroplasticity*[1] was the concept [35] used to design a learning hypothesis that would allow the system to organize itself such that the learning could take place. From this, machines were created that implemented such learning, and eventually the *perceptron* was created [36]. The perceptron is an algorithm that is used in supervised learning[2] in order to complete a classification problem.

The perceptron is used for binary classifiers, and returns a binary value, given

[1]Neuroplasticity designates the ability of a neural network to reorganize itself or grow within a brain, which is associated to learning.

[2]Supervised learning means that, given the task to classify or predict the values of a specific feature that is not in the dataset, the machine is given data that is already classified, or in the case of regression has a value associated to the feature it needs to evaluate. It will train itself onto this data, and also verify the reliability of its acquired knowledge by dividing the given data into a training dataset and a validation dataset.

a real vector :

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{else.} \end{cases} \tag{3.1}$$

The vector $\mathbf{w}$ correspond to the *weights* of the algorithm, and b is the bias associated to it. These values are adjusted so that the classification, given a specific x, is most likely correct.

While the perceptron constitutes one *neuron*, the *multilayer perceptron*(MLP) will be used in this thesis as it is a common neural network, but also mimics the structure of directed acyclic graphs considered in [11] when dealing with circuits. The multilayer perceptron is actually composed of several perceptron, put into several layers, such that each perceptron corresponds to one neuron, and the whole system constitutes a neural network. Another important feature of the multilayer perceptron is that for each perceptron, the function f can be another function than the binary-valued function introduced right before; as instead, it can be any *activation function*. Activation functions are functions that mimic the activation of a neuron such as in the regular perceptron, but allow for more flexibility and usefulness depending on the problem given. This usually allows for less complex network as nodes having the adequate activation function are more efficient in their task.

In this work, the structure and algebra of multilayer perceptrons will be considered. It is not the thermodynamics of the learning that will be evaluated, as this would imply to also consider training algorithms such as backpropagation. In such scenarios, the update of weights in itself is a process to be considered in addition to the computation of the neurons, going beyond the scope of the formalism introduced by D. Wolpert in [11]; it would however be interesting to study such scenario, in particular based on elements introduced in [37]. Nevertheless, only the thermodynamics of the evaluation through a multilayer perceptron will be studied, where the computations considered lie within each neuron, each perceptron.

3.1.2 Artificial neuron gate

The translation from the results by [11] to artificial neural network circuits relies on translating 2-inputs binary gates into non-binary n-inputs gates. As can be seen in the illustration below, the output value would therefore not necessarily be a logical circuit gate but rather a function, would it be linear or not.

The Σ represents the learning taking action within the gate (*e.g.* in the case of a perceptron, the output would be $y = \sigma(\sum_{i=1}^{3} x_i w_i + b)$, where w_i are the weights, b is the bias and σ the activation function) . For the rest of this section, learning performed by a perceptron will be considered as the basis of the computations.

In the logical circuit framework, the conditional distribution $\pi_g(x_g|x_{\mathrm{pa}(g)})$ of a XOR gate takes the form

$$\pi_g(x_g|x_{\mathrm{pa}(g)}) = \delta(x_g, \mathsf{XOR}(x_{\mathrm{pa(g)}})). \tag{3.2}$$

Naturally, the ANN translation of the conditional distribution is therefore defined as

$$\pi_g(x_g|x_{\mathrm{pa(g)}}) = \delta(x_g, \sigma(\mathbf{w} \cdot x_{\mathrm{pa(g)}} + b)) \tag{3.3}$$

where $\mathbf{w}$ is the weight vector.

It is important to keep in mind that the artificial neural network considered has already been trained in this framework, meaning its weights are fixed. Actualization of weights isn't considered, and it is assumed that the gate can be considered as a serial-reinitialized circuit. Therefore, the nodes $n(g)$ of the gate evolve according to

$$P_g(x'_{n(g)}|x_{n(g)}) = \pi_g(x'_g|x_{\mathrm{pa(g)}})\Pi_{v\in\mathrm{pa}(g)}\delta(x'_v,). \tag{3.4}$$

The entropy flow can then be computed according to Eq. 2.30. Since π_g implements a deterministic function $f(x) = \sigma(\mathbf{w} \cdot x + b)$, the set of islands is defined as $L(\pi_g) = \{f^{-1}(y) : y \in \mathcal{Y}\}$ where $\mathcal{Y}$ is the set of outputs.

The remaining terms in Eq. 2.30 correspond to the gate's *Landauer cost* and to the gate's *mismatch cost*, the latter being particularly interesting to evaluate in a thermodynamic framework. Indeed, as the mismatch cost depends on both the Kullback–Leibler divergence between the non-optimal and optimal input distributions, and the KL divergence between the non-optimal and optimal output distributions. The resulting quantity relates to the amount of energy that is lost into the system's environment when the computations are performed without using the optimal distributions for the system. It is therefore important to evaluate whether this energetic cost is negligible. Note that in this simple example, only a single gate is considered, whereas a real-world application would involve multiple gates, layered. This extension will be considered later, as a combination of gates taking other gates as inputs; but the fundamental part of this combination has yet to be properly analysed. This will be done through the following sections.

3.2 Landauer and Mismatch cost of a gate in a neural network

3.2.1 Landauer cost

Recall that given a probability distribution p, the Shannon entropy of the distribution is defined as

$$S(p(X)) = -\sum_{x\in\mathcal{X}} p(x)\ln p(x), \tag{3.5}$$

where $\mathcal{X}$ is the set of states. Since the random variables are real numbers, in numerical neural networks they have to be represented by a finite number of bits. Each reachable number in this bit-quantification can be interpreted as a state.

Example 3.1. Consider a 16-bit quantification. There are therefore 2^{16} possible configurations that a random variable X can take. If X takes its value uniformly across the state space, we get that

$$\begin{aligned} S(p(X)) &= -\sum_{x\in\mathcal{X}} \frac{1}{2^{16}}\ln\frac{1}{2^{16}} && (3.6)\\ &= -2^{16}\left(\frac{1}{2^{16}}\ln\frac{1}{2^{16}}\right) && (3.7)\\ &= 16\ln 2. && (3.8) \end{aligned}$$

Note that the above definition of the Shannon Entropy is expressed in *nats*. The corresponding value in bits would therefore be 16 since $1\text{nat} = \frac{1}{\ln 2}\text{bits}$.

This definition of entropy applies for distributions whose set of states are discrete, hence its usual name *discrete entropy*. If the input distribution follows a continuous distribution (*e.g.* a normal distribution), it is more appropriate to consider using the *continuous entropy* :

Definition 3.2. The **continuous entropy**[3] (or **differential entropy**) of a random variable X following a probability density function p is defined as

$$h(p(X)) = -\int_{\mathcal{X}} p(x)\ln p(x)\mathrm{d}x \tag{3.9}$$

where $\mathcal{X}$ is the support of p.

[3]See [17] for further details about continuous entropy.

It is not clear however how this quantity fit the computation of information theoretic quantities associated to an artificial neuron gate, as $\mathcal{X}$ cannot always be considered as continuous, in particular when dealing with numerical artificial neural networks, as real numbers can only be represented by a finite number of bits. It is therefore important to think about a *discretized* framework when considering continuous distributions.

This will be done as described in [17], and details can be retrieved in the appendix. The quantity obtained is expressed as

$$S_{\text{discrete}}(p(X)) = -\sum_{i=1}^{N} p_i^{\Delta} \ln p_i^{\Delta} \tag{3.10}$$

where $p_i^{\Delta} = \int_{i\Delta}^{(i+1)\Delta} p(x)\mathrm{d}x = p(x_i)\Delta$ with Δ being the length of the i^{th} interval from the N intervals in which $\mathcal{X}$ is divided. Since $P(i\Delta \leq X \leq (i+1)\Delta) = \int_{i\Delta}^{(i+1)\Delta} p(x)\mathrm{d}x$, this decomposition will be useful when evaluating information theoretic quantities associated to y (the gate output) based on simulations – that is, when the exact distribution of y cannot be analytically obtained. Note that when the continuous entropy can be analytically obtained for both input and output distributions, the Landauer cost can be obtained, since the $\ln \Delta$ factors cancel each other.

Given the previous expressions of the Shannon entropy, it is now possible to compute the Landauer cost of an artifical neuron gate for any input distribution[4]. The Landauer cost of computing π_g for an initial distribution p is defined by

$$\mathcal{L}(p) = S(p(X_{\text{IN}})) - S(\pi_g p(X_{\text{IN}})) \tag{3.11}$$

where π_g corresponds to the conditional distribution expressed in Eq 3.3. If the second term of 3.11 is not analytically obtainable (due to non-linear activation function or non-Gaussian input distributions), a Monte-Carlo simulation of the gate is run in order to get a representation of the output's distribution. Then, by Eq. 3.10 the Landauer cost of this gate can be obtained[5].

Example 3.3. Consider this simple example of a gate with 3 inputs where the input distribution p is such that $X \overset{iid}{\sim} \mathcal{N}(0, \Sigma)$, with Σ being the covariance matrix , and where the weights of the unbiased neuron are constants given by $a = (a_1, a_2, a_3)$ with $a_i \in]-5, 5[\,, i = 1, 2, 3$, and the activation function the identity.

[4]Note that in the case of layered gates, a gate's Landauer cost is different from the whole system's Landauer cost. When evaluating the whole network Landauer cost, the contribution of each gate must be accounted for.

[5]This method can also be applied to the inputs, if their distribution law is unknown but a histogram representing it is given.

Using the notations introduced at the beginning of this chapter, $\sigma(x) \equiv x$ and the bias $b = 0$. The output is therefore given by $Y = a_1X_1 + a_2X_2 + a_3X_3$ and is such that $Y \overset{iid}{\sim} \mathcal{N}(0, a\Sigma a^T)$. This distribution can be written as π_Φ, which can be also be defined as the conditional distribution of the final joint state of the output gates given the initial joint state of the inputs (See Eq.2.23, as defined in [11]). The continuous entropies associated to the inputs and output respectively are given by :

$$h(p(X_1, X_2, X_3)) = \frac{1}{2}\left(3\ln 2\pi + 3 + \ln\det\Sigma\right), \tag{3.12}$$

$$h(\pi_\Phi p(X_1, X_2, X_3)) = \frac{1}{2}\ln(a\Sigma a^T) + \frac{1}{2}\left(\ln 2\pi + 1\right). \tag{3.13}$$

The resulting Landauer cost of running π_Φ with an initial distribution p as described above is thus

$$\begin{aligned}\mathcal{L}(p) &= h(p(X_1, X_2, X_3)) - h(Y) \\ &= 1 + \ln 2\pi + \frac{1}{2}\ln\left(\frac{\det\Sigma}{a\Sigma a^T}\right).\end{aligned} \tag{3.14}$$

3.2.2 Mismatch cost

The mismatch cost is, by definition, the drop in Kullback-Leibler (KL) divergence between the optimal distribution of inputs of a circuit p_{opt} and the actual distribution of the parent or input nodes p as both evolve under the conditional distribution (π_g in the case of a single gate), namely[6] :

$$\mathcal{M}(p) = D\left(p||p_{\text{opt}})\right) - D\left(\pi_g p||\pi_g p_{\text{opt}}\right). \tag{3.15}$$

This quantity characterizes the entropy flow incurred because the input nodes are not distributed optimally. For a single gate, the optimal distribution is such that it minimizes the entropy production. When these gates are layered, the non-optimality of each gate is considered; but it is wiser to stick with the single gate case for now, as defining what is optimal is non-trivial. In fact, it depends on the underlying technology used for such gate, but one could decide what the best distribution for such a gate should be like.

[6]The reader might highlight the difference between Eq. 3.15 and Eq. 2.31; in the first, the intuitive definition is given, as the first examples will only consider single gates with 2 inputs and 1 output, such that the mismatch cost then computed is the one corresponding to one gate in a circuit, or in other words its contribution to the total mismatch cost. In Eq. 2.31, the sum is over $g \in G\backslash W$, thus nodes that are neither the input or output nodes. As it may bring the confusion, the upcoming examples of the mismatch cost derivation can be perceived as zoom-in's onto the artificial neural network; their contribution to the global mismatch cost. The examples will offer an insight about the mismatch cost behavior through the several layers of a circuit.

Again, it is useful to know how to compute the KL divergence when the distribution functions are analytically unknown but numerically computable. Using the same method as for differential entropy, and assuming both optimal and non-optimal distributions share the same domain[7], the KL divergence can be computed as :

$$D\left(p(x)||p_{\text{opt}}(x)\right) = \sum_{i=1}^{N} p_i^{\Delta} \ln \frac{p_i^{\Delta}}{p_{\text{opt},i}^{\Delta}} \tag{3.16}$$

$$= -S_{\text{discrete}}(p(X)) - \sum_{i=1}^{N} p_i^{\Delta} \ln p_{\text{opt},i}^{\Delta} \tag{3.17}$$

where $p_i^{\Delta} = \int_{i\Delta}^{(i+1)\Delta} p(x)\mathrm{d}x$ and $p_{\text{opt},i}^{\Delta} = \int_{i\Delta}^{(i+1)\Delta} p_{\text{opt}}(x)\mathrm{d}x$. As to compute the Landauer cost, this decomposition into N intervals helps to compute the mismatch cost when the exact distribution of the output cannot be analytically obtained. It is important however to highlight that in the case of the KL divergence, the optimal distribution must share the same support as the real distribution for it to make sense; as it aims to evaluate how much two distributions differ from one another, it is only normal – to make a valid comparison – that the KL divergence is computed over the same support, such that both distributions take values always greater than zero.

Some hypothesis were introduced on what should the optimal distribution look like. The first hypothesis was the normal law $\mathcal{N}(0, \mathbb{1})$ for the inputs, as it fits the assumption that the input data follow a normal law. This would therefore correspond to a specific choice of circuit, for which the mismatch cost is zero when the input distribution is $\mathcal{N}(0, \mathbb{1})$.

The second hypothesis states that the optimal law follows a uniform, as it is the least informative and therefore has the most entropy over its domain. Choosing maximum entropy probability distribution as optimal law is a coherent choice when no prior information on optimal distributions is available; these are the least informative distributions and thus allow to determine a bound relative to future priors.

The upcoming discussions and examples will offer an insight over subsystems within a circuit and how their architecture influences the energetic loss of not running the most fitting distribution for the circuit. This most fitting distribution depends on the circuit; as it is valid to assume that the inputs – the data that is fed to the neural network – will follow a normal law, one may choose to construct circuits such that the normal law is the most fitting distribution – the optimal

[7]This holds as the distributions considered take their values in $\mathbb{R}$; when dealing with a uniform distribution (*e.g.* $U(]-5, 5[)$) only the intersection of their domains is considered as it would be 0 outside this interval.

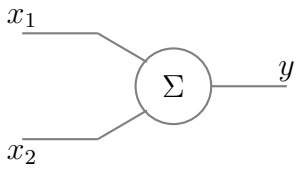


Figure 3.1: 2-input ANN gate

distribution. As the discussions and examples serve as illustration to how the mismatch cost behave through one or more gates, and not an entire circuit, the notion of *input nodes* and their state x_{IN} as used in Chapter 2 must be left aside, as well as the *output nodes* and their state x_{OUT}. In fact, the IN and OUT indices refer to the input and output distribution of the *whole* circuit, namely the very beginning and the very end respectively. As Eq. 2.31 will be considered to compute the mismatch cost of each configuration, all nodes g illustrated in the upcoming examples will be considered as gates, such that $g \in G$.

First approach : Normal law is optimal

Considering that the optimal distribution of the inputs is such that $X_{\text{opt}} \overset{iid}{\sim} \mathcal{N}(0, \mathbb{1})$, given the vector of weights $a = (a_1, \ldots, a_n)$ the optimal distribution of the outputs should be $Y_{\text{opt}} \overset{iid}{\sim} \mathcal{N}(0, |a|^2)$.

Assuming the inputs also follow a normal law p with mean μ and covariance matrix Σ, *i.e.* $X \sim \mathcal{N}(\mu, \Sigma)$, then the outputs should also follow a normal law, namely $Y \sim \mathcal{N}(a \cdot \mu, a\Sigma a^T)$. Given the distributions of both outputs and inputs, their KL divergence with the optimal case can be computed as follows :

$$D\left(x||x_{\text{opt}}\right) = \frac{1}{2}\left(\text{tr}\Sigma + \mu^T\mu - k - \ln\det\Sigma\right), \tag{3.18}$$

$$D\left(y||y_{\text{opt}}\right) = \frac{1}{2}\left(\frac{a\Sigma a^T}{|a|^2} + (a\cdot\mu)^2 - 1 - \ln\frac{a\Sigma a^T}{|a|^2}\right), \tag{3.19}$$

where k is the dimension of the input vector, or in other words the number of inputs. The mismatch cost is then simply the difference between Eq. 3.18 and 3.19 :

$$\mathcal{M}(p) = \frac{1}{2}\left(\text{tr}\Sigma - \ln\det\Sigma - (k-1) - \frac{a\Sigma a^T}{|a|^2} - (a\cdot\mu)^2 + \ln\frac{a\Sigma a^T}{|a|^2}\right). \tag{3.20}$$

This computed mismatch cost relates to the induced loss of energy when the input distribution isn't the optimal distribution for this specific gate, assuming that the normal distribution with a mean $\mu = 0$ and the identity as covariance matrix,

meaning that variables are decorrelated, is the optimal distribution. There is a singularity in the mismatch cost function when $\det \Sigma = 0$ or $a\Sigma a^T = 0$, and by definition of covariance matrix Σ is positive semi-definite, meaning that $\det \Sigma \geq 0$ and $x\Sigma x^T \geq 0,\ \forall x \in \mathbb{R}^k$. However, when Σ is singular, meaning that $\det \Sigma = 0$, there is no proper probability density function; rather, in this extreme case one can assume that a delta function covers a subspace satisfying this extreme condition (*e.g.* For $k = 2$, given $X_1 \sim \mathcal{N}(0, 1)$ and $X_2 = -X_1$, then the covariance matrix takes the form $\Sigma = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$, and the determinant is zero. The probability density function therefore is zero everywhere, except on the line where $X_1 = -X_2$). Such an extreme case is therefore quite unlikeley, but rises some interrogations in terms of physical interpretations : can the mismatch cost reach an infinite amount of energy loss, or is it somehow limited due to some constraint? While these questions have to be addressed, let's first dive into the study in non-extreme cases of the mismatch cost function.

