Lorenzo Valerio

Stochastic mobility models in space and time

Ledizioni LediPublishing

Lorenzo Valerio
Department of Mathematics
University of Milan
Via C. Saldini n.50
20133 Milan - Italy

Advisor:
Prof. Bruno Apolloni
Department of Computer Science
University of Milan
Via Comelico 39, Milan, Italy

Series Editors:
Giovanni Naldi,
Coordinator of the PhD Program in Mathematics and Statistics for Computational Sciences;
Bert van Geemen,
Coordinator of the PhD Program in Mathematics;
Luca Barbieri Viale,
Director of the Graduate School.

Editorial Assistant: Stefania Leonardi

© 2013 Lorenzo Valerio
Open Access copy available at air.unimi.it
First Edition: January 2013
ISBN 978-88-6705-071-0
All rights reserved for the print copy.
For the digital copy rights please refer to air.unimi.it

Ledizioni LediPublishing
Via Alamanni 11 – 20141 Milano – Italy
http://www.ledipublishing.com
info@ledipublishing.com

UNIVERSITÀ DEGLI STUDI DI MILANO

GRADUATE SCHOOL IN MATHEMATICAL SCIENCES

DEPARTMENT OF MATHEMATICS "FEDERIGO ENRIQUES"

PHD COURSE IN
MATHEMATICS AND STATISTICS FOR COMPUTATIONAL SCIENCES
XXIV° CICLE

Doctoral Thesis

Stochastic mobility models in space and time

Settori scientifico-disciplinari: MAT /06 - INF /01

Author
Lorenzo VALERIO

Supervisor
Prof. Bruno APOLLONI

PhD Programme Coordinator
Prof. Giovanni NALDI

Academic Year 2010/11

Contents

1 Introduction 1

2 Mathematical tools 3
2.1 Lèvy processes 4
2.1.1 Preliminary notions 4
2.1.2 Lèvy family 5
2.1.3 Infinite divisibility 5
2.1.4 Stable random variables 8
2.2 Point processes and Palm Calculus 11
2.2.1 Poisson Point Process 11
2.2.2 Palm Theory 14
2.2.3 Stationarity 15
2.3 Statistical framework 17
2.3.1 Sampling mechanism 17
2.3.2 Population Bootstrap 19
2.4 Artificial Neural Network 20
2.4.1 Basic definitions 20
2.4.2 Perceptron 22
2.4.3 Gradient descent minimization method 23
2.4.4 The back-propagation algorithm 24
2.5 BICA: Boolean Independent Component Analysis 25
2.5.1 A peculiar clustering framework 25
2.5.2 The representation problem 26
2.5.3 The Boolean option 28

3 Capturing aggregation dynamics in space and time 31
3.1 The dynamics of a social community 32
3.2 Processes with memory 33
3.2.1 The Pareto family 35
3.3 Reasoning about scalable processes 37
3.3.1 The sampling mechanism benefits 41
3.4 A Model to maintain memory in a time process 42
3.4.1 Completing the mobility model 47
3.5 Real world benchmarks and artificial datasets 48
3.5.1 Benchmarks from the WEB 50
3.5.2 The ground truth 53
3.5.3 The artificial dataset 55
3.6 Fitting the model traces 57
3.6.1 The Statistical Bases 57
3.6.2 Testing the inference algorithm 60
3.6.3 An overall evaluation 64
3.7 Understanding the mobility model 66
3.8 Contrasting the literature 70
3.9 Concluding remarks 73

4 Information driven dynamics 75
4.1 A biological insight of intentionality 76
4.2 Moving from biological neural networks to Artificial Neural Networks . 80
4.3 An Artificial Neural Network with mobile neurons 80
4.3.1 The model 81
4.3.2 Potentials 82
4.3.3 Error term 86
4.4 Training the network 86
4.5 Numerical Experiments 87
4.5.1 Assessing the new training procedure 89
4.5.2 Discussing some results 92
4.6 Further improvements 96
4.7 Concluding remarks 103

5 Conclusions 107

A Derivation rules for the back-propagation algorithm 109
A.1 Potential derivatives 109
A.1.1 Gravitational attraction P_1 109
A.1.2 Elastik repulsion P_2 110
A.2 Computation of the acceleration derivative 112

A.3 Error derivatives 112
A.4 Thresholds 113
References 115

List of Figures

2.1 Graph of the function E_s with $n = 2$. 29

3.1 CCDF LogLogPlot when T follows: (a) a Pareto law with $\alpha = 1.1$ and $k = 1$; (b) a negative exponential law with $\lambda = 0.09$. Parameters were chosen to have the same mean. 35

3.2 Trajectories described by: (a) Brownian motion, (b) Lévy flights, and (c) the proposed mobility model. 42

3.3 Joint traces of two cars (plain and dashed curves respectively) when: (a) both move according to a Brownian motion behavior; (b) the former moves only in one quadrant (absolute value of the Brownian motion components) from a trigger time on; and (c) an oracle rotates this trajectory toward the other car with some approximation (quantified by the ray of a proximity circle). 43

3.4 (a) CDF Plot of a shifted-Pareto distribution, and (b) LogLogPlot representation of its complement 46

3.5 CCDF LogLogPlot of contact times with a trigger time varying according to distribution law: (a) Pareto; (b) negative exponential; and (c) uniform. Parameter $\alpha = 0.9$ in (3.41) for all distributions, while parameters specific to the various trigger distributions are set in order to have the same expected value for the latter. 47