Example 3.4. To illustrate the result obtained above, the explicit form taken by the mismatch cost for a gate with two inputs where the activation function is the identity and the covariance matrix is defined by $\Sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & \sigma_{22} \end{pmatrix}$, is given by :

$$\begin{aligned} \mathcal{M}(p) = \frac{1}{2}\Big[&(\sigma_{11} + \sigma_{22}) - \ln\left(\sigma_{11}\sigma_{22} - \sigma_{12}^2\right) - 1 \\ &- \frac{a_1^2\sigma_{11} + 2a_1a_2\sigma_{12} + a_2^2\sigma_{22}}{|a|^2} - (a \cdot \mu)^2 \\ &+ \ln \frac{a_1^2\sigma_{11} + 2a_1a_2\sigma_{12} + a_2^2\sigma_{22}}{|a|^2}\Big]. \end{aligned} \tag{3.21}$$

The behavior of the mismatch cost is illustrated in Fig. 3.2 below. One can see that when X_1 and X_2 are highly correlated, the mismatch cost grows drastically. This divergence is explicitly described by the logarithmic term involving the determinant of the covariance matrix. On the other hand, when both variables are independent from each other, the mismatch cost is minimal. In Fig. 3.2, the variance of both variables is set to 1; but when they follow a law with larger variance the mismatch cost is bound to be higher than zero, even in the case of independence, since the variance does not match the optimal one. The lower bound is therefore higher up in the graph, as can be seen in Fig. 3.3. One can also notice how the minimum shifted from a correlation of zero to a non-zero one; this is due to the weights also playing their part in the behavior of the mismatch cost. An extreme case of the influence of the weights occurs when both weights share the same absolute value, which can be seen mathematically as Eq. 3.21 reduces to (assuming that $\mu = (0, 0)$

) :

$$\mathcal{M}(p) = \frac{1}{2}\Big[\frac{1}{2}\sigma_{11} + \frac{1}{2}\sigma_{22} + \ln(\frac{1}{2}\sigma_{11} + \frac{1}{2}\sigma_{22} + \mathrm{sgn}(a_1 a_2)\sigma_{12}) \\ - \ln(\sigma_{11}\sigma_{22} - \sigma_{12}^2) - 1 - \mathrm{sgn}(a_1 a_2)\sigma_{12}\Big] . \tag{3.22}$$

The mismatch cost is then highly dependent on the variance on the interval $[-1, 1]$, as can be seen in Fig. 3.4.

One can compare the value of the mismatch cost in comparison to the Landauer cost of the gate, as both share the same unit, as illustrated in Fig. B.4 for the same parameters as in Fig. 3.3.

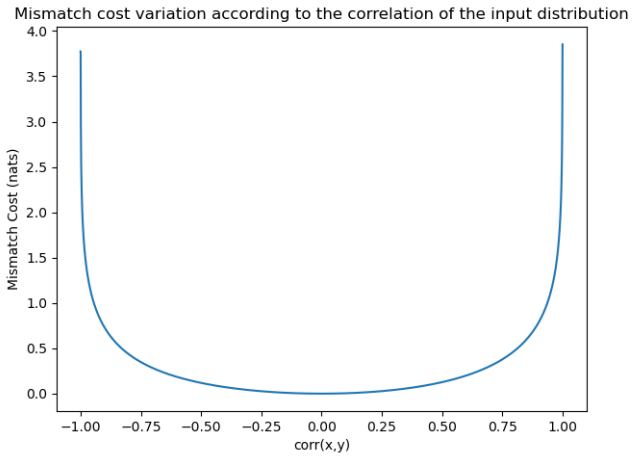


Figure 3.2: $\mathcal{M}(p)$ for the 2-input ANN gate, with weights $= [-0.1, 3]$, $\sigma_{11} = 1$ and $\sigma_{22} = 1$.

Another variable of interest in the study of the mismatch cost for a single gate would be the variance of the inputs. Since the mismatch cost depends similarly for both variances, the only possible difference between the two behaviors lies in the value of the weights, and the correlations between the two variables.

Let's first consider two weakly correlated variables ($\rho_{12} \equiv \mathrm{corr}(X_1, X_2) = 0.1$). For equal weights, the mismatch cost evolves equally under the variation of both variances, which is shown in Fig. 3.5. One can see that for values of variance that are above the optimal variance (thus greater than 1), and up to some variance

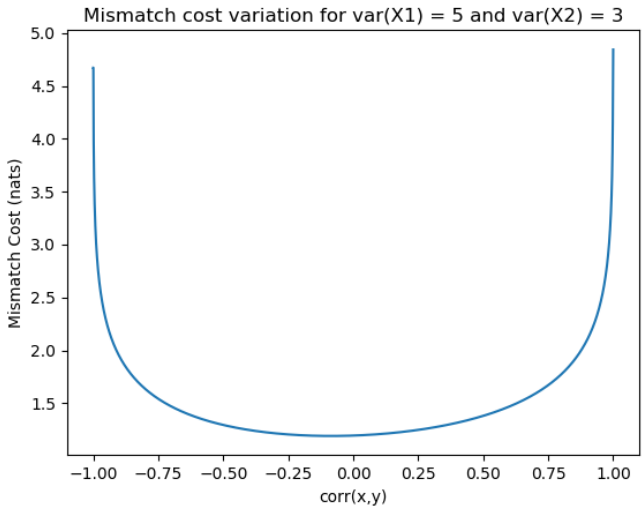


Figure 3.3: $\mathcal{M}(p)$ for the 2-input ANN gate, with weights $= [-0.1, 3]$, $\sigma_{11} = 5$ and $\sigma_{22} = 3$.

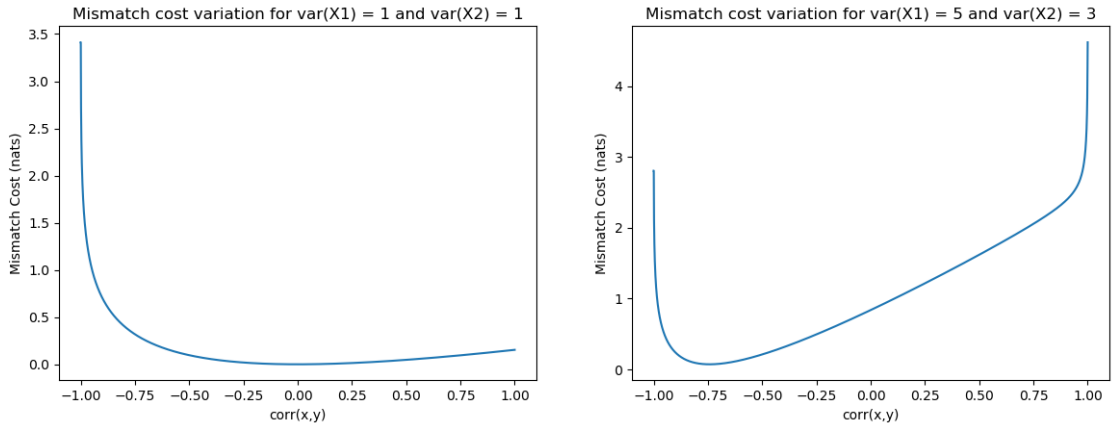
taken by both inputs such that

$$\frac{\partial \mathcal{M}(p)}{\partial \sigma_{11}} = \frac{1}{2}\left(1 - \frac{1}{\sigma_{11}} - \frac{a_1^2 - a_1 a_2\sqrt{\sigma_{22}}\rho_{12}}{|\mathbf{a}|^2} + \frac{a_1^2 + a_1 a_2\sqrt{\sigma_{22}}\rho_{12}}{a_1^2\sigma_1 1 + 2a_1 a_2\sigma_{12} + a_2^2\sigma_{22}}\right) = 0, \tag{3.23}$$

$$\frac{\partial \mathcal{M}(p)}{\partial \sigma_{22}} = \frac{1}{2}\left(1 - \frac{1}{\sigma_{22}} - \frac{a_2^2 - a_1 a_2\sqrt{\sigma_{11}}\rho_{12}}{|\mathbf{a}|^2} + \frac{a_2^2 + a_1 a_2\sqrt{\sigma_{11}}\rho_{12}}{a_1^2\sigma_1 1 + 2a_1 a_2\sigma_{12} + a_2^2\sigma_{22}}\right) = 0, \tag{3.24}$$

the mismatch cost corresponds to a hyperbolic paraboloid. Obviously, it is clear that the extremas introduced right before are minimas, as is clearly shown in Fig. 3.5. When a weight is lower than the other, the plane shifts, such that the dependency of the mismatch cost relies only on one variable's variance, such as shown in Fig. 3.5. In this example, one can see that for each value of var(X_2), the mismatch cost can take an infinite amount of values, while on the other hand the set of values it can take for var(X_1) is rather limited. When the weight for the variable X_2 is much greater than the other, the cut-away view from var(X_1) is approximately a thin line, meaning that the mismatch cost is heavily dependent on the variance of the variable associated to this weight.

When using more correlated inputs, one can see that the second minima (the one that matches higher values of variances) gets closer to the other minima, as this part of the hyperplane "folds" itself closer to the center of the graph. This is because of the restriction in domain due to the correlation factor restricting the



a) $\sigma_{11} = 1$, $\sigma_{22} = 1$ b) $\sigma_{11} = 5$, $\sigma_{22} = 3$

Figure 3.4: $\mathcal{M}(p)$ of the 2-inputs ANN gate for weights $= [-1, 1]$ and different variances.

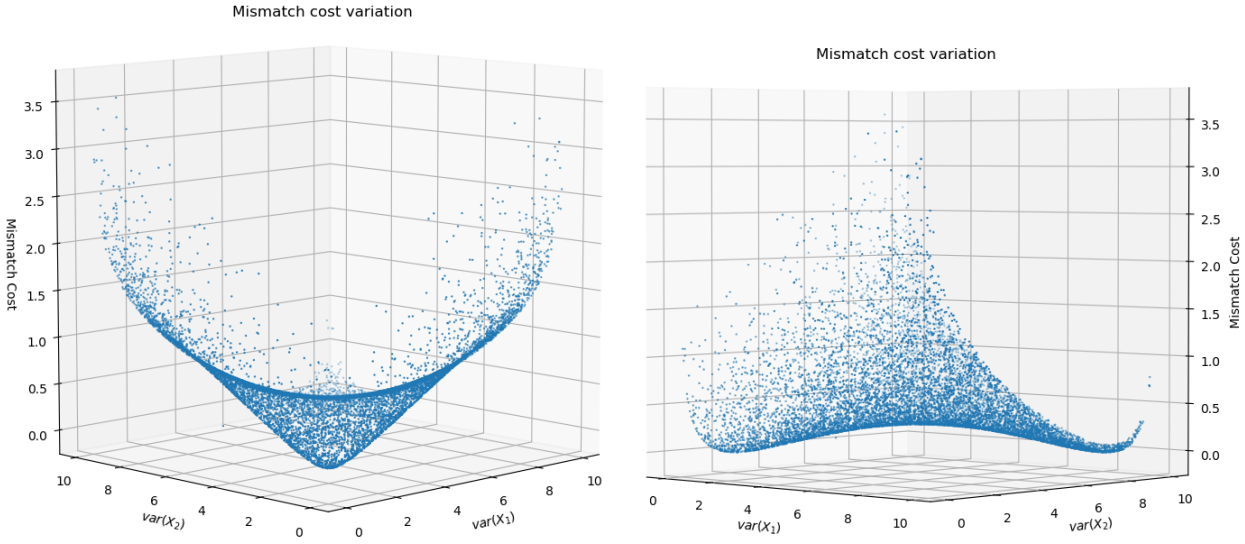
possible values of the variance. The reader is invited to have a look to appendix B to see how the mismatch cost evolves in terms of the variances according to correlation between the two input variables. When changing the weights, the plane still shifts in the same fashion, keeping the "u" shape defined by the plane at some spots.

Second approach : Uniform law is optimal

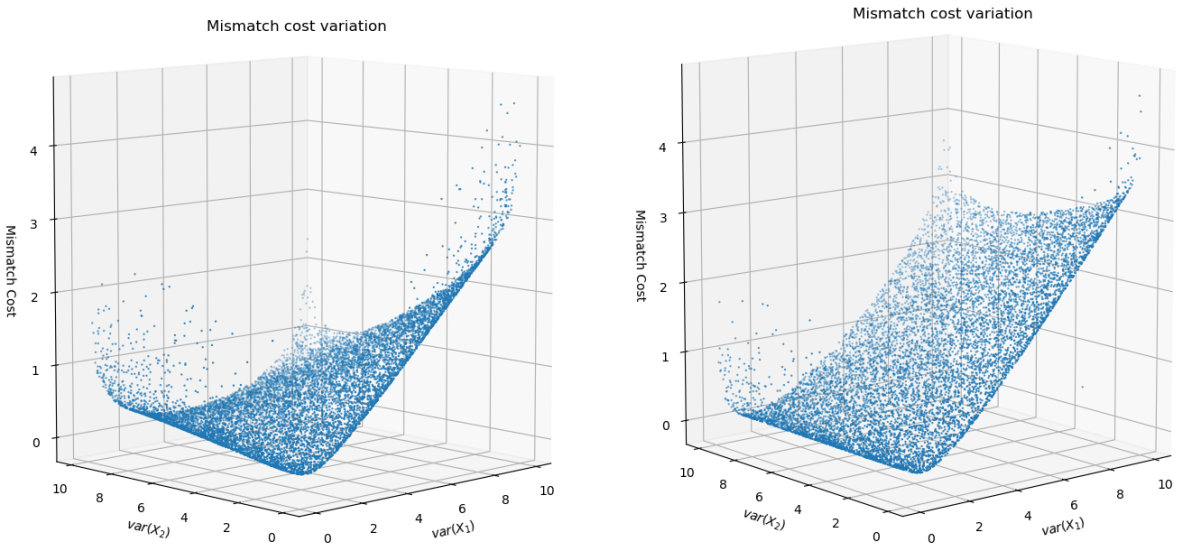
Assume now that the distribution yielding the minimum cost when it is run through the circuit is the uniform distribution. In this framework, Monte-Carlo simulations constitute a solid basis in understanding the behavior of the mismatch function for non-gaussian optimal distributions[8], as it becomes hard to analytically describe the function. Such simulations will also be useful when dealing with non-trivial activation function; however for the sake of this introductory approach, the activation function considered is still the identity.

To perform such simulations, given a random vector of weights $\mathbf{a} = (a_1, a_2)$ which is fixed, the KL divergence of both inputs and output distribution with respect to the optimal distribution in each case needs to be computed. Since the distribution of the output is non-trivial, Monte Carlo simulations offer a proper approximation of such distribution. Furthermore, since it is assumed that the state space is discrete (due to the fact that real numbers cannot be properly implemented;

[8]Since the support of the uniform distribution is bounded, the input distribution is bounded on the same interval.



a) $\mathcal{M}(p)$ of the 2-inputs ANN gate according to variance for weights = $[1, 1]$



b) weights = [1,2]

c) weights = [1,5]

Figure 3.5: $\mathcal{M}(p)$ of the 2-inputs ANN gate according to variance for different weights

instead, they are limited in precision, which defines the limit at which the space is discretized). Equation 3.17 is therefore used to compute the KL divergence of the ouptut, based on the simulations. Given an input distribution that follows a normal law with fixed variance, the behavior of the mismatch cost is illustrated in Fig. 3.6.

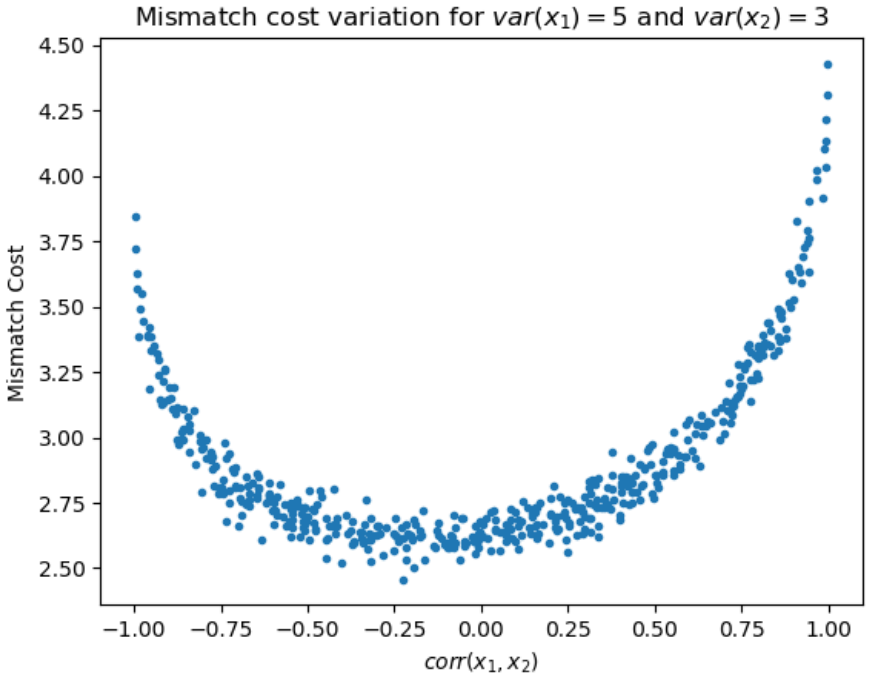


Figure 3.6: Mismatch cost of an ANN gate where the uniform distribution over the interval $[-1, 1]$ is assumed to be the optimal law.

As expected, highly correlated variables induce a higher mismatch cost; however independent variables still yield some non-zero mismatch cost, due to the nature of their distribution. In fact, since the optimal distribution in this case is the one yielding the most entropy (as it is the least predictable), the input distribution follows a law that does not match the optimal one in terms of stochasticity. This stochastic mismatch therefore yields a lower bound to the mismatch cost of the gate.