3.6 Recovering the intercontact ECCDF shape through our mobility model. First row: CDF, second row: CCDF. Columns: different mobility parameters. Gray curves: experimental distributions; black curves: their interpolations. 49

3.7 LogLogPlot of ECCDF intercontact times for datasets CH_1 (a), CH_2 (b) and CH_3 (c). Gray curves: individual intercontacts; black curves: merge of the former. 51
3.8 Agent trajectories and tracks for HTMD and Nokia datasets, one per benchmark in Table 3.1. Gray ECCDF curves: individual inter-contact times; black ECCDF curves: merge of the former ones. 52
3.9 PTR architecture. 54
3.10 Tracks from the PTR datasets. Gray ECCDF curves: individual inter-contact times; black ECCDF curves: merge of the former ones. 55
3.11 Agent trajectories and tracks from the Artificial datasets. Gray ECCDF curves: individual intercontact times; black ECCDF curves: merge of the former ones. 56
3.12 Curves fitting with compatible parameters. (a) Sample size: 30; (b) sample size: 500. Thick plain curves: sample ECCDF; gray curves: 200 population replicas; thick dashed curves: median of the replicas. Light gray region: 0.90 confidence region. 59
3.13 Fitting agent tracks drawn from the dataset in Table 3.1 through our Shifted-Pareto distribution. First row $\rightarrow$ 0.90 confidence region and median curve for single agents; same notation as in Fig. 3.12. Second row$\rightarrow$ merge track of all agents from the same dataset (gray curve) and its fit (black dashed curve). 61
3.14 ECDF of samples drawn according to the sampling mechanism $\mathcal{U}_i \equiv u_i^{\left(\frac{u_{i-1}}{k}\right)^r}$ (gray curves) and $\mathcal{U}_i \equiv u_i^d$ (black curves) when: (a) $r = -1, h = 1, d = 4, \rho_{U_i,U_{i+1}} = -0.24$; and (b) $r = 1, h = 3, d = 0.25, \rho_{U_i,U_{i+1}} = 0.37$. 62
3.15 Discovering the seed bias from a correlation rooted on reconstructed data. (a) The reconstruction mechanism. (b) Course of correlation with the biasing exponent. 63
3.16 Relation between s-parameters and m-parameters in the artificial dataset. Surfaces: best fitting curves; points: s- and m-parameters. 67
3.17 CCDF LogPlot of shifted-Pareto distribution (3.42) with a ranging from 2 (black curve) to 3.6 (light gray curve). 67
3.18 Mobility parameters emerging from smearing the s-parameters of our experimental benchmark tracks on the surfaces in Fig. 3.16. First three columns $\rightarrow$ gray points: same simulated parameters as in Fig. 3.16; black points: replicas compatible with the processed dataset; white points: median parameters among the replicas (graphically hidden in general by the former ones). Last column $\rightarrow$ gray points: mobility parameters colored according to the cluster they belong to; bullets: cluster centroids. 69

3.19 CCDF LogLogPlot of a T randomly varying according to: (a) a double Pareto; (b) a tapered Pareto. 71
3.20 Comparison between median curves (thick black) and the best fitting ones (dashed black) computed according to: (a) CvM test, and (b) Akaike criterion, for a $\text{Helsinki}_{\text{dense}}$ track. Same notation as in Fig. 3.13. 73

4.1 (Up-left) The wall of the brain vesicles Initially consists of only two layers, the marginal zone and the ventricular zone. Each cell performs a characteristic ”dance” as it divides, shown here from left to right. The circled numbers correspond to the five ”positions” described in the text. The fate of the daughter cells depends On the plane of cleavage during division. (Up-right) After cleavage in the vertical plane. both daughters remain in the ventricular zone to divide again. (Down-right) After cleavage in the horizontal plane, the daughter farthest away from the ventricle ceases further division and migrates away. 77
4.2 The first cells to migrate to the cortical plate are those that form the sub-plate. As these differentiate into neurons. the neuroblasts destined to become layer VI cells migrate past and collect in the cortical plate. This process repeats again and again until all layers of the cortex have differentiated. The sub-plate neurons then disappear. 79
4.3 Potential field generated by both attractive upward neurons (black bullets) and repulsive siblings (gray bullets). The bullet size is proportional to the strength of the field, hence either to the neuron mass (black neurons) or to the outgoing connection weight averaged similarity (gray neurons). Arrows: stream of the potential field; black contour lines: isopotential curves. 83
4.1e The initial network layouts for: (a) Pumadyn, and (b) MNIST. 90
4.7 Neuron trajectories of: (a) a not well tuned training story, and (b) a successful one where the dynamic is more gentle. Bullets: neurons of the second layer in the MNIST benchmark (grey/black: initial/final position). 90
4.9 Histograms of the state values of two typical hidden layer neurons at the end of training. 91
4.10 Course of training MSE with weight updates' number (wun) for the regression problem. Same architecture different training algorithms: light gray curve $\rightarrow$ standard back-propagation, dark gray curve $\rightarrow$ back-propagation enriched with the BICA term, black curve $\rightarrow$ our mob-neu algorithm. 92