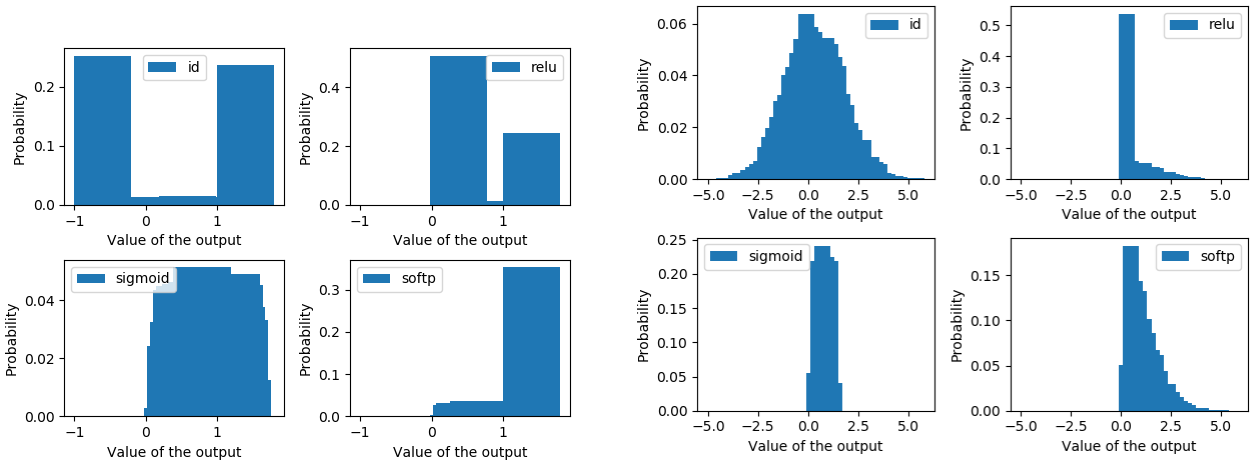
The impact of activation functions

Activation functions are crucial in neural network architectures, as they indicate how a node in the network should activate and consider the received inputs, and

thus define its output. Estimating how the activation function enhances or dampens the energetic loss due to the mismatch cost is important as it may imply to think what the appropriate activation function is considering its energetic impact. Here, 4 different cases are considered, namely :

- The identity (no activation function) : $\sigma(x) = x$,
- ReLu : $\sigma(x) = \max(0, x)$,
- Softplus : $\sigma(x) = \ln(1 + e^x)$,
- Sigmoid : $\sigma(x) = \frac{1}{1+e^{-x}}$.

An illustration of the output distributions according to the activation function is displayed in Fig. 3.7. These distributions represent the propagation of uncorrelated inputs $(X_1, X_2) \overset{iid}{\sim} \mathcal{N}(0, \mathbb{1})$ through a neuron with weights $\mathbf{a} = [1, 1]$ over the interval $[-1, 1]$ and $[-5, 5]$ respectively. One can see that the bound imposed on the values taken by the input variables imply a saturation of the distributions at those bounds, which derives from the cumulative probability density of outside the bounds.



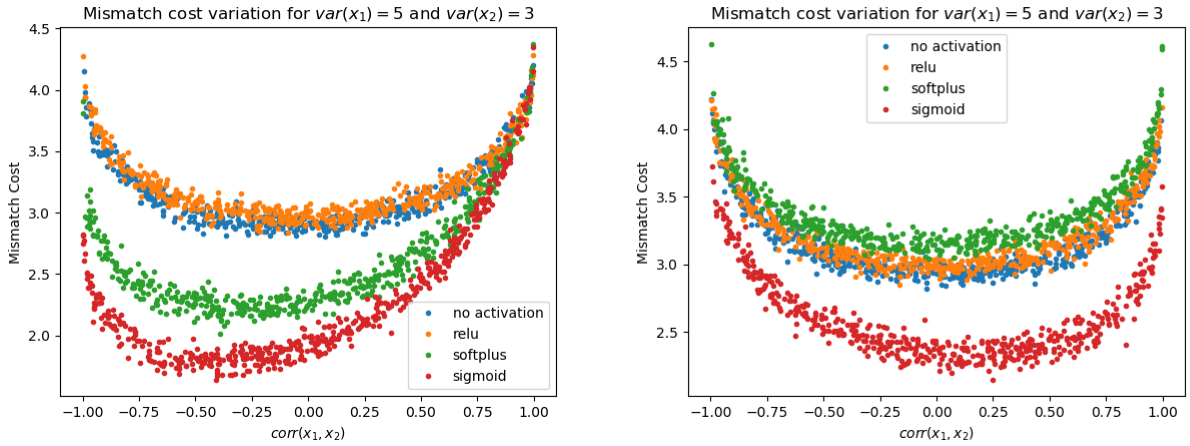
a) Over $[-1, 1]$ b) Over $[-5, 5]$

Figure 3.7: Distribution of the outputs of a single neuron ANN

The simulations are run with the uniform distribution considered as prior distribution, since it is the one yielding the most entropy.

In the case of a single gate for inputs that are bounded to the interval $[-1, 1]$, one can see that the sigmoid, also known as *softstep*, which is monotonic and turns the value taken by the activation function into a positive yet quite small value

as the value increases, yields a lower mismatch cost for weakly correlated inputs; an interesting thing to point out is that the minimum for the sigmoid activation function isn't the same as for ReLu and the identity as the activation function, which is due to the weights of the gate. Note that not only the sigmoid activation function is dependent on the weights, but also the softplus as it can be both cheaper and costlier energetically than when no activation function is considered (*i.e.* using the identity as an activation function). The sigmoid therefore takes the cake, as it is consistently cheaper than the other activation functions; this can be explained by the low values that are obtained through the sigmoid activation function, yielding a smooth distribution (as can be seen in Fig. 3.7) and thus getting closer to the optimal distribution as the uniform distribution is used as prior distribution in the numerical simulations. Note that for values that are bounded to a greater interval such as $[-5, 5]$, the activation functions don't have an impact on the mismatch cost, as illustrated in Fig. B.5.



a) weights $= [1.5, -3]$ b) weights $= [0.5, 3]$

Figure 3.8: $\mathcal{M}(p)$ of the 2-inputs ANN gate for different activation functions

Maximal mismatch cost

The extreme case of totally correlated input variables implies the maximal mismatch cost; but what is that maximum physically speaking ?

While it is true in a mathematical point of view that the mismatch cost tends to infinity when the determinant of the covariance matrix tends to zero (thus when the correlation factor tends to 1), the amount of energy lost must not be infinite; it doesn't make sense to lose an inconceivable amount of energy by having inputs that

are totally correlated, otherwise the process would simply never happen. There must be some kind of limit, induced by the logical and stochastic nature of the system.

As the Landauer principle and the Landauer bound implies some minimum heat to be generated when bringing an initial distribution to an ending distribution through some kind of process, it is expected that such maximum bound is defined by the process. Several parameters are to be considered, namely the state space and the architecture of the system – here, a gate. Real numbers in the mathematical sense cannot be properly implemented, as some limit to their digits always exists when using computers. The state space is therefore finite, and based on the maximum word size of the processor, say for example 32-bits, the size of the state space is therefore 2^{32}. The other important factor to consider, would it be for a single gate or a greater network, is the number of nodes in which computation take place. Therefore, for each neuron, it is expected that the energetic cost associated mismatch cost can only reach a value equal to the maximum entropy for each nodes, and then multiplied by the number of nodes. This will give the mismatch cost in terms of nats or bits, depending on the base of the logarithm. Considering the example used right before, and using a base 2 logarithm, the maximal energetic cost linked to the mismatch cost would therefore be :

$$\mathcal{M}_{max} = 32 \times \#\text{ neurons} \times k_B T \tag{3.25}$$

where T is the temperature of the heat bath and k_B is the Boltzmann constant. This maximal bound comes from the fact that the distribution yielding the most entropy, the uniform distribution, is to be considered to be the first prior to consider in general [11], by the *principle of maximum entropy*[9] [38][39].

Extension the single-gate case

These results obtained for a single gate can be perceived as a building block for other more complex yet concrete examples. In fact, the mismatch cost obtained is simply the contribution of one gate to the whole neural network; as the contribution of each gate needs to be considered, one could evaluate how the mismatch cost behave throughout such network. In order to study the behavior of the mismatch cost according to the size of the network, assumptions concerning the optimal distributions must be taken : is the optimal distribution actualized based on the preceding gate or is it the same for each gate? In other words, given an optimal distribution p_{opt} for a gate A, is the optimal distribution for the following gate the

[9]This principle states that considering all prior information and current knowledge about the possible probability distribution, the one yielding the maximal information entropy is the one to choose to describe the system, as it is the one that leaves the most room to additive information and is the less likely to induce false assumptions.

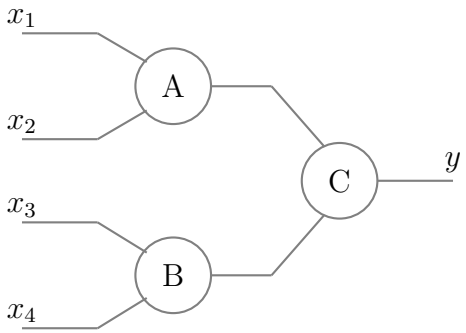


Figure 3.9: 4-input simple ANN

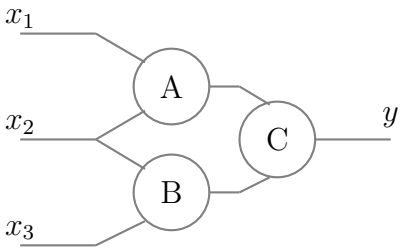


Figure 3.10: 3-input simple ANN

same or does it depend on $\pi_g p_{\text{opt}}$? Both cases will be covered, but while the first situation might be interesting theoretically[10], it may be that, physically, the second option is the closest to reality as the same factory parts can be used. Different architectures of network will be considered; one where an input is involved in two gates, and the other where there is no "overlap". Both networks are simple model, but can be extended and offer the first step towards the evaluation of the mismatch cost function behavior according to the number of layers.

Note that the SR implementation as explained in Chapter 2 states that once a gate is run, the parents of that gate are reinitialized. In neural networks, each neuron usually contributes to more than one neuron in the next layer; in that sense, for each non-output node v, it is assumed that only the last of its children reinitializes its state, while the other children apply the identity map to v, as is explained in Appendix C of [11].

[10]This scenario is the one considered in [11] and [10], as the term $\pi_g p_{\text{opt}}$ denotes the propagation of the optimal initial distribution. The following scenario is one that remains interesting due to the "factory" approach of creating the appropriate hardware for the artificial neuron; while the optimal distribution should nonetheless be determined using prior informations, it is still interesting to compare the differences in both scenarios.

3.3 Beyond the single gate case

Adding another layer between the inputs and the output also adds a layer of complexity in the analysis of the mismatch cost function. For example, in cases illustrated by Fig. 3.10 and 3.9, the increasing number of inputs implies an increasing number of variables to consider, namely the correlation between inputs that do not interact in the first layer. In cases such as in Fig. 3.10 where an input contributes to several gates, its variance is also a variable to take into account, as the upcoming results concerning the mismatch cost of each gate will illustrate.

The inputs are assumed to follow a normal law $X \sim \mathcal{N}(\mu, \Sigma)$ where μ is the vector of means and Σ the covariance matrix. In terms of notations, σ_{ii} will depict the variance of input variable x_i, and σ_{ij} its covariance with variable x_j.

Putting the architecture aside, the optimal law for each gate in this extended case is also an important factor to consider. Two scenarios are to be considered :

- The optimal law for subsequent gates corresponds to the law resulting from the outputs, given that the inputs followed the optimal law. The optimal distributions were obtained by propagating the optimal input distribution in the network, written as $\pi_g p_{\text{opt}}$ (*e.g.* Considering a similar case as Fig. 3.9 : assuming, for this analytical analysis, that the optimal inputs follow a normal law $X_{\text{opt}} \sim \mathcal{N}(0, \mathbb{1})$, then, dealing with the identity as the activation function, and considering no bias, the output of gate A is such that $Y_A \sim \mathcal{N}(0, |\mathbf{a}|^2)$, with $\mathbf{a}$ the weights of gate A. Inputs of the second layer therefore follow a normal law where the covariance matrix isn't the identity matrix);

- The optimal law is the same for each inputs of each gate, meaning that inputs of the second layer should optimally follow the same law as inputs of the gates in the first layer.

This distinction brings non-trivial consequences, as the mismatch costs of the final gates in the first case depend on a lot more parameters.

3.3.1 Analytical results for 2 layers

Let's consider an extended network as in Fig. 3.10 (3 inputs) and 3.9 (4 inputs), with an additional layer. Each layer will respectively be depicted with variables X, Y and Z. The input layer is constituted of two gates, namely gate A and B. The inputs of these gates are denoted X_A and X_B, while their outputs are denoted Y_A and Y_B. The final gate C take as inputs Y_{inp} which follows the joint distribution of Y_A and Y_B, and the final output is denoted as Z. When considering the optimal distribution for each gate, the variables will be denoted explicitely as $X_{A,\text{opt}}, X_{B,\text{opt}}, Y_{A,\text{opt}}, Y_{B,\text{opt}}, Y_{\text{inp,opt}}, Z_{\text{opt}}$.

Evolution of the input distribution across the network

Given any inputs x_1, x_2, the resulting output of a gate such as gate A is given by $y = \sigma(\sum_{i=1}^{2} a_i x_i + b_0)$ where σ is the activation function, b_0 the bias and $\mathbf{a} = (a_1, a_2)$ are the weights of gate A. Since $\sigma(x) \equiv x$, and $b_0 = 0$, one can find the distribution law of the output.

Starting with inputs such that $X \sim \mathcal{N}(\mu, \Sigma)$, given two gates A and B, one can split the covariance matrix of the inputs into two sub-matrices for each gate's inputs, namely Σ_A and Σ_B. In the same fashion, the mean vector μ can be split into two sub-vectors μ_A and μ_B, containing the mean of each inputs that are involved in gate A or B respectively. The covariance matrices of both gates gather the covariance and variance of their respective inputs. Since $y_A = \mathbf{a} \cdot \mathbf{x}_A$, (*e.g.* $y_A = a_1 x_1 + a_2 x_2$ in both Fig. 3.10 and 3.9), its distribution law can be obtained as Y_A is a linear combination of two gaussian random variables. One can find that $Y_A \sim \mathcal{N}(\mathbf{a} \cdot \mu_A, \mathbf{a}\Sigma_A\mathbf{a}^T)$ and $Y_B \sim \mathcal{N}(\mathbf{b} \cdot \mu_B, \mathbf{b}\Sigma_B\mathbf{b}^T)$ where $\mathbf{a}$ and $\mathbf{b}$ are the weights of gate A and B, respectively.

Both outputs need to be considered jointly in order to find the distribution law of the final output Z. Correlations between the initial inputs that weren't taken into account in the first layer will define the covariance matrix of the final layer's inputs. Since the examples consider 2-dimensional inputs, this "global" covariance factor that will be denoted ξ will only appear in the off-diagonal. For the sake of simplicity, the elements of the covariance matrix will be denoted as $\sigma_{i,j}$ for the covariance between X_i and X_j; when $i = j$ it will simply correspond to the variance of X_i. The value of ξ varies between the 3-inputs case as in Fig. 3.10 and the 4-inputs case :

3-inputs : The covariance between Y_A and Y_B is given by

$$\begin{aligned}
\xi &\equiv \mathrm{cov}(Y_A, Y_B) && (3.26)\\
&= \mathrm{cov}(a_1 X_1 + a_2 X_2, b_1 X_2 + b_2 X_3) && (3.27)\\
&= a_1 b_1 \mathrm{cov}(X_1, X_2) + a_1 b_2 \mathrm{cov}(X_1, X_3) \\
&+ a_2 b_1 \mathrm{cov}(X_2, X_2) + a_2 b_2 \mathrm{cov}(X_2, X_3) && (3.28)\\
&= a_1 b_1 \sigma_{12} + a_1 b_2 \sigma_{13} + a_2 b_2 \sigma_{23} + a_2 b_1 \sigma_{22}. && (3.29)
\end{aligned}$$

The variance of X_2, σ_{22}, therefore plays a greater role than the variance of the other inputs in this scheme, as X_2 contributes to both input gates displayed in Fig. 3.10.

4-inputs : The covariance between Y_A and Y_B is given by

$$\begin{aligned}\xi &\equiv \text{cov}(Y_A, Y_B) && (3.30)\\ &= \text{cov}(a_1X_1 + a_2X_2, b_1X_3 + b_2X_4) && (3.31)\\ &= a_1b_1\text{cov}(X_1, X_3) + a_1b_2\text{cov}(X_1, X_4)\\ &+ a_2b_1\text{cov}(X_2, X_3) + a_2b_2\text{cov}(X_2, X_4) && (3.32)\\ &= a_1b_1\sigma_{13} + a_1b_2\sigma_{14} + a_2b_1\sigma_{23} + a_2b_2\sigma_{24}. && (3.33)\end{aligned}$$

Obviously, with the increasing number of inputs, the number of parameters to take into account when computing the mismatch cost will increase, and this is already demonstrated by the number of parameters involved in the covariance between the output of gate A and the output of gate B. Someone familiar with ANNs and the multilayer perceptron will note that Fig. 3.10 is more likely to correctly represent a neural network; this will be discussed later in this work, however the 4-inputs case is still interesting in order to evaluate the contribution of correlations for gates that were initially disconnected, in the sense that their inputs were not in common.

Sticking with ξ for the sake of convenience and generalization, the law of distribution of the inputs of the last layer is given by $Y_{\text{inp}} \sim \mathcal{N}(\mu', \Sigma')$ where $\mu' = \begin{pmatrix}\mathbf{a}\cdot\mu_A\\ \mathbf{b}\cdot\mu_B\end{pmatrix}$ and $\Sigma' = \begin{pmatrix}\mathbf{a}\Sigma_A\mathbf{a}^T & \xi\\ \xi & \mathbf{b}\Sigma_B\mathbf{b}^T\end{pmatrix}$. Given $\mathbf{c}$ the weights of the final gate, the distribution of the output is therefore $Z \sim \mathcal{N}(\mathbf{c}\cdot\mu', \mathbf{c}\Sigma'\mathbf{c}^T)$. Now that all possible distributions are known given the inputs, the KL-divergence for each input and output can be obtained. The only thing to determine before computing the exact values is whether the optimal distribution evolves across the network or not.

Actualized optimal distribution

In this scenario, the optimal inputs are independent, and follows a normal law such that $X_{\text{opt}} \sim \mathcal{N}(0, \mathbb{1})$. Plugin this into the results that were just obtained, one gets that $\sim \mathcal{N}(0, \mathbf{a}\mathbb{1}\mathbf{a}^T = |\mathbf{a}|^2)$ and $Y_{B,\text{opt}} \sim \mathcal{N}(0, |\mathbf{b}|^2)$. Since the inputs are independent, the only scenario where ξ_{opt} is not zero is when at least one input contributes to two gates, as in Fig.3.10. There's no reason to remove ξ_{opt} from the result, therefore

$$Y_{\text{inp,opt}} \sim \mathcal{N}\left(0, \Sigma'_{\text{opt}} = \begin{pmatrix}|\mathbf{a}|^2 & \xi_{\text{opt}}\\ \xi_{\text{opt}} & |\mathbf{b}|^2\end{pmatrix}\right). \quad (3.34)$$

The final output distribution is therefore given by

$$\begin{aligned}Z_{\text{opt}} &\sim \mathcal{N}(0, \mathbf{c}\Sigma'_{\text{opt}}\mathbf{c}^T) && (3.35)\\ &\sim \mathcal{N}(0, c_1^2|\mathbf{a}|^2 + 2c_1c_2\xi_{\text{opt}} + c_2^2|\mathbf{b}|^2) && (3.36)\end{aligned}$$

Computing the contribution of each gate to the mismatch cost as indicated in Eq. 3.15, the several KL divergences that are involved are given below :

$$D_{KL}(X_A||X_{A,opt}) = \frac{1}{2}\left(\mathrm{tr}\Sigma_A + \mu_A^T\mu_A - 2 - \ln\det\Sigma_A\right) \tag{3.37}$$

$$D_{KL}(X_B||X_{B,opt}) = \frac{1}{2}\left(\mathrm{tr}\Sigma_B + \mu_B^T\mu_B - 2 - \ln\det\Sigma_B\right) \tag{3.38}$$

$$D_{KL}(Y_A||Y_{A,opt}) = \frac{1}{2}\left(\frac{\mathbf{a}\Sigma_A\mathbf{a}^T}{|\mathbf{a}|^2} + \frac{(\mathbf{a}\cdot\mu_A)^2}{|\mathbf{a}|^2} - 1 - \ln\frac{\mathbf{a}\Sigma_A\mathbf{a}^T}{|\mathbf{a}|^2}\right) \tag{3.39}$$

$$D_{KL}(Y_B||Y_{B,opt}) = \frac{1}{2}\left(\frac{\mathbf{b}\Sigma_B\mathbf{b}^T}{|\mathbf{b}|^2} + \frac{(b\cdot\mu_B)^2}{|\mathbf{b}|^2} - 1 - \ln\frac{\mathbf{b}\Sigma_B\mathbf{b}^T}{|\mathbf{b}|^2}\right) \tag{3.40}$$

$$D_{KL}(Y||Y_{opt}) = \frac{1}{2}\left(\mathrm{tr}(\Sigma'^{-1}_{\mathrm{opt}}\Sigma') + \mu'\Sigma'^{-1}_{\mathrm{opt}}\mu'^T - 2 - \ln\frac{\det\Sigma'}{\det\Sigma'_{\mathrm{opt}}}\right) \tag{3.41}$$

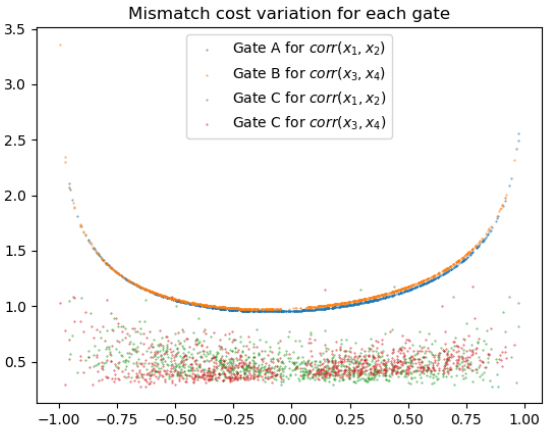
$$D_{KL}(Z||Z_{opt}) = \frac{1}{2}\left(\frac{(\mathbf{c}\cdot\mu')^2}{\mathbf{c}\Sigma'_{\mathrm{opt}}\mathbf{c}^T} + \frac{\mathbf{c}\Sigma'\mathbf{c}^T}{\mathbf{c}\Sigma'_{\mathrm{opt}}\mathbf{c}^T} - 1 - \ln\frac{\mathbf{c}\Sigma'\mathbf{c}^T}{\mathbf{c}\Sigma'_{\mathrm{opt}}\mathbf{c}^T}\right) \tag{3.42}$$

The mismatch cost of each gate is then simply given by the KL divergence from the optimal distribution of the input distribution of the gate, minus the KL divergence of its outputs, namely :

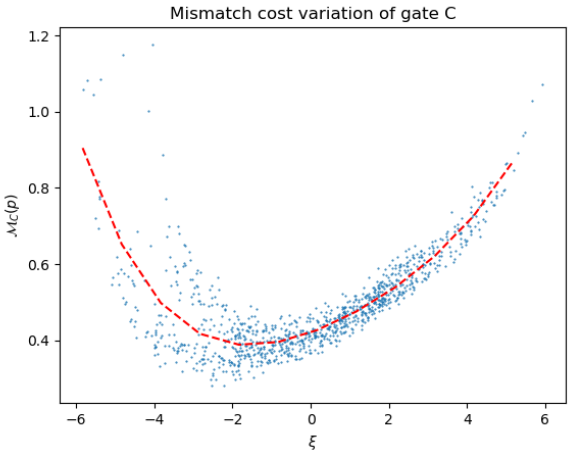
$$\mathcal{M}_A(p) = D_{KL}(X_A||X_{A,\mathrm{opt}}) - D_{KL}(Y_A||), \tag{3.43}$$
$$\mathcal{M}_B(p) = D_{KL}(X_B||X_{B,\mathrm{opt}}) - D_{KL}(Y_B||Y_{B,\mathrm{opt}}), \tag{3.44}$$
$$\mathcal{M}_C(p) = D_{KL}(Y||Y_{\mathrm{inp,opt}}) - D_{KL}(Z||Z_{\mathrm{opt}}). \tag{3.45}$$

Based on both 3-inputs and 4-inputs cases, the behavior of the mismatch cost for each gate according to the correlations and variance of each variable can be evaluated.

Regarding the correlations of the input variables, the variation of the mismatch cost of gate C – the gate in the second layer – for the 4-inputs case corresponding to Fig. 3.9 is displayed in Fig. 3.11. These graphs corresponds to the analytical implementation of the mismatch cost, if the normal law $\mathcal{N}(0,\mathbb{1})$ was the optimal law, with $\mathbf{a} = (-0.05, 2)$, $\mathbf{b} = (-0.15, 1.5)$ and $\mathbf{c} = (0.02, -0.05)$ and $\sigma_{11} = 5, \sigma_{22} = 3, \sigma_{33} = 4$ and $\sigma_{44} = 1$. The graph on the left corresponds to the variation according to the correlation of the inputs of gate A and B, and the graph on the right corresponds to the variation according to ξ as introduced in Eq. 3.33, which gathers the correlations of all inputs that have no impact in the first layer (composed of gate A and B), in the sense that some of the inputs do not interact with each other in that layer.



a) According to $\text{corr}(X_1, X_2)$ and $\text{corr}(X_3, X_4)$



b) According to ξ

Figure 3.11: $\mathcal{M}_C(p)$ of the 4-inputs case

While the behavior of the mismatch cost for gates A and B leaves no surprise, one can observe that the correlation between the inputs of gate A or gate B doesn't play a big role in determining the mismatch cost of the subsequent gate. On the other hand, ξ helps describing the evolution of $\mathcal{M}_C(p)$. This was expected since ξ plays a similar role as the covariance of the inputs when dealing with a 2-input gate, since ξ is the off-diagonal element in the covariance matrix of gate C's inputs. Although this parameters depends on almost all the covariances between the input variables, if the variables are independent it is clear that $\xi = 0$ and will thus yield a minimal cost. Conversely, given that for example all weights are positive, highly positively correlated inputs would yield a great ξ and thus greater mismatch cost. One could point out that the spread of the mismatch cost curve is surprising as it results from the analytical expression of $\mathcal{M}_C(p)$, but the number of parameters (σ_{12} in $\mathbf{a}\Sigma_A\mathbf{a}^T$, σ_{34} in $\mathbf{b}\Sigma_B\mathbf{b}^T$ and $\sigma_{13}, \sigma_{14}, \sigma_{23}, \sigma_{24}$ in ξ) involved in its computation implies different values of $\mathcal{M}_C(p)$ given the same ξ; neither ξ and $M_C(p)$ are injective functions in terms of the covariances of the inputs as several sets of covariances can give the same ξ value.

When considering the variance for the 4-inputs ANN case, it turns out – without surprise – that it doesn't play a major role in the estimation of the mismatch cost of gate C, as can be seen in Fig. 3.12.

The preceding results about the influence of variance on the mismatch cost of running gate C are most likely incorrect when considering scenarios like the 3-inputs case, where one input contributes to multiple gates. One can expect the variance of X_2 to have an impact on $\mathcal{M}_C(p)$, but also on the overall mismatch

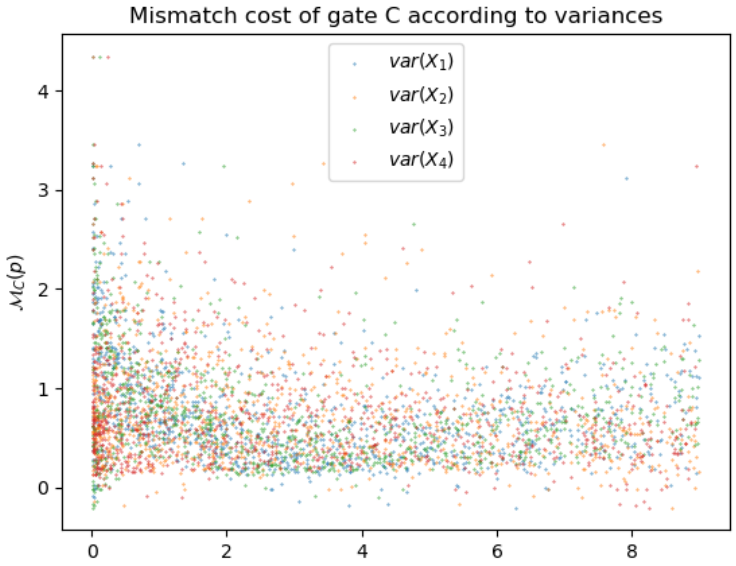


Figure 3.12: Mismatch cost of the final gate in the 4-inputs case in terms of the variance of the inputs

cost since it contributes to both gates in the first layer. Indeed, for uncorrelated inputs, fixed variances $\text{var}(X_1) = 5, \text{var}(X_3) = 5$ and given the following weights[11] : $\mathbf{a} = (1.2, 0.4)$, $\mathbf{b} = (1.7, 2.25)$, $\mathbf{c} = (0.05, 0.75)$, the plot of the mismatch cost of each gate in terms of $\text{var}(X_2) = \sigma_{22}$ is shown in Fig. 3.13 . While the mismatch cost of the gates in the first layer tend to explode for a low variance, it is not the case for the mismatch cost of gate C. This can simply be explained by the fact that the variance is involved in the ξ parameter, for which the mismatch cost explodes for high and low values; since the variance can only be negative, and a low variance is more likely to mean a lower ξ, such an explosion occurs only at higher variance. Note however that for fixed variances for the X_1 and X_2 variables, the variance taken by X_2 must respect some conditions such that the covariance matrix is positive-semidefinite[12]. In addition, the correlation between the inputs doesn't change the behavior of the mismatch cost with relation to $\text{var}(X_2)$, as it only restricts the domain on which the variance can take its values.

When the gates are weighted such that X_2 has a greater importance than the other variables, the influence of $\text{var}(X_2)$ changes, as can be seen in Fig. 3.14.

[11] As for other graphs, the weights were arbitrarily considered for a visualization purpose.

[12] Of course, this applies for both X_1 and X_3 if one fixed the variances of the other variables, but here the 3-inputs case focuses on the importance of the variance of the input variable contributing to several gates, which is X_2.

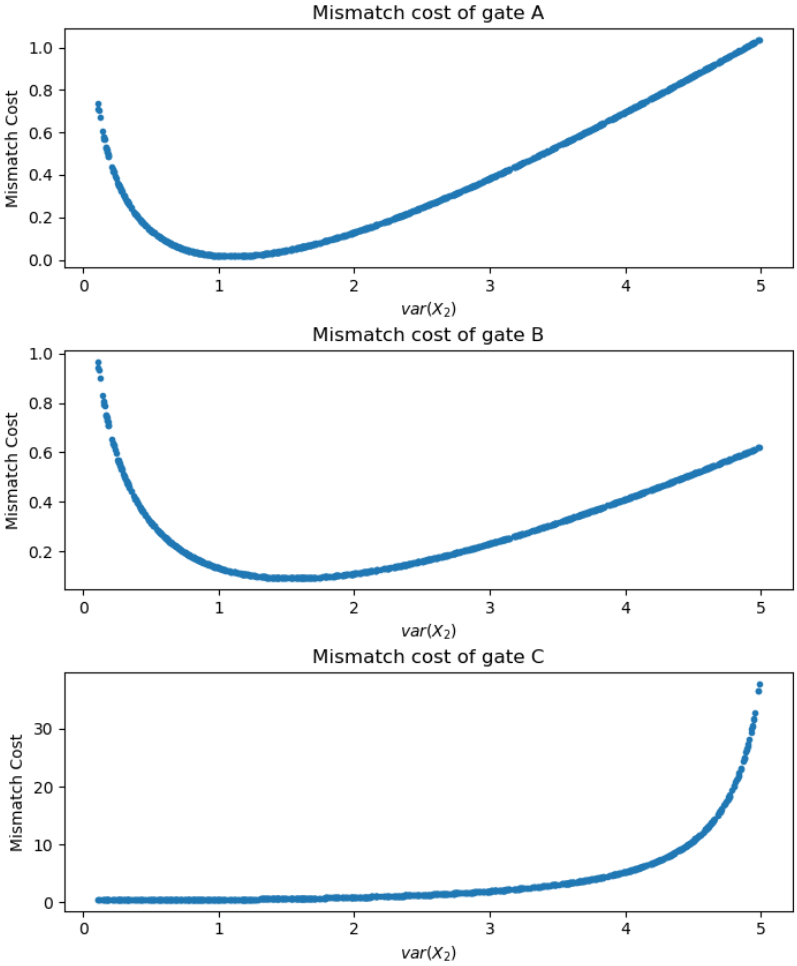


Figure 3.13: Mismatch cost for uncorrelated inputs in the 3-inputs case

The weight associated to X_2 in gate A is 50 times greater than for X_1, and 100 times greater than for X_3 in gate B. Fig. 3.14 shows however that the mismatch cost behaves similarly for both gates, while the mismatch cost of gate C behaves in a completely different manner than with regular weights. It may come as a surprise that $\mathrm{var}(X_2)$ takes values only up to 1; however having such heavy weights imposes restrictions to the values that can be taken by the covariance matrix of the new distributions (the ones generated once gate A and B are run). While in real-life the new distributions exist no matter what, a decision was taken to avoid such problematic scenarios implementation-wise[13]. It still provides a great insight nonetheless, as one can see that for a variance close to the optimal one, due to the rigged weights that enhance the influence of X_2, the mismatch cost reaches its minimal value around the optimal value.

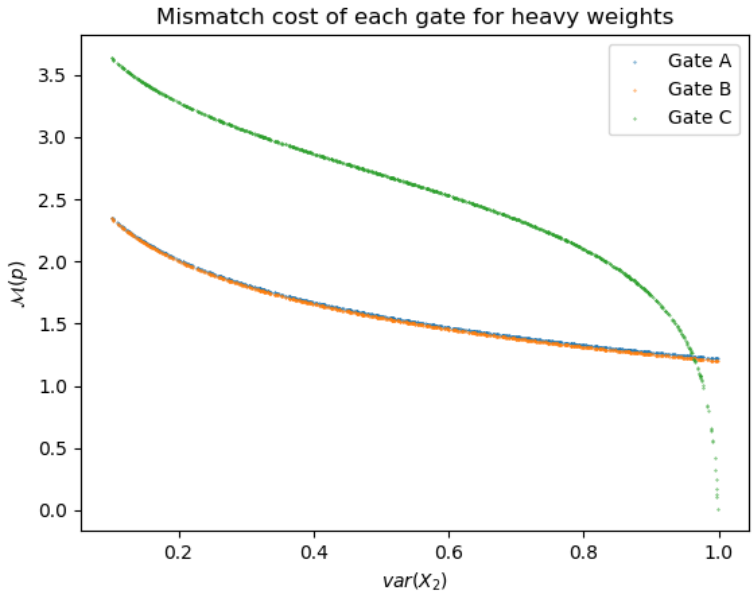


Figure 3.14: Mismatch cost for the heavily weighted 3-inputs case

The influence of the correlations in the 3-input case is shown in Fig. C.1 in the appendix, with weights $\mathbf{a} = (1.5, 0.75), \mathbf{b} = (0.2, 1)$ and $\mathbf{c} = (1, 1)$. The branches that appear are due to the number of variables involved; a two-dimensional plot is not enough to properly illustrate the dependency. One can still point out that, as expected, highly correlated variables yield a higher mismatch cost. Further comments about the impact of the correlation according to the structure of the

[13]The problem rises when Σ'_{opt} is computed, as its determinant is negative for high values of $\mathrm{var}(X_2)$ since $\det \Sigma'_{\mathrm{opt}} = |\mathbf{a}|^2|\mathbf{b}|^2 - a_1 b_0 \mathrm{var}(X_2)$.

circuit will be discussed below.

Concerning the impact of activation functions, it is absolutely non-trivial to compute the distribution of the outputs of each gate, since this output would no longer be a simple linear combination of the inputs. While it may be possible in some cases to compute the probability density of the output, the simulations remain trustful when it comes to emulate the propagation of the inputs through an ANN with specific activation functions. This will therefore be discussed in the numerical analysis of the mismatch cost.

Special wiring choice based on correlation

As mentionned in the last section of Chapter 2, wiring together highly correlated variable would minimize the Landauer loss, the unavoidable additionnal entropy flow that is incurred by the specific SR implementation[14]. One may wonder if such assumption applies for the mismatch cost and the mismatch loss, although the latter can take negative value, in opposition to the Landauer loss and the mismatch cost.

Since the 3-inputs and 4-inputs cases were introduced as smaller network within a bigger one, the output considered constitutes only one node, which could be considered as the output node for the sake of the study of the mismatch loss function; however by definition this function relies on the multi-divergence of the output distribution, information theoretic quantity that only applies to multidimensional state space. Due to this ambiguity, the mismatch loss function will be studied in other cases, limiting ourselves to the study of the mismatch cost, given the initial wiring.

Two situations are possible : either wiring highly correlated inputs together such that they go through the same gate in the first layer will yield lower mismatch cost, due to the fact that the outputs of the gates in the first layer will be weakly correlated (or at least less than the input variables) ; or the resulting mismatch cost from the gates in the first layer will be so great that the resulting "decorrelation" of the outputs of the second layer will not be sufficient to counter the huge mismatch cost created by the highly correlated variables. For the 4-inputs and 3-inputs cases, the influence of the highly correlated variables when they go through the same gate in the first layer or not will be studied.