4.11 Errors on regressing Pumadyn. Course of: (a) training MSE with weight updates' number , (b) network output with sorted target patterns (stp), and (c) sorted absolute test error of our model (black curve) and competitor mlp-mc-1 and mlp-wd-1 (gray curves) in Table 4.2. 95
4.12 The final layout of the multilayer perceptrons in Fig. 4.1e. 96
4.13 Examples of wrongly and correctly classified '9's. 97
4.14 Course of the single digits testing errors with the number of weight updates. 98
4.15 Dendritic structure in the production of digits: (a) '3', and (b) '4'. 99
4.16 Cliques of highly correlated neurons on the same digits of Fig. 4.15. Bullets: locations of all 2-nd layer neurons. 99
4.17 Example of the membership functions determining the influence received by the downward neurons from upward ones. 100
4.18 Distribution of λ coefficients on the various layers ℓs at the last training iteration. 100
4.19 Course of the train (black) and test (gray) misclassification error on MNIST dataset with an increasing number of output neurons ψ_o associated to the single target classes. 101
4.20 Example of last last layer layout in the replicated output configuration. . 103

List of Tables

3.1 Description of real world and synthetic benchmarks. 50
3.2 Synopsis of the parameters fitting the benchmark tracks. Cell values: single track column $\mapsto$ median and MAD (in brackets) of the estimated parameters within the dataset; merge track column $\mapsto$ parameters fitting this track. 64
3.3 Statistical comparison between competitor models. Rows: benchmarks; column: models; cells: CvM test acceptance rate (upper line) and Akaike criterion winning rate (lower line). 68

4.1 Gradient expressions for the backward phase from last-but-one layer down. 88
4.2 Comparison of regression performances. Raw: method; column: size of the pumadyn8-nm training set; cell: MSE average and standard deviation (in brackets) 94
4.3 Confusion matrix of the MNIST classifier. 97
4.4 Confusion matrix of the MNIST classifier on $60,000$ train samples. 102
4.5 Confusion matrix of the MNIST classifier on $10,000$ test samples in the *replicated output experiment* 102
4.6 Comparison between several methodologies applied on MNIST dataset [77]. All the techniques do not use preprocessed data. 105

Chapter 1
Introduction

An interesting fact in nature is that if we observe agents (neurons, particles, animals, humans) behaving, or more precisely moving, inside their environment, we can recognize - tough at different space or time scales - very specific patterns. The existence of those patterns is quite obvious, since not all things in nature behave totally at random, especially if we take into account thinking species like human beings. On the contrary, their analysis is quite challenging. Indeed, during the years we can find in the literature a lot of efforts to understand the behavior of complex systems through mathematical laws. If a first phenomenon which has been deeply modeled is the gas particle motion [68] as the template of a totally random motion, other phenomena, like foraging patterns of animals such as albatrosses [44], and specific instances of human mobility [56] wear some randomness away in favor of deterministic components. Thus, while the particle motion may be satisfactorily described with a Wiener Process (also called Brownian motion) - hence particle coordinates distributed like Gaussian variables with variance increasing with time - other phenomena like albatrosses' foraging patterns or human mobility are better described by other kinds of stochastic processes called *Levy Flights*. Both may be simulated at discrete time with an infinite sum of equally distributed steps. But the former are Gaussian, the latter have heavy tailed distributions like Cauchy or Pareto distributions. Many researcher did try to explain this peculiar aspects in terms of stochastic models often based on complex superstructures, yet only seldom they got a strong plausibility. Minding at these phenomena in a unifying way, in terms of motion of agents – either inanimate like the gas particles, or animated like the albatrosses –the point is that the latter are driven by specific interests, possibly converging into a common task, to be accomplished. Since the moment when these needs arise, agents move from one place to another moulding long or even short lasting communities, whose importance has led the scientific community to terms as *social communities*. These kind of spontaneous groups

are as interesting as the motions *per se*: indeed we can consider social communities as a real effect of the intentional (either physical or virtual) motion.

The whole thesis work turns around the concept of agent intentionality at different scales, whose model may be used as key ingredient in the statistical description of complex behaviors. The two main contributions in this direction are:

1. the development of a "wait and chase" model of human mobility having the same two-phase pattern as animal foraging [44] but with a greater propensity of local stays in place and therefore a less dispersed general behavior [13];
2. the introduction of a mobility paradigm for the neurons of a multilayer neural network and a methodology to train these new kind of networks to develop a collective behavior. The lead idea is that neurons move toward the most informative mates to better learn how to fulfill their part in the overall functionality of the network.

With these specific implementations we have pursued the general goal of attributing both a cognitive and a physical meaning to the intentionality so as to be able in a near future to speak of intentionality as an additional potential in the dynamics of the masses (both at the micro and a the macro-scale), and of communication as another network in the force field. This could be intended as a step ahead in the track opened by the past century physicists [50] with the coupling of thermodynamic and Shannon entropies in the direction of unifying cognitive and physical laws.

This thesis is organized as follows:

- Chapter 2 is intended as a compendium of useful preliminary notions and concepts to read the rest of the thesis. It comprises both the necessary mathematical foundations and the main basic definitions to support next chapters, without any exhaustiveness pretense.
- Chapter 3 runs along a path starting from some theoretical consideration on stochastic processes with and without memory and the relationship lying between them. Then, it continues with the presentation of a human mobility model that supports a real application embedded in concrete contexts - the opportunistic networks, gps communities, etc. - as an implementation of the previous theoretical considerations.
- Chapter 4 prosecutes the study at a micro scale. Here, taking inspiration by brain morphogenesis, we add a mobility functionality to the neurons of an artificial neural network and toss this new cognitive device on well known benchmarks. We prove the social attitude acquired by the neurons thanks to their mobility to be beneficial to the learning task of a deep architecture neural network.
- Chapter 5 contains some concluding remarks and considerations for a future work.