Note however than since the "global correlation factor" for the second layer is given by ξ, weights will play a huge role. Indeed, in the 4-inputs case for example, if σ_{13} and σ_{24} are great and equal due to the high correlation between X_1 and X_3, and between X_2 and X_4, their resulting contribution to ξ could be zero if $a_1 b_1 = -a_2 b_2$ (*e.g.* $(a_1, a_2) = (1, -2)$ and $(b_1, b_2) = (2, 1)$) but could also be huge if

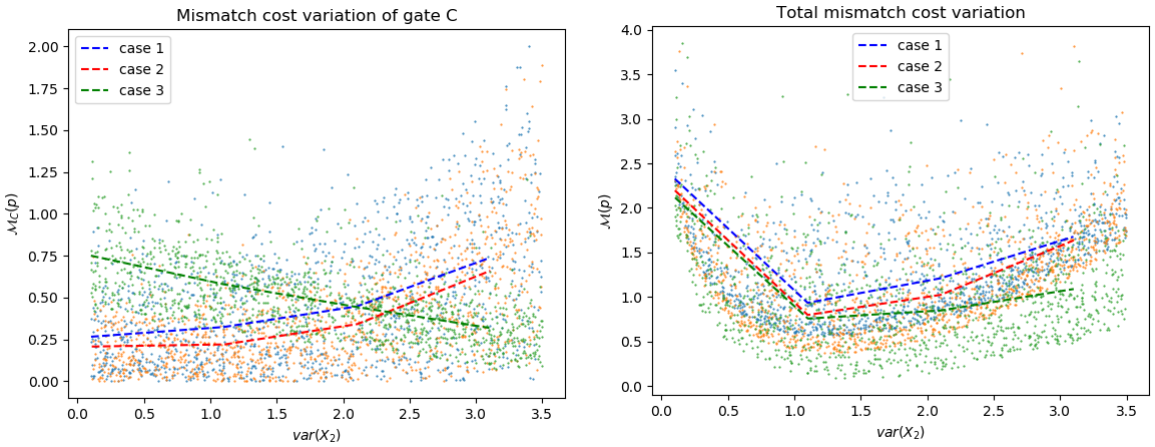
[14]Not to be confused with the Landauer cost, which is the minimal entropy flow generated by running the system.

the weights are great and positive. For the sake of simplicity, only positive weights and correlation factors will be considered. The mismatch cost of gate C and the total mismatch cost yielded will be compared.

Starting with the 3-inputs case, given the following correlations : $\rho_{12} = 0.9, \rho_{23} = 0.01, \rho_{13} = 0.01$, the configurations are denoted as :

1. X_1 and X_2 as inputs of gate A, X_2 and X_3 as inputs of gate B;
2. X_3 and X_2 as inputs of gate A, X_2 and X_1 as inputs of gate B;
3. X_2 and X_3 as inputs of gate A, X_3 and X_1 as inputs of gate B;

with the color codes being blue, red and green for cases 1, 2 and 3 respectively. The behavior of the mismatch cost for gate C as well as the total mismatch cost is indicated below according to solely the variance of X_2; the other variance dependencies are displayed in appendix C (see Fig. C.3). One can see that, as the variance increases, it is preferable to decouple the highly correlated variables in the first layer, such that only weakly correlated variables are coupled within the first layer. When looking at the total mismatch cost, it is clear that event the loss in mismatch cost in gate C for low variances does not counter the increase due to highly correlated variables serving as inputs for one of the two gates in the first layer; the best decision is to keep correlated variables as decoupled as possible.



a) Mismatch cost of gate C

b) Total mismatch cost

Figure 3.15: Evolution of the mismatch cost for the 3-inputs case

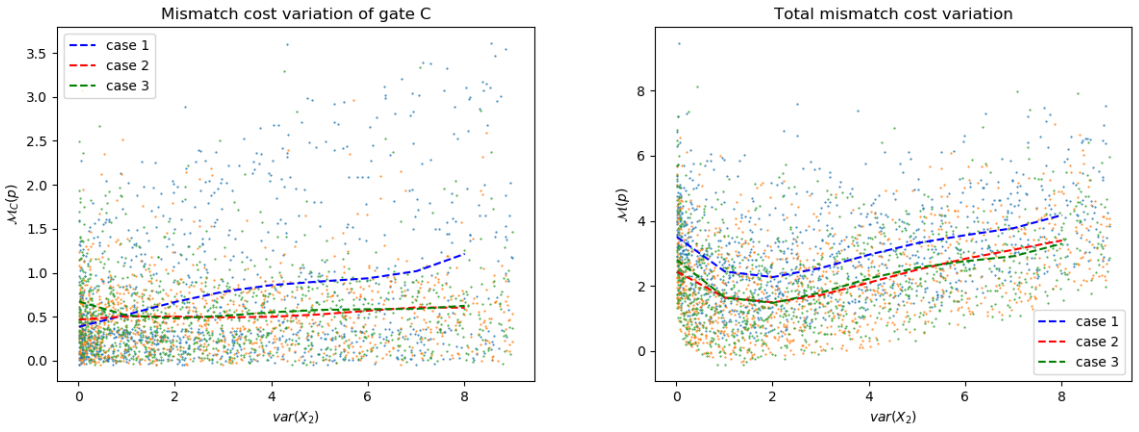
Concerning the 4-inputs case, the same strong and weak correlations are considered but for two other couple, such that $\rho_{12} = 0.9, \rho_{34} = 0.9, \rho_{23} = 0.01, \rho_{13} = 0.01, \rho_{14} = 0.01, \rho_{24} = 0.01$. Only a few combinations of wiring a considered, namely

1. X_1 and X_2 as inputs of gate A, X_3 and X_4 as inputs of gate B;
2. X_4 and X_1 as inputs of gate A, X_3 and X_2 as inputs of gate B;
3. X_1 and X_3 as inputs of gate A, X_4 and X_2 as inputs of gate B;

with the same color code as before. Since the choice of the variable as x-axis doesn't have as much influence as in the 3-inputs case, the mismatch costs are only plotted according to var(X_2) and are displayed in Fig.3.16. As expected from the 3-input case, coupling the most correlated variables so that they all contribute to the same gates in the first layer results in higher mismatch cost, would it be in the first layer or second as can be seen in the graph on the left side. Again, it is then better to keep highly correlated variables decorrelated.

A choice therefore must be made between minimizing the Landauer loss, or minimizing the mismatch cost. One way to solve this problem is to use a initial probability distribution extremely close to the optimal one, as then it would correspond to the best wiring that was chosen by solving the minimization problem associated to the Landauer loss.

The reader might point that such choice of wiring doesn't really apply to neural networks such as the multilayer perceptron, since every neuron from a layer contributes to each neuron of the next layer. It is however still interesting to have in mind the impact of correlated variables on the energetic optimization of such networks, as it might give insights on how to elaborate different architectures to avoid such issues.



a) Mismatch cost of gate C

b) Total mismatch cost

Figure 3.16: Evolution of the mismatch cost for the 4-inputs case

Non-actualized optimal distribution

The main difference between the actualized scenario and this one is the actualization of the optimal distributions of the inputs for the gates in the next layers. The optimal distribution of the gate's output still depends on the optimal distribution of its input, as its propagation through the gate when it is run is considered; but as can be seen in the expression of the KL divergence of the inputs of gate C, the optimal covariance matrix is nothing but the identity matrix, as the optimal distribution of gate A and B's outputs do not influence the optimal distribution of gate C's inputs. In particular, the KL divergences of each gate and its parents is depicted below.

$$D_{KL}(X_A||X_{A,opt}) = \frac{1}{2}\left(\mathrm{tr}\Sigma_A + \mu_A^T\mu_A - 2 - \ln\det\Sigma_A\right) \tag{3.46}$$

$$D_{KL}(X_B||X_{B,opt}) = \frac{1}{2}\left(\mathrm{tr}\Sigma_B + \mu_B^T\mu_B - 2 - \ln\det\Sigma_B\right) \tag{3.47}$$

$$D_{KL}(Y_A||Y_{A,opt}) = \frac{1}{2}\left(\frac{\mathbf{a}\Sigma_A\mathbf{a}^T}{|\mathbf{a}|^2} + (\mathbf{a}\cdot\mu_A)^2 - 1 - \ln\frac{\mathbf{a}\Sigma_A\mathbf{a}^T}{|\mathbf{a}|^2}\right) \tag{3.48}$$

$$D_{KL}(Y_B||Y_{B,opt}) = \frac{1}{2}\left(\frac{\mathbf{b}\Sigma_B\mathbf{b}^T}{|\mathbf{b}|^2} + (\mathbf{b}\cdot\mu_B)^2 - 1 - \ln\frac{\mathbf{b}\Sigma_B\mathbf{b}^T}{|\mathbf{b}|}\right) \tag{3.49}$$

$$D_{KL}(Y||Y_{opt}) = \frac{1}{2}\left(\mathrm{tr}(\Sigma') + \mu'\mu'^T - 2 - \ln\det\Sigma'\right) \tag{3.50}$$

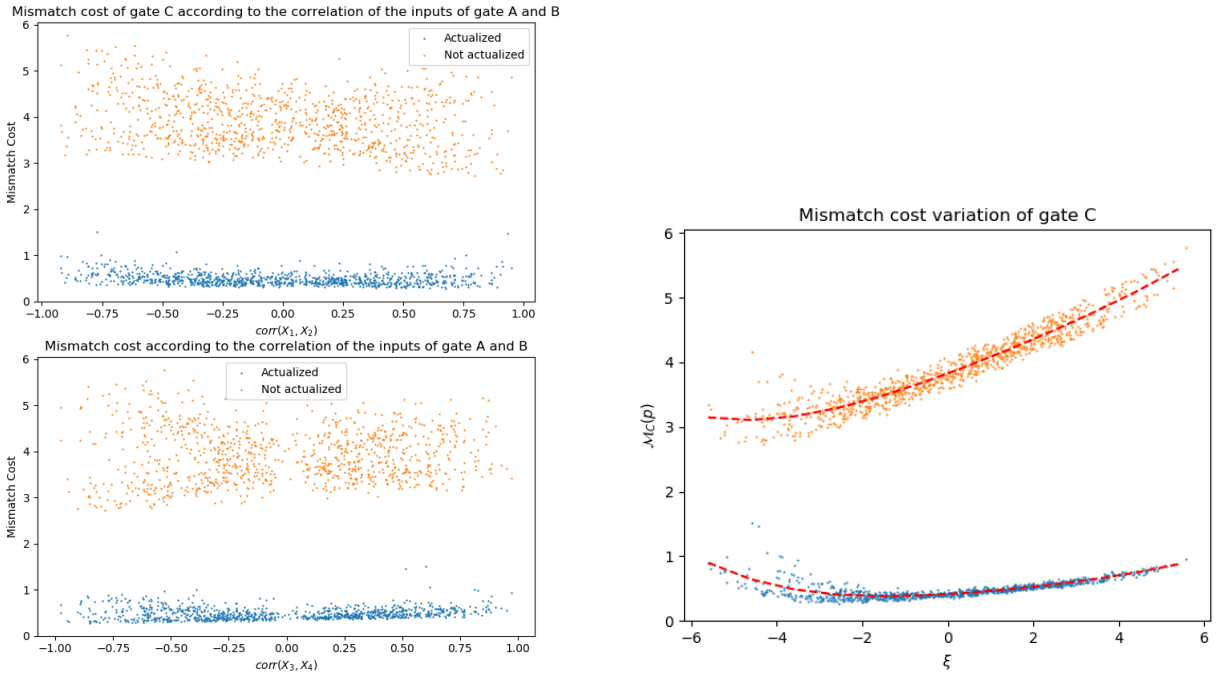
$$D_{KL}(Z||Z_{opt}) = \frac{1}{2}\left(\frac{\mathbf{c}\Sigma'\mathbf{c}^T}{|\mathbf{c}|^2} + (\mathbf{c}\cdot\mu')^2 - 1 - \ln\frac{\mathbf{c}\Sigma'\mathbf{c}^T}{|\mathbf{c}|^2}\right) \tag{3.51}$$

This assumption of considering the same optimal distribution for each gate's inputs (the parent nodes) comes from a more practical point of view, where each neuron is in itself a piece of hardware that has been built in bulk, such that each neuron wasn't constructed specifically for the actualized optimal distribution; while [11] considers the updated distribution after propagating the initial distribution through a gate which implements some conditional distribution π_g, from the serial-reinitialized circuit standpoint one can perceive each gate as a solitary process with its optimal distribution being only considered within that solitary process, putting aside any alteration to the optimal distribution that could come from the other processes to which it is not coupled.

The behavior of the mismatch cost is directly compared for both 4-inputs and 3-inputs cases to the mismatch cost in the actualized scenario. The weights and

other parameters used for each case will be the same as when dealing only with the actualized scenario.

In the 4-input case, while no dependency on $\text{corr}(X_1, X_2)$ and $\text{corr}(X_3, X_4)$ is shown for both the actualized and not actualized situations, it is clear that the mismatch cost for the same weights as in Fig. 3.11 is greater than in the actualized scenario. This is even clearer when putting into comparison the mismatch cost according to ξ for both scenarios. While the mismatch cost behaves somewhat similarly in both cases, it is however more sensitive to the variation in ξ when gates are not actualized; this is due to the fact that ξ depends on both the correlations and weights, and the optimal covariance matrix for the inputs of gate C Σ'_{opt} appearing in the actualized scenario, which depends on the weights, does not appear in the non-actualized framework; this factor helped normalize the influence of the weights, without it the mismatch cost depends more on the weights and thus the impact of higher correlations between the input variables is less dampened as weights may enhance the value taken by ξ without something countering such increase.



a) According to $\text{corr}(X_1, X_2)$ and $\text{corr}(X_3, X_4)$

b) According to ξ

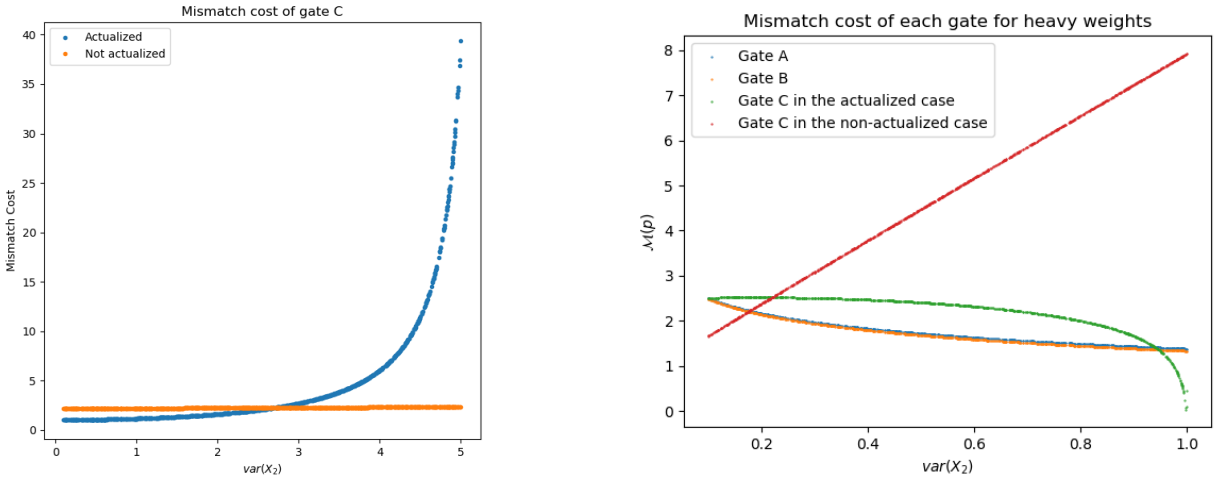
Figure 3.17: $\mathcal{M}_C(p)$ of the 4-inputs case

Let's now see what happens in the interesting 3-inputs case, as ξ depends on the variance of the second input, even when the optimal distribution is considered,

implying non-zero off-diagonal element in the optimal input covariance matrix Σ'_{opt}. As in the actualized case, the impact of the correlation is limited, as can be seen in Fig.C.2. The focus of the comparison is therefore on the impact of $\text{var}(X_2)$.

Quite interestingly, for normal weights, the mismatch cost stays constant whatever how great the value of $\text{var}(X_2)$ becomes; since the variance is only involved in Σ' through $\mathbf{a}\Sigma_A\mathbf{a}^T$ and $\mathbf{b}\Sigma_B\mathbf{b}^T$, which makes the evolution of the mismatch cost under changes in the variance heavily dependent of the weights of the network. It is then with no surprise that, for a network involving heavy weights the input variable X_2 contributing to both gates in the first layer, the mismatch cost in the non-actualized case explodes, in opposition to the actualized scenario.

As the vast majority of neural networks involve the contribution of one specific neuron output in the $i-1^{th}$ layer to several neurons in the i^{th} layer, the non-actualized scenario would yield higher mismatch cost than the actualized one, due not only to its circuit implementation but also by its piece-wise structure, in the sense that each gate behaves the same although the outputs do not reset their distribution. This additional cost reminds of the mismatch loss introduced in chapter 2, which evaluates how much energy was lost due to the serial-reinitialized implementation of the circuit, instead of just considering the initial and ending distributions. Since the non-actualized case adds another cost due to the nature of its implementation, it is better to stick to the actualized scenario as it allows for a better understanding theoretically, since no additional cost than the ones already mentionned within [11] is considered.



a) With regular weights

b) With heavy weights associated to X_2

Figure 3.18: $\mathcal{M}_C(p)$ of the 3-inputs case

3.3.2 Numerical simulations

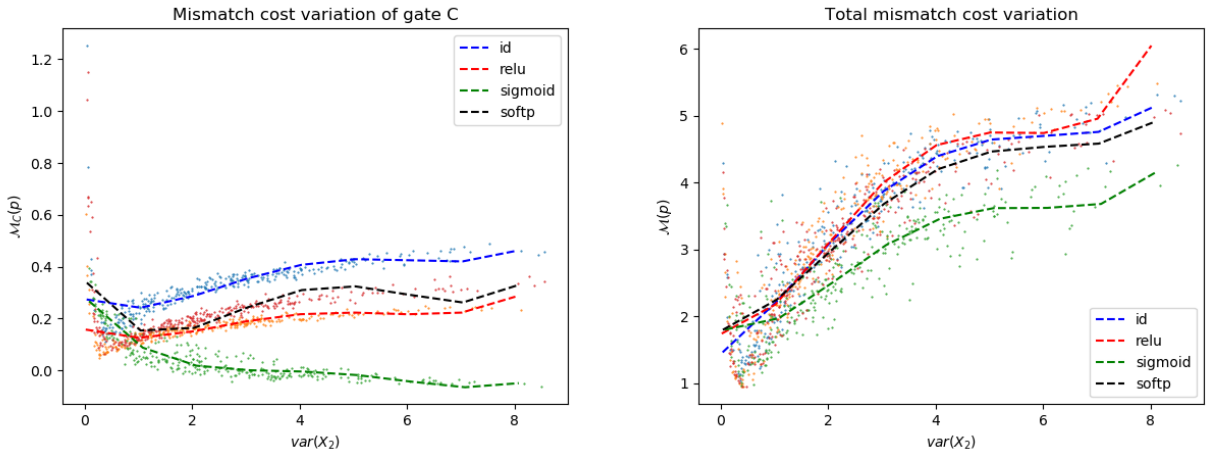
As the analytical case where the normal distribution $\mathcal{N}(0, \mathbb{1})$ is considered optimal reached its limits of study, it is time to get a better grasp on how the mismatch cost behaves for the other optimal distribution that was mentionned (which is probably the best to consider due to the principle of maximum entropy), which is the uniform distribution. As the algebra of random variables starts to become crazily non-trivial as soon as the laws are not gaussian, numerical simulations come in handy to estimate the quantity we are interested in.