Chapter 2
Mathematical tools

In this chapter we will present a streamline of theoretical results which are at the basis of both the motivations and the methodologies of this thesis work.

By first, we introduce the Levy flights, as the reference point process of our mobility model. Actually, we will locate our model at an intermediate position between them and the Brownian motion, hence owning the same infinite divisibility properties. To study this process, we cannot strictly apply the tools from Palm calculus which we discuss in this chapter. Rather, we simply obtain from them some insights about an almost stationarity of the process. As a result, the relevant times follows a heavy tailed distribution law, but with a more contained variance, thus allowing for suitable statics to infer the distribution parameters. To accomplish this inference task we adopt the Algorithmic Inference framework that we recall in the third part of this chapter. Here we introduce only the basic concepts; while the actual inference procedures will be discussed at the moment of their implementation later on. Then we list in a very essential way some definition and algorithms attaining the neural networks that we will use for accomplishing the training task of our intentional agents. They are very popular notions that we recall with the unique aim of fixing the notation. We conclude this roundup with the blowup of a special expedient we use to improve the signal representation during the training of our neural network. It consists of extracting boolean features from the signal, and we show an entropic device to get them.

We just list definitions and results in order to provide the reader some pointers to topics that are covered with more details (and often a greater rigor as well) in numerous textbooks an papers in the literature.

2.1 Lèvy processes

2.1.1 Preliminary notions

Before introducing the definition of Lèvy Process, we provide some preliminary notions.

Let S be a non-empty set and $\mathscr{F}$ a collection of subsets of S. We call $\mathscr{F}$ a σ-*algebra* if the following hold.

1. $S \in \mathscr{F}$
2. $A \in \mathscr{F} \Rightarrow A^c \in \mathscr{F}$
3. If $(A_n, n \in \mathbb{N})$ is a sequence of subsets in $\mathscr{F}$ then $\bigcup_{n=1}^{\infty} A_n \in \mathscr{F}$

The pair $(S, \mathscr{F})$ is called a *measurable space*.
A *measure* on $(S, \mathscr{F})$ is a mapping $\mu : \mathscr{F} \mapsto [0, \infty]$ that satisfies:

1. $\mu(\emptyset) = 0$,
2. $$\mu\left(\bigcup_{n=1}^{\infty} A_n = \sum_{n=1}^{\infty} \mu(A_n)\right)$$

   for every sequence $(A_n, n \in \mathbb{N})$ of mutually disjoint sets in $\mathscr{F}$

The triple $(S, \mathscr{F}, \mu)$ is called a *measure space*.
For our purposes it is useful to define two measures:

- Let S be a subset of $\mathbb{R}^d$. We equip S with the topology induced from $\mathbb{R}^d$, so that $U \subseteq S$ is open in S if $U \cap S$ is open in $\mathbb{R}^d$. Let $\mathscr{B}(S)$ denote the smallest σ-algebra of subsets of S that contains every open set in S. We call $\mathscr{B}(S)$ the *Borel* $\sigma - algebra$ *of* S. Elements of $\mathscr{B}(S)$ are called *Borel sets* and any measure on $(S, \mathscr{B}(S))$ is called a *Borel measure*. The linear space of all bounded Borel measurable functions from S to $\mathbb{R}$ will be denoted by $\mathscr{B}_b(S)$
- Here we usually write $S = \Omega$ and take Ω to represent the set of outcomes of some random experiment. Elements of $\mathscr{F}$ are called *events* and any measure on $(\Omega, \mathscr{F})$ of total mass 1 is called *probability measure* and denoted by P. The triple $(\Omega, \mathscr{F}, P)$ is then called *probability space*.

Definition 2.1. (Measurable mapping). For $i = 1, 2$ let $(S_i, \mathscr{F}_i)$ be measurable spaces. A mapping $f : S_1 \mapsto S_2$ is said to be $(\mathscr{F}_1, \mathscr{F}_2)$-*measurable* if $f^{-1} \in \mathscr{F}_1$ for all $A \in \mathscr{F}_2$. If each $S_1 \subseteq \mathbb{R}^d, S_2 \subseteq \mathbb{R}^m$ and $\mathscr{F}_i = \mathscr{B}(S_i)$, f is said to be *Borel measurable*.

Remark 2.1. In what follows, we will speak only of measurable mappings that are equipped with a Borel σ-algebra.

Definition 2.2. (Random Variable). Given the probability space $(\Omega, \mathscr{F}, P)$, the measurable mapping $X : \Omega \mapsto \mathbb{R}^d$ is called *random variable*.

2.1.2 Lèvy family

Definition 2.3. (Stochastic Process). A *stochastic process*

$$\{X_t\}_{t\in T}$$

is a collection of random variables X_t, taking values in a common measure space $(S,\mathscr{F})$, indexed by a set T.

Lèvy processes are essentially stochastic processes with stationary and independent increments. Their importance in probability theory stems from the following facts:

- they are the analogues of random walks in continuous time;
- they form special subclasses of both semi-martingales and Markov processes for which the analysis is on the one hand much simpler and on the other hand provides valuable guidance for the general case;
- they are the simplest examples of random motion whose sample paths are right-continuous and have a number (at most countable) of random jump discontinuities occurring at random times, on each finite time interval.
- they include a number of very important processes as special cases, including Brownian motion, the Poisson process, stable and self decomposable processes.