The 3-input case being less computationally heavy, but also more coherent when studying artificial neural networks than the 4-inputs case, the latter will not be considered. Instead, an additional scenario will be considered, one where the newtork is composed of several layers of two neurons, with each neuron contributing to both neurons of the next layer.

Note on the implementations

To ensure computational feasibility and valid computation of the KL divergence between the several distributions, the set of outcomes is bounded, such that the values taken by the output of the first layer are bounded to this space, from which are taken the inputs. This also allows for comparison when computing the KL divergence between the normal distribution and the uniform distribution, using the discretized formalism; while it is true that the KL divergence between a continuous distribution and a discrete distribution cannot be computed, using the discretized formalism on a bound space allows for comparison over "chunks" of this space, such that for each distribution there is a non-zero probability of being in each chunk. The KL divergence can then easily be computed, as each state is associated to a non-zero probability for both distributions considered.

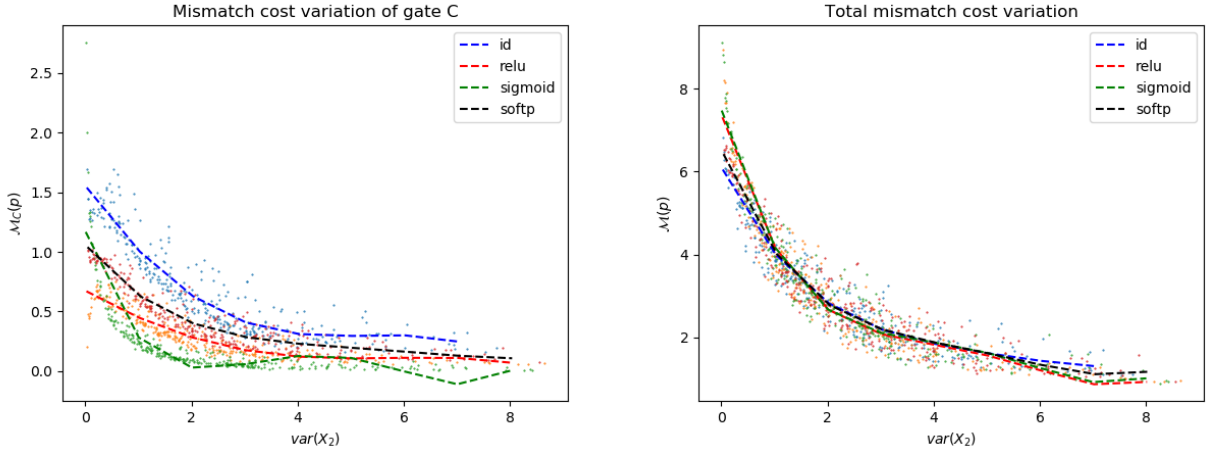
The distributions are registered thanks to Monte Carlo simulations as dictionaries, with keys corresponding to the chunk or interval that is associated to the value registered, and the values being the number of times a value fell into that chunk or interval. It is then easy to transform such values to probabilities. In order to properly cover the whole state space, the number of simulations is at least 100 times greater than the number of states. Obviously, as the number of dimensions increases (*e.g.* for n variables taking their values within N possible states, the size of their joint state space is therefore N^n), it becomes more and more challenging to properly simulate the joint distribution. This is why only simple cases are considered, as the computational resources are limited.



a) Mismatch cost of gate C

b) Total mismatch cost

Figure 3.19: Study of the mismatch cost in the 3-inputs for inputs limited in the interval $[-1, 1]$



a) Mismatch cost of gate C

b) Total mismatch cost

Figure 3.20: Study of the mismatch cost in the 3-inputs for inputs limited in the interval $[-5, 5]$

3-inputs case

Different results occur when changing the bounds of the values that can be taken by the input variables. As a reminder, these bounds are set such that the optimal distribution shares the same support as the input distribution, such that the KL divergence between both distributions makes sense.

This can be explained due to the distribution of the inputs once they are propagated through each neuron, as it was displayed in Fig. 3.7. As for greater intervals the distribution of the outputs covers a wider spread, the mismatch cost tend to decrease as variance increases since the distribution is more diffused, therefore being similar to the optimal distribution. With restricted bounds, probability distribution is extreme as the probability density is concentrated on a few states. As variance increases, values taken by the output would go beyond the bounds imposed, therefore saturating the states at the boundaries, amplifying this bimodal distribution of the outputs. Nevertheless, the sigmoid activation function still remains a reliable activation function when it comes to energy usage; but the difference between activation functions when dealing with variables that take values over greater intervals is negligible. The nature of the problem therefore directs the choice of the activation function, based on the results but also on the values taken by the inputs.

Cumulative double gates

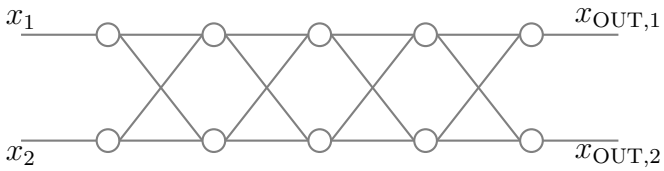
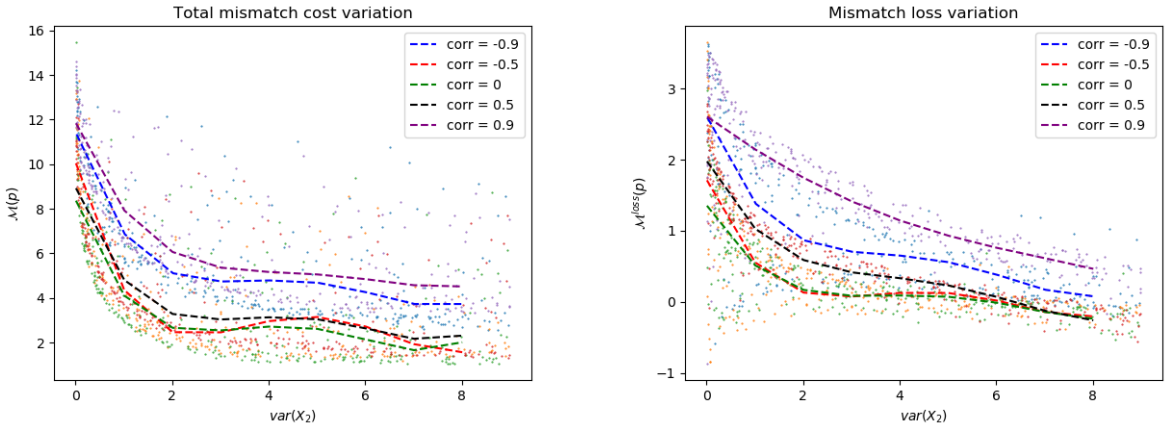


Figure 3.21: 2-inputs ANN with consecutive double gates

The network considered for this experiment is precisely the one depicted in Fig. 3.21. The goal of this experiment is to determine how the mismatch cost evolves in each gate, and how correlation between the input variables affect the total mismatch cost of a network with more than two layers. Since the distribution of the output states x_{OUT} covers a multidimensional space, this is an opportunity to compute the mismatch loss of such system, to see what is lost due to the SR implementation of the system.

The results are shown in Fig. 3.22. Several correlation settings have been considered, while only one variance dependency is shown since the dependency is symmetric (apart from the influence of the weights throughout the network, which

are set randomly). One can see that the observation made at the very beginning concerning a single gate still holds, as the mismatch cost is at its highest when the input variables are highly correlated, while it reaches its lowest when they are independent. This results also holds for the mismatch loss, as the energetic impact of running a SR implementation increases as the variables that go through each gate are correlated. In Fig. C.4, the impact of the correlation on each layer is shown[15]. It is clear that whatever the correlation between each input variables, the mismatch cost in each layer tends to zero; this can be expected as the propagated distribution of both optimal and initial distribution tend to the same distribution, as can be seen in Fig. C.5.



a) Total mismatch cost

b) Mismatch loss of the system

Figure 3.22: Mismatch cost and mismatch loss of the cumulative double gates

3.4 Trained neural network

As a way to evaluate precisely how energetic loss applies and can be avoided with real machine learning scenarios, a neural network will be trained on a simple dataset for a classification problem to set the appropriate weights on the structure described in Fig. 3.23. This mismatch cost will be put in comparison to the actual Landauer cost of the system.

[15] Actually, it's the mismatch cost of one of the two gates within each layer, as the network is symmetric – at least in its structure, without considering the input variances and the network's weights.

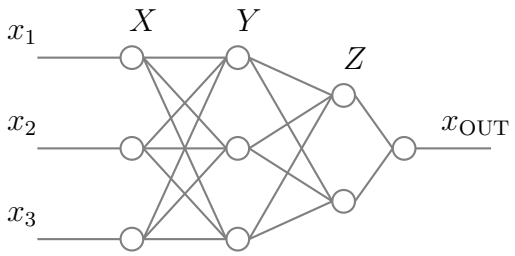


Figure 3.23: ANN of the trained neural network for the classification problem

3.4.1 Problem and dataset description

The dataset considered to train our model is the "*Connectionist Bench (Sonar, Mines vs. Rocks) Data Set*" [40], which gathers frequency patterns from sonars that are then labelled as either corresponding to a rock or a mine. There are 111 patterns corresponding to signals bouncing off mines (actually metal cylinders, such that the dataset creation is less letal), and 97 bouncing off rocks. Each pattern is explained through 60 attributes, each one being a real-valued number that corresponds to the average energy within a frequency band, and each pattern is labelled either "R" for rock or "M" for mine. The task is then to build a classification model that can easily detect if the signal received bounced off a rock or a mine.

3.4.2 Feature selection

As there are numerous features in comparison to the number of samples, and in order to optimize our network (as the input size is limited for computational reasons, as the computation of the Kullback-Leibler divergence requires to simulate the distributions at each node), it is best to do some feature selection first, thus picking the most relevant attributes in the dataset.

One way to do as such is through Principal Component Analysis (PCA) [41], which will perform a change of basis within the data such that the principal components, those explaining the most variance in the dataset (thus explaining the most variations, the most changes, the most relevant factors to guess which label a sample belongs to), are received in decreasing order with the relation to the amount of variance each component can explain. In other words, the first column of the transformed dataset, which corresponds to the first principal component, describes the largest amount of variance in the dataset. The best features therefore correspond to the first columns of this transformed table.

Another important feature of PCA is that the new attributes are uncorrelated, as they serve as an orthonormal basis to the prinicpal component space. This will allow for comparison between a model trained with untreated data, and a model

that went through feature selection. It is expected that the mismatch cost of the untreated model will be greater than the other, since its attributes will most likely be more correlated.

3.4.3 Results

The upcoming results are based on the network structure as described in Fig. 3.23, using the Keras package from Tensorflow [42], which allows to construct any neural network and retrieve its associated weights once it is fully trained.

Trained with the transformed dataset

The input distribution is assumed to follow a normal law such that $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$ where

$$\mu = \begin{pmatrix} -2.27 \times 10^{-17} \\ -4.553 \times 10^{-17} \\ -1.253 \times 10^{-16} \end{pmatrix} \text{ and } \Sigma = \begin{pmatrix} 12.53 & -2.08 \times 10^{-15} & -9.36 \times 10^{-16} \\ -2.08 \times 10^{-15} & 11.82 & 1.44 \times 10^{-15} \\ -9.36 \times 10^{-16} & 1.44 \times 10^{-15} & 5.25 \end{pmatrix}.$$

The weights for each gate can be found in the appendix in Table C.1, where the numeration goes from top to bottom as indicated in Fig.3.23. With such settings, the model achieved a 81% accuracy on its training dataset, and 75% on the validation set, on which it was not trained. While these scores are quite low compared to a properly built and trained model, it is sufficient in this case as the goal is only to use its weights to estimate the mismatch cost. The loss function used was the cross-entropy[16], which is usually the best for binary classification problems. The weights are actualized through gradient-descent, and the activation function used in the first and second layer is ReLu, while the last layer (determining the output) uses the sigmoid. These activation functions were the one yielding the best results in terms of accuracy.

As can be seen by the covariance matrix of the inputs, the inputs are indeed pretty much uncorrelated. Considering $\mathbf{X}$ as the input nodes, the nodes belonging to G using the notations of Chapter 2 are $Y_1,\ Y_2,\ Y_3,\ Z_1$ and Z_2. The parent gates of Y_i are $X_1,\ X_2,\ X_3\ \forall i \in \{1, 2, 3\}$, and the parent gates of Z_1 and Z_2 are $Y_1,\ Y_2,\ Y_3$. The simulations therefore brings the following results, given p the initial distribution as explained above :

[16]The cross entropy $S(p, q)$ is somewhat similar to the KL divergence $D_{KL}(p||q)$, as it measures how two distributions differ; they are actually related through the following equation :

$$S(p, q) = S(p) + D_{KL}(p||q).$$

- $\mathcal{L}(p) = 8.9895$ nats;
- $\mathcal{M}(p) = 1.0864$ nats.

The emmitted heat due to these two factors is then 4.075×10^{-20}J, assuming the heat bath is at 20°C. The only missing quantities to properly evaluate the entropy flow generated by the network are the residual entropy production and the Landauer loss; however, as gates with more than one child are considered, the Landauer loss is non-zero[17], and the Landauer and mismatch costs can serve as a lower bound to the total entropy flow :

$$\mathcal{Q}(p) >= \mathcal{L}(p) + \mathcal{M}(p). \tag{3.52}$$

In particular, the mismatch cost consists as a lower bound to the entropy production associated to the neural network. One can see that the mismatch cost here is almost 9 times lower than the Landauer cost, thus contributing barely to the total entropy flow. This will not be the case when considering correlated data as in the following section.

Trained with the original dataset

It is assumed that th model was trained using untreated data as well, since it wouldn't make sense to give an untreated dataset to a model trained with the transformed training dataset as the attributes do not share the same space (the first principal component is not comparable to the best column in the original dataset as it defines another basis).

As the number of features selected is limited by the structure of the network, the most relevant features were considered based on a χ^2 statistical test called univariate feature selection. By doing so, the resulting model still remains somewhat relevant, with an accuracy of 78% on its training dataset, and 71% on its validation set, but without modifying the values of the dataset.

The inputs are assumed to follow a normal law such that $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$ where $\mu = \begin{pmatrix} 0.236 \\ 0.197 \\ 0.385 \end{pmatrix}$ and $\Sigma = \begin{pmatrix} 0.017 & 0.004 & -0.005 \\ 0.004 & 0.023 & 0.01 \\ -0.005 & 0.01 & 0.07 \end{pmatrix}$. The weights of the model as well as the biases are also diplayed in the Appendix, in Table C.2. In this setting, the inputs are more correlated, thus the mismatch cost is expected to be higher than in the previous section. Computing the Landauer cost and the mismatch cost as before, the results are the following :

- $\mathcal{L}(p) = 1.823$ nats;

[17] It can even be arbitrarily large, as shown in Appendix C of [11]

- $\mathcal{M}(p) = 20.263$ nats;

thus confirming the assumption that was made. Considering the same heat bath, the emitted heat contributed by the two quantities would then be 8.935×10^{-20}J.

Interestingly, the Landauer cost is lower; this is because the entropy of the inputs is maximum when variables are uncorrelated, while the entropy of the output isn't altered as much based on the correlation of the inputs. However this gap in inevitable cost is filled by the mismatch cost in the untreated dataset case, which thus yields a greater lower bound to the entropy flow (and entropy production) of the system.

3.5 Discussion

In this chapter, an extensive search has been made on the mismatch cost, which corresponds to the generated energetic cost of running a circuit with a non-optimal input distribution. In the case of artificial neural networks, the correlations of the inputs plays a major role in the resulting mismatch cost as highly correlated variables generates a higher energetic expense. Data preparation can have an impact on such expense since some feature selection algorithms provide uncorrelated transformed data, as demonstrated with PCA. Another factor to take into account is the variance of the inputs, as each input contributes to several neurons, which will contribute to several neurons as well; however the values taken by the variances of the inputs are restricted when studying the mismatch cost analytically as the weights define the distribution of the states of the following nodes.

In fact, the analytical analysis of the mismatch cost is quite limited, as it heavily relies on having a "gentle" activation function, but also on having a prior distribution that allows for comparison over the same support. With activation functions other than the identity mapping, establishing the updated distributions becomes a non-trivial challenge that may not be worth the trouble.

Numerical simulations thus allowed some freedom when testing different network settings, as was shown with the varieties of experiments that were performed since the analytical results. It has been shown that in the non-actualized scenario, where the prior distribution for each gate is the same, the mismatch cost will reach higher values in the 4-input case, considering that each gate only has one child. While it was assumed that additional energetic cost was incurred due to this gate-specific optimal distribution, one could argue that this scenario deserves more study. In fact, this is true as one scenario perfectly fitting to the formalism introduced by David Wolpert in [11] was not covered : studying the mismatch loss of the network considering $\pi_\Phi p_{IN}$ being the propagated probability of the inputs while the

summation as described in Eq. 2.39 involves the optimal distribution considered in the non-actualized case. Since the main focus of this thesis was the mismatch cost, as it is a non-negative quantity whose minimum value can be more easily identified, the study of the mismatch loss was set aside.