Definition 2.4. Suppose that we are given a probability space $(\Omega,\mathscr{F},P)$. A Lèvy process $X=(X(t),t\geq 0)$ taking values in $\mathbb{R}^d$ is essentially a stochastic process having stationary and independent increments; we always assume that $X(0)=0$ with probability 1. So:

- each $X(t)$ is define d in $\Omega\mapsto\mathbb{R}^d$;
- given any selection of distinct time-points $0\leq t_1<t_2<\cdots<t_n$, the random vectors $X(t_1),X(t_2)-X(t_1),X(t_3)-X(t_2),\ldots,X(t_n)-X(t_{n-1})$ are all independent;
- given two distinct times $0\leq s<t<\infty$, the probability distribution of $X(t)-X(s)$ coincides with that of $X(t-s)$.

2.1.3 Infinite divisibility

2.1.3.1 Convolution of measures

Definition 2.5. Let $\mathscr{M}(\mathbb{R}^d)$ denote the set of all Borel probability measures on $\mathbb{R}^d$. We define the convolution of two probability measures as follows:

$$(\mu_1 * \mu_2)(A) = \int_{\mathbb{R}^d} (A - x)\mu_2(dx) \tag{2.1}$$

for each $\mu_i \in \mathscr{M}(\mathbb{R}^d),\ i = 1,2$,and each $A \in \mathscr{B}(\mathbb{R}^d)$, where we note that $A - x = \{y - x, y \in A\}$.

Proposition 2.1. *The convolution $\mu_1 * \mu_2$ is a probability measure on $\mathbb{R}^d$.*

Proposition 2.2. *If $f \in \mathscr{B}_b(\mathbb{R}^d)$, then for all $\mu_i \in \mathscr{M}(\mathbb{R}^d),\ i = 1,2,3$,*

$$\int_{\mathbb{R}^d} f(y)(\mu_1 * \mu_2)(dy) = \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} f(x+y)\mu_1(dy)\mu_2(dx),$$

$$\mu_1 * \mu_2 = \mu_2 * \mu_1,$$

$$(\mu_1 * \mu_2)\mu_3 = \mu_1 * (\mu_2 * \mu_3).$$

Now let X_1 and X_2 be independent random variables defined on a probability space $(\Omega, \mathscr{F}, P)$ with joint distribution p and marginals μ_1 and μ_2 respectively.

Corollary 2.1. *For each $f \in \mathscr{B}_b(\mathbb{R}^d)$,*

$$\mathbf{E}(f(X_1 + X_2)) = \int_{\mathbb{R}^d} f(z)(\mu_1 * \mu_2)(dz). \tag{2.2}$$

By Corollary 2.1, we see that convolution gives the probability law for the sum of two independent random variables X_1 and X_2, i.e.

$$P(X_1 + X_2 \in A) = \mathbf{E}(\chi_A(X_1 + X_2)) = (\mu_1 * \mu_2)(A) \tag{2.3}$$

where χ_A is the *indicator function* defined for any $A \in \mathscr{F}$ by

$$\chi_A(x) = \begin{cases} 1, \text{ if } x \in A \\ 0, \text{ if } x \notin A \end{cases}$$

Proposition 2.2 tells us that $\mathscr{M}(\mathbb{R}^d)$ is an abelian semigroup under $*$ in which the identity element is given by the Dirac measure δ_0, where we recall that in general, for $x \in \mathbb{R}^d$,

$$\delta_x(A) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ otherwise} \end{cases}$$

for any Borel set A, so we have $\delta_0 * \mu = \mu * \delta_0 = \mu$ for all $\mu \in \mathscr{M}(\mathbb{R}^d)$.

We define $\mu^{*^n} = \mu * \cdots * \mu$ (n times) and say that μ has a *convolution n-th root*, if there exists a measure denoted $\mu^{1/n} \in \mathscr{M}(\mathbb{R}^d)$ for which $(\mu^{1/n})^{*^n} = \mu$.

Remark 2.2. In general, the convolution n-th root of a probability measure may not be unique. However, it is always unique when the measure is infinitely divisible

Definition 2.6. Infinite divisibility. Let X be a random variable taking values in $\mathbb{R}^d$ with law μ_X. We say that X is infinitely divisible if, for all $n \in \mathbb{N}$, there exist i.i.d. random variables $Y_1^{(n)}, \ldots, Y_n^{(n)}$ such that

$$X = Y_1^{(n)} + \cdots + Y_n^{(n)} \tag{2.4}$$

Let $\Psi_X(u) = \mathbf{E}(e^{\mathrm{i}\langle u,X\rangle})$ denote the characteristic function of X, where $u \in \mathbb{R}^d$. More generally, if $\mu \in \mathscr{M}(\mathbb{R}^d)$ then $\Psi_\mu(u) = \int_{\mathbb{R}^d} e^{\mathrm{i}\langle u,y\rangle}\mu(dy)$.

Proposition 2.3. *The following are equivalent:*

1. *X is infinitely divisible;*
2. *μ_X has a convolution n-th root that is itself the law of a random variable, for each $n \in \mathbb{N}$;*
3. *Ψ_X has an nth root that is itself the characteristic function of a random variable, for each $n \in \mathbb{N}$.*

Proposition 2.3(2) suggests that we generalize the definition of infinite divisibility as follows: $\mu \in \mathscr{M}(\mathbb{R}^d)$ is infinitely divisible if it has a convolution nth root in $\mathscr{M}(\mathbb{R}^d)$ for each $n \in \mathbb{N}$.

2.1.3.2 Lèvy-Khintchine formula

Now we will present a formula, first established by Paul Lèvy and A.Ya. Khintchine in the 1930s, which gives a characterization of infinitely divisible random variables through their characteristic functions. First we need a definition.

Definition 2.7. Let ν be a Borel measure defined on $\mathbb{R}^d - \{0\} = \{x \in \mathbb{R}^d, x \neq 0\}$.We say that it is a *Lèvy measure* if

$$\int_{\mathbb{R}^d - \{0\}} \frac{|\mathbf{y}|^2}{1+|\mathbf{y}|^2}\nu(dy) < \infty \tag{2.5}$$

for each $y \in \mathbb{R}^d$.

Theorem 2.1. ***(Lèvy-Khintchine)*** *$\mu \in \mathscr{M}(\mathbb{R}^d)$ is infinitely divisible if there exist a vector $\mathbf{b} \in \mathbb{R}^d$, a positive definite symmetric $d \times d$ matrix A and a Lèvy measure ν on $\mathbb{R}^d - \{0\}$ such that, for all $\mathbf{u} \in \mathbb{R}^d$,*

$$\psi_m(\mathbf{u}) = \exp\left\{ \mathrm{i}\langle \mathbf{b}, \mathbf{u}\rangle - \frac{1}{2}\langle \mathbf{u}, A\mathbf{u}\rangle + \int_{\mathbb{R}^d - \{0\}} \left[e^{\mathrm{i}\langle \mathbf{u}, \mathbf{y}\rangle} - 1 - \mathrm{i}\langle \mathbf{u}, \mathbf{y}\rangle \chi_{\hat{B}}(\mathbf{y}) \right] \nu(dy) \right\} \quad (2.6)$$

where $\hat{B} = B_1(0)$[1]. Conversely, any mapping of the form (2.6) is the characteristic function of an infinitely divisible probability measure on $\mathbb{R}^d$.

The Lèvy-Khintchine formula represents all infinitely divisible random variables as arising through the interplay between Gaussian and Poisson distributions. So, a vast set of different behavior appears between these two extreme cases.

2.1.4 Stable random variables

We consider the general central limit problem in dimension $d = 1$, so let $(Y_n, n \in \mathbb{N})$ be a sequence of real-valued random variables and construct the sequence $(S_n, n \in \mathbb{N})$ of rescaled partial sums

$$S_n = \frac{Y_1 + Y_2 + \cdots + Y_n - b_n}{\sigma_n}$$

where $(b_n, n \in \mathbb{N})$ is an arbitrary sequence of real numbers and $(\sigma_n, n \in \mathbb{N})$ an arbitrary sequence of positive numbers. We are interested in the case where there exists a random variable X for which

$$\lim_{n\to\infty} P(S_n \leq x) = P(X \leq x) \quad (2.7)$$

for all $x \in \mathbb{R}$, i.e. $(S_n, n \in \mathbb{N})$ converges in distribution to X. If each $b_n = nm$ and $\sigma_n = \sqrt{n}\sigma$ for fixed $m \in \mathbb{R}, \sigma > 0$, then $X \sim N(m, \sigma^2)$by the Laplace-de-Moivre central limit theorem.
More generally a random variable is said to be stable if it arises as a limit, as in (2.7). It is not difficult to show that (2.7) is equivalent to the following. There exist real-valued sequences $(c_n, n \in \mathbb{N})$ and $(d_n, n \in \mathbb{N})$ with each $c_n > 0$ such that

$$X_1 + X_2 + \cdots + X_n = c_n X + d_n \quad (2.8)$$

where $X_1, X_2, \ldots, X_n$ are independent copies of X. In particular, X is said to be strictly stable if each $d_n = 0$.

[1] The open ball of radius r centered at x in $\mathbb{R}^d$ is denoted $B_r(x) = \{y \in \mathbb{R}^d; |y - x| < r\}$ and we will write $\hat{B} = B_1(0)$.

To see that (2.8) $\Rightarrow$ (2.7) take each $Y_j = X_j, b_n = d_n$ and $\sigma_n = c_n$. In fact it can be shown that the only possible choice of c_n in (2.8) is of the form $\sigma n^{1/\alpha}$, where $0 < \alpha \leq 2$. The parameter α plays a key role in the investigation of stable random variables and is called the *index of stability*. It follows immediately from (2.8) that all stable random variables are infinitely divisible. The characteristic functions in the Lèvy-Khintchine formula are given by the following result.

Theorem 2.2. *If X is a stable real-valued random variable, then its characteristic function must take, according to (2.6), one of the two following forms:*

1. *when $\alpha = 2$, $v = 0$, so $X \sim N(\mathbf{b}, A)$;*
2. *when $\alpha \neq 2$, $A = 0$ and*

$$\nu(dx) = \frac{c_i}{x_{1+\alpha}} \chi_{(0,\infty)}(x)dx + \frac{c_2}{|x|^{1+\alpha}} \chi_{(-\infty,0)}(x)dx,$$

where $c_1 \geq 0, c_2 \geq 0$ and $c_1 + c_2 > 0$

A careful transformation of the integrals in the Lèvy-Khintchine (2.6) formula gives a different form for the characteristic function, which is often more convenient.