The covered experiments are however limited and do not properly apply to real-case artificial neural networks, as the number of inputs, neurons and layers was limited due to the difficulty of implementation and lack of computational resources. This therefore limits the results of the research as an introductory approach to the study of the thermodynamic cost of not running the optimal distribution. Another limitation for this study was the space state, as the space in which the values were taken was discretized. The best discretization possible would be to use a state space such that 2^{32} possible states; however this does not remove the issue of switching from a continuous distribution to a discrete distribution when the input distribution is assumed to follow a gaussian distribution. It would be convenient to perform further analysis on how such discretization may impact the results that were obtained, as it may impact in a non-negligible manner the values taken by the Shannon entropy and the Kullback-Leibler divergence. Talking about the KL divergence, restricting two distributions to the same support in order to make a valid comparison using this quantity may also have an impact, as the actual (and not theoretical) values taken by the input distribution could be outside the bounds defined to match the optimal distribution, when considering an optimal distribution.

Nevertheless, it is obvious that neural networks are complex systems that involve a lot of parameters. Evaluating how distributions evolve accross a network is already a challenging task when a few layers and simple activation functions are involved, it is without doubt that more complex distributions and other type of neural networks than the multilayer perceptron will require even more work to be properly analyzed. Its link with thermodynamic quantities such as entropy flow and entropy production still must be properly formulated, as stochastic thermodynamics is only starting to tackle this topic [10][11].

Chapter 4

Conclusion

Entropy has been at the center of this work, despite not having computed it explicitely many times. The relationship between information and thermodynamics was at the core of this thesis, as it allowed to use the formalism developped by David Wolpert and Artemy Kolchinsky in [11] based on the important Landauer principle.

Thanks to stochastic thermodynamics, a clear link between probability distributions and heat was described through the decomposition of the entropy flow. This decomposition introduced the notion of mismatch cost, in addition to the already known Landauer cost. While the Landauer cost is the inevitable cost of running the circuit in the optimal framework where the wiring has no impact, based on the input and output distributions, the mismatch cost takes the implementation of the system into account as it sums the difference in Kullback-Leibler divergence, or relative entropy, between the input and the optimal distribution. Such optimal distribution is hard to determine, this is why considering the distribution yielding the greatest entropy is the best choice as it implies the least assumptions. Based on this choice, a lot of experiments about the mismatch cost have been performed.

The main result that came from this analysis is that highly correlated input variables imply a great mismatch cost, while uncorrelated variables minimize it; at least in most cases, as sometimes, due to the configuration of the system and its associated parameters – the weights, this minimum is shifted. Nevertheless, high correlated variables generate the greatest mismatch cost, but their impact tends to lower as more and more layers are considered. In fact, the updated distribution of the nodes in further layers doesn't change much whether the input variables are correlated or not; the major impact of this correlation occurs in the first layer. It has been proven that if it is possible to avoid having highly correlated variables involved within the same gate in the first layer, then the mismatch cost would be lower.

The impact of uncorrelated variables was demonstrated when applied to a

trained model, as prepared data through principal component analysis resulted in a lower mismatch cost, and a lower overall entropy flow. It may come as a surprise that data preparation is not only useful to optimize a model and avoid overfitting, but also to minimize the energetic expense of the network, although the energetic expense of data preparation might compensate this energy saving.

The description of the mismatch cost comes as a challenge when considering non-trivial activation functions and special distributions, as the logical and binary formalism used in [11] doesn't apply for real valued inputs. In such cases, it is a good idea to rely on numerical simulations, as they offer a good insight on how the function behaves, and if the assumptions made are relevant.

It is however hard to properly implement real-case networks, as the joint state of their neurons is too big to emulate with relatively low computational resources. The experiments run therefore correspond to first steps into this study, as a lot more is left to properly understand. While this study was focused on artificial neural networks, it might be insightful to have a look at other types of computation (which are not logical circuits, as it was already covered), and see how the quantities involved in this thesis and in [11] evolve according to the input variables.

As this work was about the computation occuring on an already trained model, the next step would be to evaluate the energetic expense of learning, which is expected to be more costly as more computations are involved throughout the network. Training algorithm such as backpropagation defy the directed acyclic graph description of a circuit stated in [11], as the computations occur in both directions during the learning. A better formalism must therefore be established, more fitting with such problem. Nevertheless, it looks like complete and proper thermodynamic formalism for neural networks, machine learning and artificial intelligence is yet to be constructed, and would come as a handy tool in the future, as they partake an increasing place in our society.

Appendix A

Discretization of the entropy of a continuous distribution

The proof comes from [17]. Consider X a random variable having a probability density function p. Let's suppose that the domain of the function is divided in small intervals. For an interval of length Δ, by the mean value theorem there exists a value x_i within each interval such that

$$p(x_i)\Delta = \int_{i\Delta}^{(i+1)\Delta} p(x)\mathrm{d}x. \tag{A.1}$$

Let's say that X^Δ is the discretized random variable defined by $X^\Delta = x_i$ if $i\Delta \leq X < (i+1)\Delta$. The probability that $X^\Delta = x_i$ is then given by

$$p_i^\Delta = \int_{i\Delta}^{(i+1)\Delta} p(x)\mathrm{d}x = p(x_i)\Delta. \tag{A.2}$$

Consequently, the entropy of this discretized random variable is given by

$$S(p(X^\Delta)) = -\sum_{i=1}^{N} p_i^\Delta \ln p_i^\Delta \tag{A.3}$$

$$= -\sum_{i=1}^{N} p(x_i)\Delta \ln p(x_i) - \sum_{i=0}^{N} p(x_i)\Delta \ln \Delta \tag{A.4}$$

$$= -\sum_{i=1}^{N} p(x_i)\Delta \ln p(x_i) - \ln \Delta \tag{A.5}$$

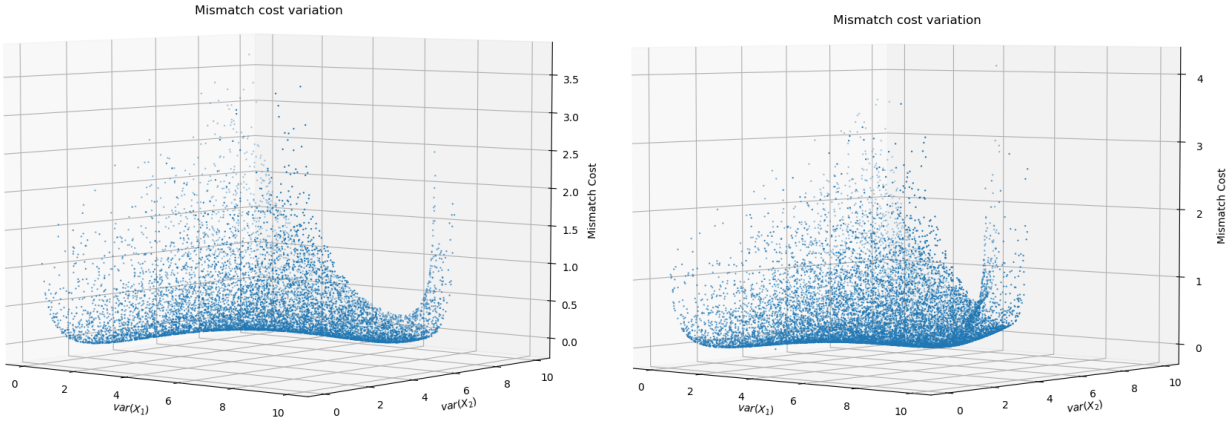
with N the number of intervals of length Δ in which the domain is divided. This holds if p is Riemann integrable.

A result is that the entropy of a n-bit discretization of a continuous random variable X is approximately $h(X) + n$, as $H(X^\Delta) + \ln \Delta \to h(p(X))$ when $\Delta \to 0$.

Appendix B

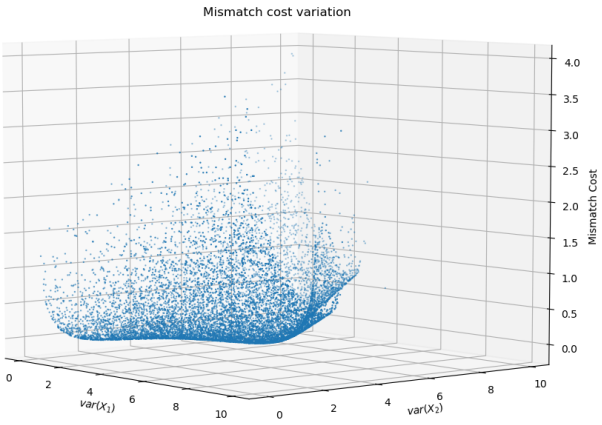
Mismatch cost of a single gate

The shape of the mismatch cost in terms of the variances of the inputs is displayed for several cases, where the correlation of the inputs varies.

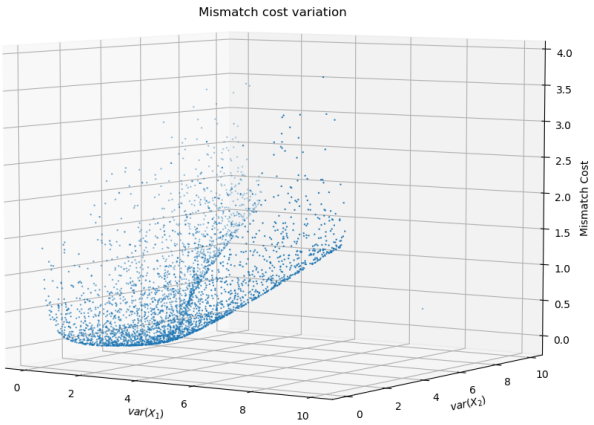


a) $\text{corr}(X_1, X_2) = 0.12$ b) $\text{corr}(X_1, X_2) = 0.15$

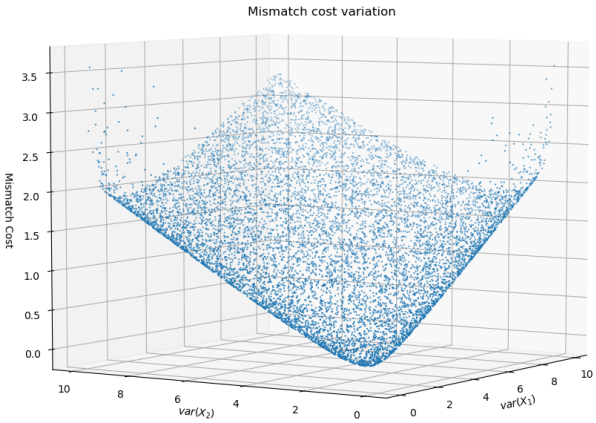
Figure B.1: $\mathcal{M}(p)$ of the 2-inputs ANN gate according to variance for weights $= [1, 1]$



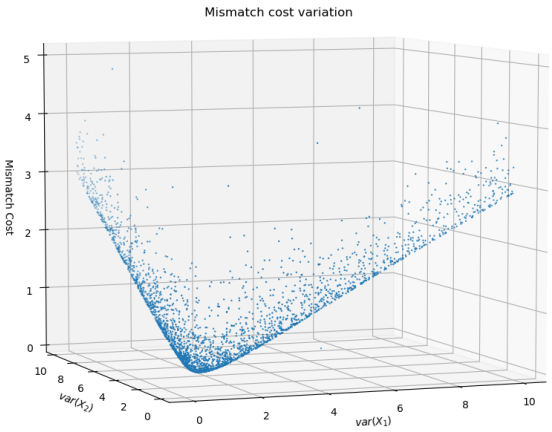
a) $\text{corr}(X_1, X_2) = 0.17$



b) $\text{corr}(X_1, X_2) = 0.3$

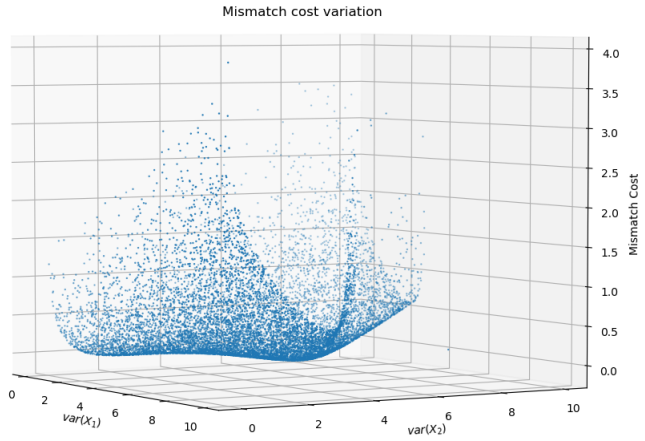


c) $\text{corr}(X_1, X_2) = 0.0$

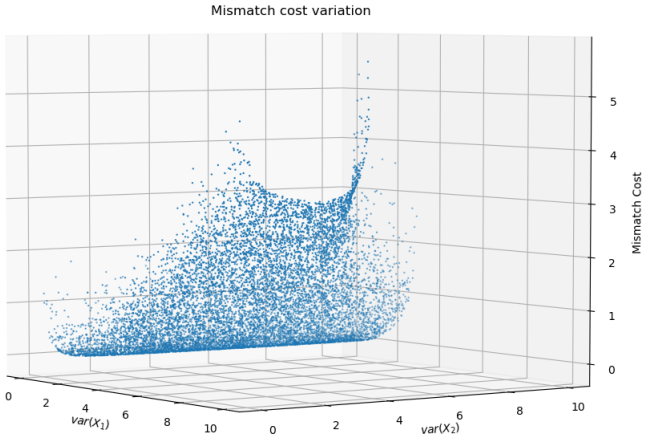


d) $\text{corr}(X_1, X_2) = 0.75$

Figure B.2: $\mathcal{M}(p)$ of the 2-inputs ANN gate according to variance for weights $= [1, 1]$



a) weights $= [1, 1]$



b) weights $= [1, 5]$

Figure B.3: $\mathcal{M}(p)$ of the 2-inputs ANN gate for different weights

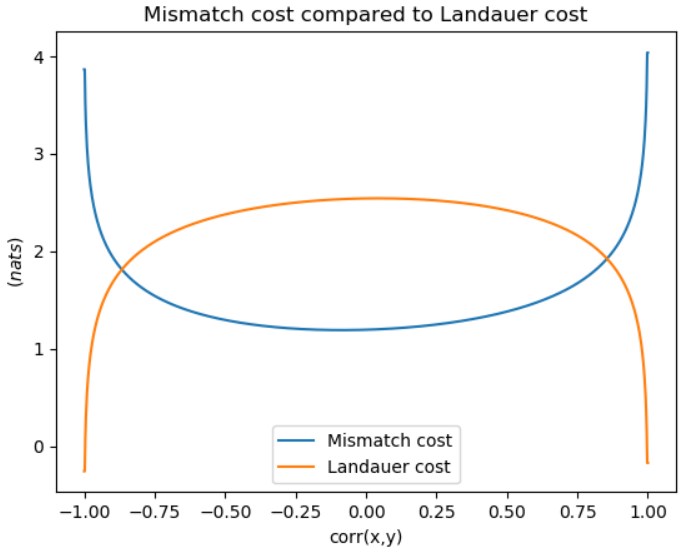


Figure B.4: $\mathcal{M}(p)$ and $\mathcal{L}(p)$ for the 2-input ANN gate, with weights $= [-0.1, 3]$, $\sigma_{11} = 5$ and $\sigma_{22} = 3$.

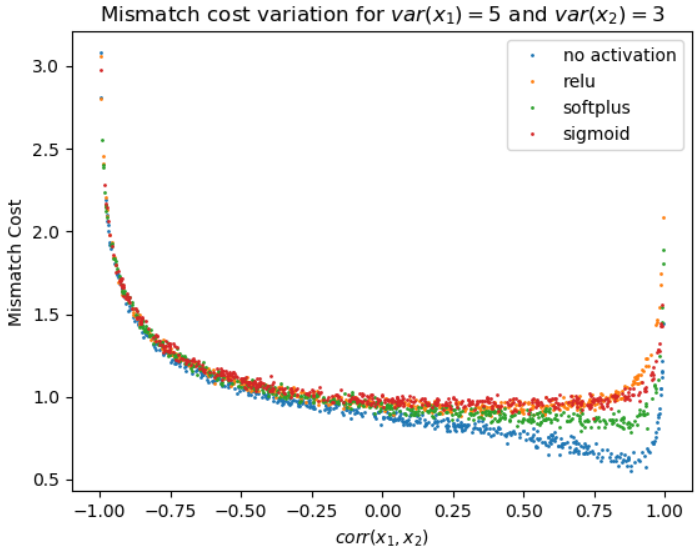


Figure B.5: Mismatch cost for inputs that take values in the interval $[-5, 5]$ with weights = [1.5,-3].

Appendix C

Mismatch cost in complex scenarios

As the number of parameters grows with the number of inputs, a complete work of visualization can only be done through multiple graphs showing different point of views of the same problem. Here are depicted some of them, that were introduced in the main text.

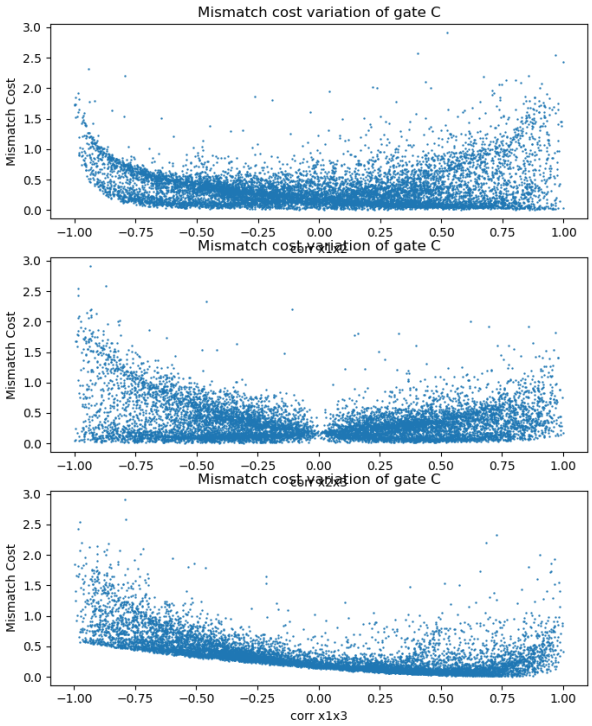


Figure C.1: Mismatch cost of gate C

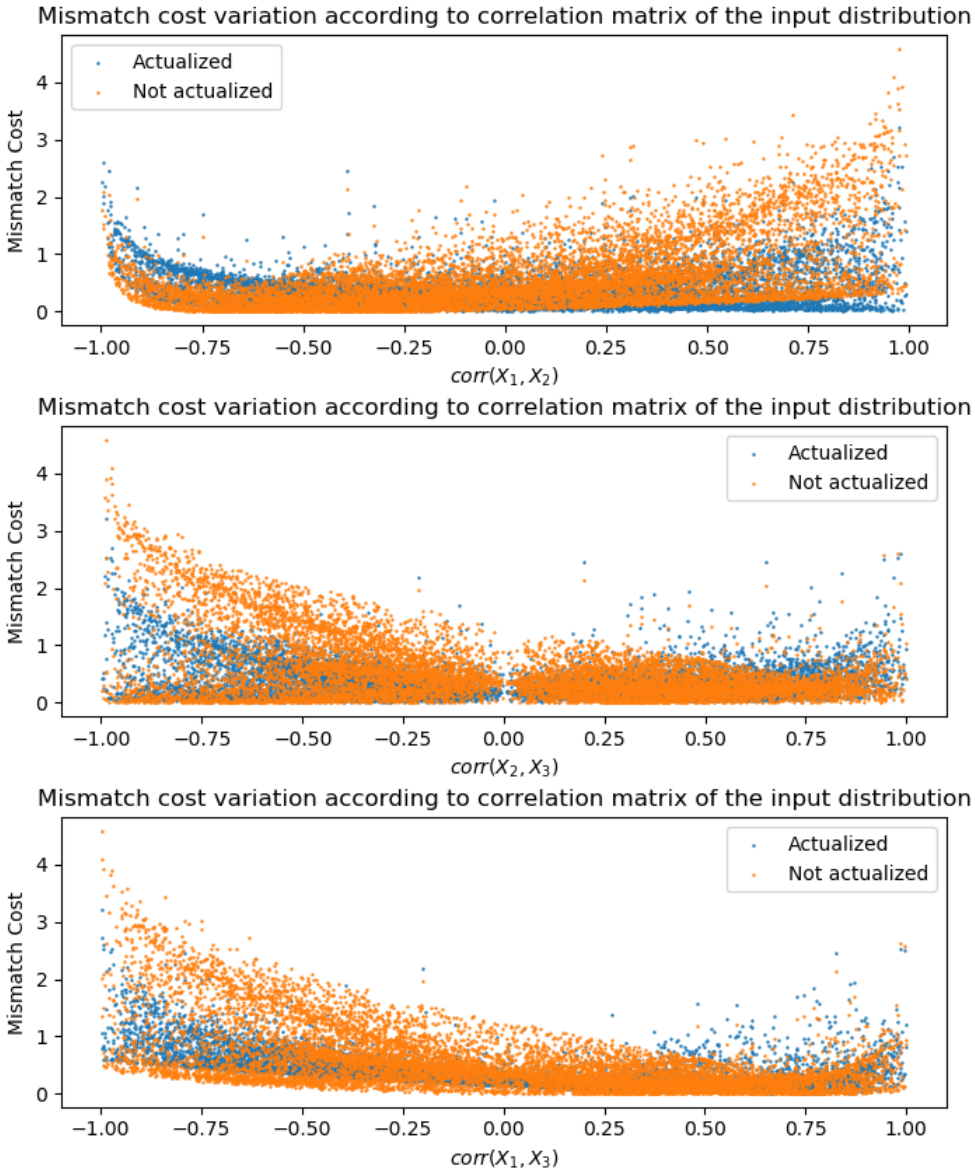


Figure C.2: Comparison of the impact of the correlations in the 3-inputs case

Neuron	Input1	Input2	Input3	Bias
Y1	-0.61	-0.409	-1.273	0.0401
Y2	-0.524	1.044	-0.592	-0.194
Y3	0.062	-0.575	-1.353	0.145
Z1	0.485	-0.758	0.606	-0.085
Z2	-1.044	0.293	-0.789	-0.278
Out	0.971	-0.435		-0.806

Table C.1: Weights for the model trained on the transformed dataset

Neuron	Input1	Input2	Input3	Bias
Y1	-0.663	-0.202	-0.164	0.
Y2	0.723	1.9	-0.8655319	0.177
Y3	1.619	0.428	-0.175	-0.095
Z1	1.084	1.562	1.152	-0.26
Z2	-0.37	1.175	1.174	-0.227
Out	-1.132	-1.18		1.15

Table C.2: Weights for the model trained on the original dataset

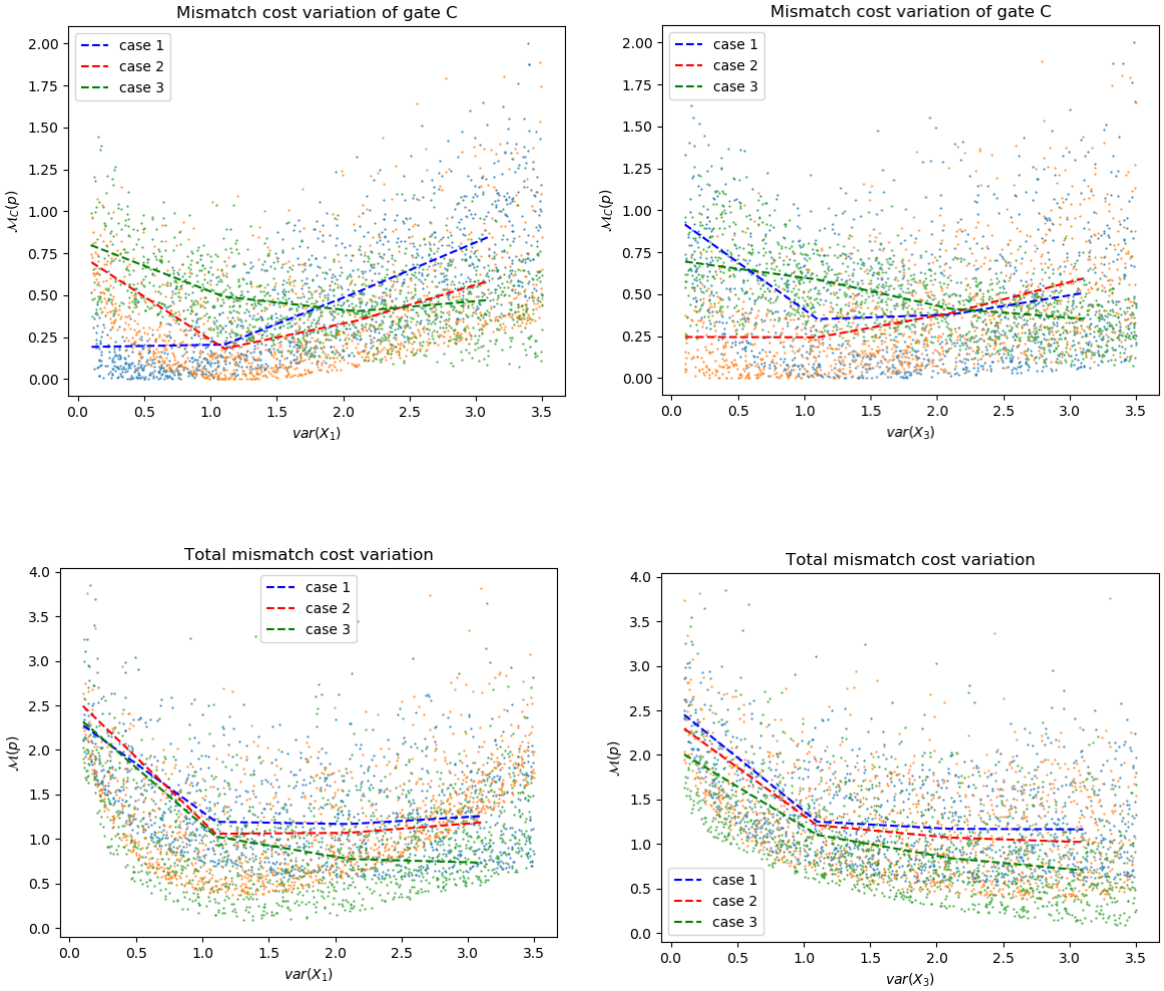


Figure C.3: $\mathcal{M}_C(p)$ and $\mathcal{M}(p)$ of the 3-inputs case for different wirings

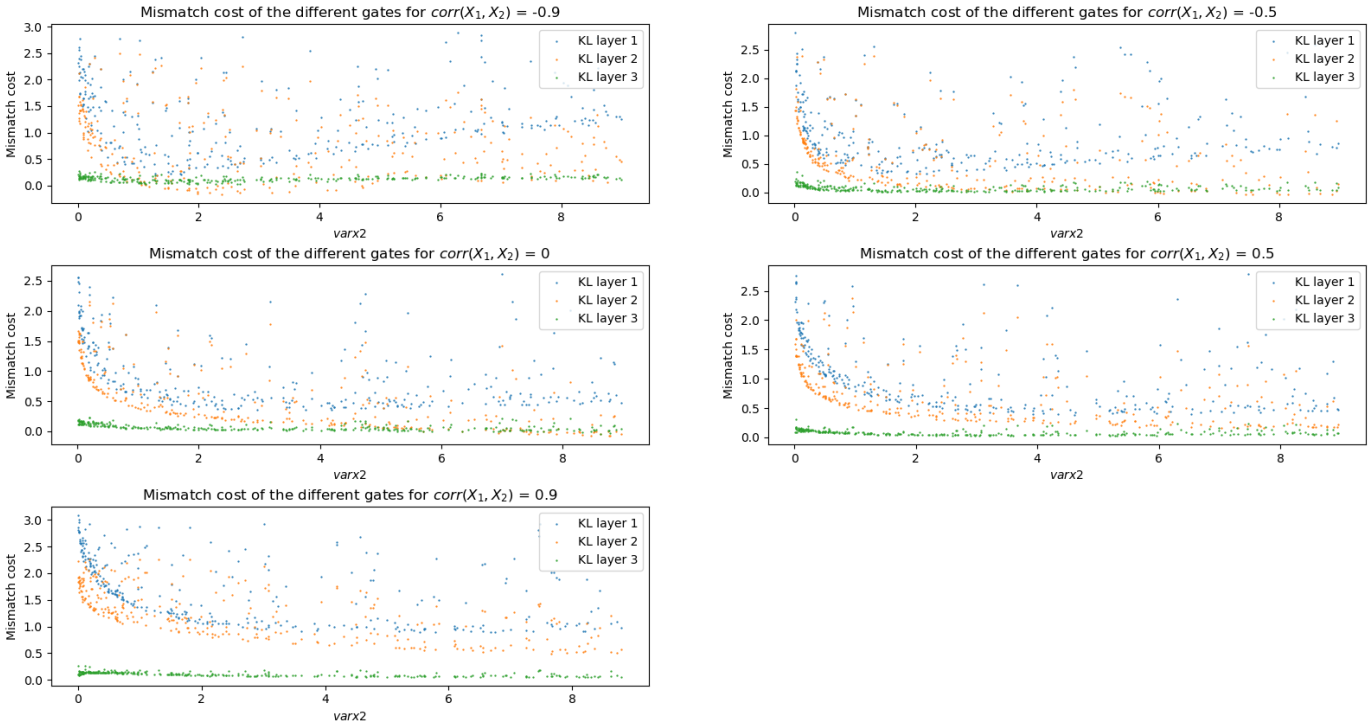
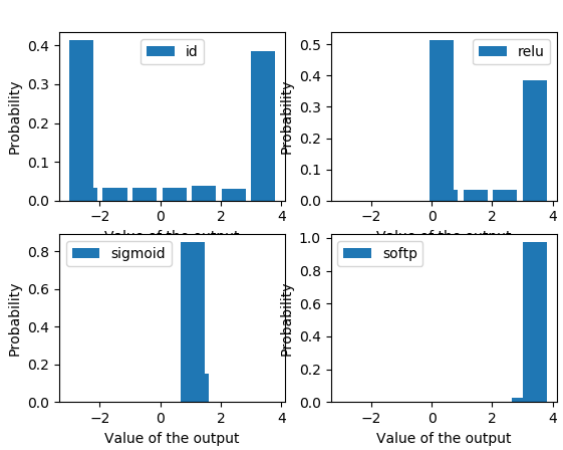
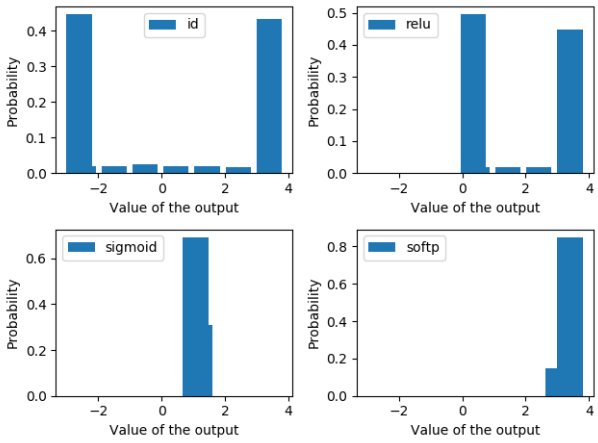


Figure C.4: Comparison of the mismatch cost of each layer for several correlation factors



a) $\mathcal{N}(0, \mathbb{1})$ as initial distribution over $[-3, 3]$



b)$U(-3, 3)$ as initial distribution over $[-3, 3]$

Figure C.5: Comparison of the propagated distribution when run throughout several layers

Bibliography

[1] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, May 2013.

[2] Hakan Özköse, Emin Sertaç Arı, and Cevriye Gencer. Yesterday, today and tomorrow of big data. *Procedia - Social and Behavioral Sciences*, 195:1042–1050, July 2015.

[3] Kenneth Cukier and Viktor Mayer-Schönberger. The rise of big data: How it's changing the way we think about the world. In *The Best Writing on Mathematics 2014*, pages 20–32. Princeton University Press, December 2014.

[4] Stephanie E Hampton, Carly A Strasser, Joshua J Tewksbury, Wendy K Gram, Amber E Budden, Archer L Batcheller, Clifford S Duke, and John H Porter. Big data and the future of ecology. *Frontiers in Ecology and the Environment*, 11(3):156–162, April 2013.

[5] Martin L. Parry. *Climate Change and World Agriculture*. Routledge, October 2019.

[6] Thomas E. Lovejoy and Lee Hannah, editors. *Biodiversity and Climate Change*. Yale University Press, 2019.

[7] C. Dorland, R. S. J. Tol, and J. P. Palutikof. *Climatic Change*, 43(3):513–535, 1999.

[8] E. Bevacqua, D. Maraun, M. I. Vousdoukas, E. Voukouvalas, M. Vrac, L. Mentaschi, and M. Widmann. Higher probability of compound flooding from precipitation and storm surge in europe under anthropogenic climate change. *Science Advances*, 5(9):eaaw5531, September 2019.

[9] William K. Michener and Matthew B. Jones. Ecoinformatics: supporting ecology as a data-intensive science. *Trends in Ecology & Evolution*, 27(2):85–93, February 2012.

[10] David H Wolpert. The stochastic thermodynamics of computation. *Journal of Physics A: Mathematical and Theoretical*, 52(19):193001, April 2019.

[11] David H Wolpert and Artemy Kolchinsky. Thermodynamics of computing with circuits. *New Journal of Physics*, 22(6):063047, jun 2020.

[12] Pritam Chattopadhyay and Goutam Paul. Revisiting thermodynamics in computation and information theory, 2021.

[13] Charles H. Bennett. The thermodynamics of computation—a review. *International Journal of Theoretical Physics*, 21(12):905–940, December 1982.

[14] Charles H. Bennett. Notes on landauer's principle, reversible computation, and maxwell's demon. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 34(3):501–510, 2003. Quantum Information and Computation.

[15] Juan M. R. Parrondo, Jordan M. Horowitz, and Takahiro Sagawa. Thermodynamics of information. *Nature Physics*, 11(2):131–139, February 2015.

[16] *Maxwell's Demon: Entropy, Information, Computing*. Princeton University Press, 1990.

[17] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006.

[18] Joan A. Vaccaro and Stephen M. Barnett. Information erasure without an energy cost. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2130):1770–1778, Jan 2011.

[19] Edward Bormashenko. Generalization of the landauer principle for computing devices based on many-valued logic. *Entropy*, 21(12), 2019.

[20] John Goold, Mauro Paternostro, and Kavan Modi. Nonequilibrium quantum landauer principle. *Phys. Rev. Lett.*, 114:060602, Feb 2015.

[21] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp, 2019.

[22] Charles H. Bennett. Demons, engines and the second law. *Scientific American*, 257(5):108–117, 1987.

[23] L. Szilard. Über die entropieverminderung in einem thermodynamischen system bei eingriffen intelligenter wesen. *Zeitschrift für Physik*, 53(11-12):840–856, November 1929.

[24] Massimiliano Esposito, Upendra Harbola, and Shaul Mukamel. Nonequilibrium fluctuations, fluctuation theorems, and counting statistics in quantum systems. *Reviews of Modern Physics*, 81(4):1665–1702, December 2009.

[25] Md. Manirul Ali, Wei-Ming Huang, and Wei-Min Zhang. Quantum thermodynamics of single particle systems. *Scientific Reports*, 10(1), August 2020.

[26] Udo Seifert. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Reports on Progress in Physics*, 75(12):126001, nov 2012.

[27] Jordan M. Horowitz and Massimiliano Esposito. Work producing reservoirs: Stochastic thermodynamics with generalized gibbs ensembles. *Phys. Rev. E*, 94:020102, Aug 2016.

[28] S.M. Ross. *Introduction to Probability Models*. Elsevier, 2019.

[29] Continuous-time markov chains i. In *Markov Chains*, pages 60–107. Cambridge University Press, February 1997.

[30] Christian Maes. Local detailed balance, 2020.

[31] C. Van den Broeck and M. Esposito. Ensemble and trajectory thermodynamics: A brief introduction. *Physica A: Statistical Mechanics and its Applications*, 418:6–16, 2015. Proceedings of the 13th International Summer School on Fundamental Problems in Statistical Physics.

[32] M. Esposito and C. Van den Broeck. Second law and landauer principle far from equilibrium. *EPL (Europhysics Letters)*, 95(4):40004, aug 2011.

[33] Thomas M. Covert. 5 which processes satisfy the second law ? 2014.

[34] Jeremy A Owen, Artemy Kolchinsky, and David H Wolpert. Number of hidden states needed to physically implement a given conditional distribution. *New Journal of Physics*, 21(1):013022, jan 2019.

[35] G. L. Shaw. Donald hebb: The organization of behavior. In Günther Palm and Ad Aertsen, editors, *Brain Theory*, pages 231–233, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

[36] Paul Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math. Harvard University*. PhD thesis, 01 1974.

[37] David H Wolpert. Minimal entropy production rate of interacting systems. *New Journal of Physics*, 22(11):113013, nov 2020.

[38] E. T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106:620–630, May 1957.

[39] E. T. Jaynes. Information theory and statistical mechanics. ii. *Phys. Rev.*, 108:171–190, Oct 1957.

[40] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[41] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[42] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**