Theorem 2.3. *A real-valued random variable X is stable if and only if there exist $\sigma > 0, -1 \leq \beta \leq 1$ and $\mu \in \mathbb{R}$ such that for all $u \in \mathbb{R}$:*

- *when $\alpha = 2$,*

$$\psi_X(u) = \exp\left(\mathrm{i}u\mu - \frac{1}{2}\sigma^2 u^2\right);$$

- *when $\alpha \neq 1, 2$,*

$$\psi_X(u) = \exp\left\{\mathrm{i}\mu u - \sigma_\alpha |u|^\alpha \left[1 - \mathrm{i}\beta \operatorname{sgn}(u) \tan\left(\frac{\pi\alpha}{2}\right)\right]\right\};$$

- *when $\alpha = 1$,*

$$\psi_X(u) = \exp\left\{\mathrm{i}\mu u - \sigma|u| \left[1 + \mathrm{i}\beta \frac{2}{\pi} \operatorname{sgn}(u) \log(|u|)\right]\right\}. \tag{2.9}$$

It can be shown that $\mathbf{E}(X^2) < \infty$ if and only if $\alpha = 2$ (i.e. X is Gaussian) and that $\mathbf{E}(|X|) < \infty$ if and only if $1 < \alpha \leq 2$. All stable random variables have densities f_X, which can in general be expressed in a series form. In three important cases, there are closed forms.

1. **The normal distribution**

$$\alpha = 2 \qquad X \sim N(\mu, \sigma^2); \tag{2.10}$$

2. **The Cauchy distribution**

$$\alpha = 1, \beta = 0, \quad f_X(x) = \frac{\sigma}{\pi[(x-\mu)^2 + \sigma^2]} \tag{2.11}$$

3. **The Lèvy distribution**

$$\alpha = 1, \beta = 1,$$
$$f_X(x) = \left(\frac{\sigma}{2\pi}\right)^{1/2} \frac{1}{(x-\mu)^{3/2}} \exp\left[\frac{-\sigma}{2(x-\mu)}\right] \quad \text{for } x > \mu. \tag{2.12}$$

As we can notice, stable distributions are both Normal and Cauchy distributions, the latter belongs to the family of the so called Heavy-tailed distributions (not exponentially bounded) so as the Pareto distribution. However it is not always possible to have an analytical form for a general alpha stable distribution, and in these cases we can only write the corresponding characteristic function ψ expressed as the Fourier transform of the probability density.

More precisely, note that if a stable random variable is symmetric then Theorem 2.3 yields

$$\psi_X(u) = \exp(-\rho^\alpha |u|^\alpha) \quad \forall\, 0 < \alpha \leq 2$$

where $\rho = \sigma$ for $0 < \alpha < 2$ and $\rho = \sigma\sqrt{2}$ when $\alpha = 2$.

One of the reasons why stable laws are so important is the decay properties of the tails. For example, with $\alpha = 2$ we have an exponential decay, indeed for a standard normal X

$$P(X > y) \sim \frac{e^{-y^2/2}}{\sqrt{2\pi}y} \quad \text{as } y \to \infty$$

. On the other side, when $\alpha \neq 2$ we have a slower, polynomial, decay as expressed by:

$$\lim_{y\to\infty} y^\alpha P(X > y) = C_\alpha \frac{1+\beta}{2} \sigma^\alpha$$

$$\lim_{y\to\infty} y^\alpha P(X < -y) = C_\alpha \frac{1-\beta}{2} \sigma^\alpha$$

where $C_\alpha > 1$. The relatively slow decay of the tails of non-Gaussian stable laws makes them ideally suited for modelling a wide range of interesting phenomena, some of which exhibit *long-range dependence*.

2.2 Point processes and Palm Calculus

In the following sections we will give some basic notion on point processes and Poisson point processes and how we can exploit the tools that the Palm calculus offers us.

2.2.1 Poisson Point Process

Consider the d-dimensional Euclidean space $\mathbb{R}$.

Definition 2.8. A spatial **point process** (p.p.) Φ is a random, finite of countably-infinite collection of points in the space $\mathbb{R}^d$, without accumulation points.

One can consider any given realization ϕ of a p.p as a discrete subset $\phi = \{x_i\} \subset \mathbb{R}^d$ of the space. It is often more convenient to think of ϕ as a *counting measure* or a point measure $\phi = \sum_i \delta_{x_i}$ where δ_x is the Dirac measure at x.
Consequently, $\phi(A)$ gives the number of "points" of ϕ in A. Also, for all real functions f defined on $\mathbb{R}^d$, we have $\sum_i f(x_i) = \int_{\mathbb{R}^d} f(x)\phi dx$. We will denote by $\mathbb{M}$ the set of all point measures that do not have accumulation points in $\mathbb{R}^d$. This means that any $\phi \in \mathbb{M}$ is locally finite, that is $\phi(A) < \infty$ for any bounded $A \subset \mathbb{R}^d$ (a set is bounded if it is contained in a ball with finite radius). Note that a p.p. Φ can be seen as a stochastic process $\Phi = \Phi(A)_{A\in\mathscr{B}}$ with state space $\Phi(A) \in \mathbb{N} = \{0,1,\dots\}$ and where the index A runs over bounded Borel subsets of $\mathbb{R}^d$.

Definition and Characterization

Let Λ be a locally finite non-null measure on $\mathbb{R}^d$

Definition 2.9. The **Poisson Point Process** Φ of intensity measure Λ is defined by means of its finite-dimensional distributions:

$$P\{\Phi(A_1) = n_1, \dots, \Phi(A_k) = n_k\} = \prod_{i=1}^{k} \left(e^{-\Lambda(A_i)} \frac{\Lambda(A_i)^{n_i}}{n_i!} \right) \tag{2.13}$$

for every $k = 1,2,\dots$ and all bounded, mutually disjointed sets A_i for $i = 1,\dots,k$. If $\Lambda(ds) = \lambda dx$ is a multiple of a Lebesgue measure (volume) in $\mathbb{R}^d$, we call Φ a *homogeneous Poisson p.p.* and λ is its intensity parameter.

Remark 2.3. Φ is a Poisson p.p. iff $\forall k = 1,2,\dots$ and all bounded, mutually disjoint $A_i \subset \mathbb{R}^d$ with $i = 1,\dots,k$, $(\Phi(A_1),\dots,\Phi(A_k))$, is a vector of independent Poisson random

variables of parameter $(\Lambda(A_1),\dots,\Lambda(A_k))$, respectively. In particular, $\mathbf{E}(\Phi(A)) = \Lambda(A)$ for all A.

Remark 2.4. Let W be some bounded *observation window* and let $A_1,\dots,A_k$ be some partition of this window: $A_i \cap A_j = \emptyset$ for $i \neq j$ and $\bigcup_i A_i = W$. For all $n,\ n_1,\dots,n_k \in \mathbb{N}$ with $\sum_i n_i = n$,

$$P\{\Phi(A_1) = n_1,\dots,\Phi(A_k) = n_k | \Phi(W) = n\} = \frac{n!}{n_1!\cdots n_k!}\frac{1}{\Lambda(W)^n}\prod_i \Lambda(A_i)^{n_i}. \quad (2.14)$$

Operations Preserving the Poisson Law

Superposition

Definition 2.10. The **superposition** of p.p. Φ_k is defined as the sum $\Phi = \sum_k \Phi_k$

Note that the summation in the above definition is understood as the summation of (point) measures. It always defines a point measure, which however, in general, might not be locally finite (we do not assume the last sum to have finitely many terms). Here is a useful condition to guarantee the above property.

Lemma 2.1. *The superposition $\Phi = \sum_k \Phi_k$ is a p.p. if $\sum_k \mathbf{E}[\Phi_k(\cdot)]$ is a locally finite measure.*

A refined sufficient condition may be found by the Borel-Cantelli lemma.

Proposition 2.4. *The superposition of independent Poisson point processes with intensities Λ_k is a Poisson p.p. with intensity measure $\Phi = \sum_k \Lambda_k$ if and only if the latter is a locally finite measure.*

Thinning

Consider a function $\rho : \mathbb{R}^d \mapsto [0,1]$ and a p.p Φ.

Definition 2.11. The thinning of Φ with the *retention function* ρ is a p.p. given by

$$\Phi^\rho = \sum_k \gamma_k \delta_{x_k} \quad (2.15)$$

where the random variables $\{\gamma_k\}_k$ are independent given Φ, and $P\{\gamma_k = 1|\Phi\} = 1 - P\{\gamma_k = 0|\Phi\} = p(x_k)$. Less formally, we can say that a realization of Φ^ρ can be constructed from that of Φ by randomly and independently removing some fraction of points; the probability that a given point of Φ located at x is not removed (i.e. is retained inΦ^ρ) is equal to $\rho(x)$.

Random Transformation of Points

Definition 2.12. (Probability kernel). A measure kernel from a measurable space $(\Xi, \mathscr{X})$ to another measurable space $(\Upsilon, \mathscr{Y})$ is a function $\kappa : \Xi \times \mathscr{Y} \mapsto \overline{\mathbb{R}}^{+}$[2] such that

1. for any $Y \in \Upsilon$, $\kappa(x, \mathscr{Y})$ is $\mathscr{X}$-measurable;
2. for any $x \in \Xi$, $\kappa(x, \mathscr{Y}) \equiv \kappa_x(\mathscr{Y})$ is a measure of $(\Upsilon, \mathscr{Y})$. We will write the integral of a function $f : \Upsilon \mapsto \mathbb{R}$, with respect to this measure, as $\int f(y)\kappa(x, dy)$, $\int f(y)\kappa_x(dy)$, or , most compactly, $\kappa f(y)$.

If, in addition, κ_x is a probability measure on $\Upsilon, \mathscr{Y}$ for all x, then κ is a *probability kernel*.

Consider a probability kernel $\kappa(x, B)$ from $\mathbb{R}^d$ to $\mathbb{R}^{d'}$, where $d' \geq 1$, i.e. for all $x \in \mathbb{R}^d$, $\kappa(x, \cdot)$ is a probability measure on $\mathbb{R}^{d'}$.

Definition 2.13. The *transformation* Φ^κ of a p.p. Φ by a probability kernel $\kappa(\cdot, \cdot)$ is a p.p. in $\mathbb{R}^{d'}$ given by

$$\Phi^\kappa = \sum_i \delta_{y_i} \tag{2.16}$$

where the $\mathbb{R}^{d'}$-valued random vectors $\{y_i\}_i$ are independent given Φ, with $P\{y_i \in B'|\Phi\} = p(x_i, B')$.

In other words, Φ^κ is obtained by randomly and independently displacing each point of Φ from $\mathbb{R}^d$ to some new location in $\mathbb{R}^{d'}$according to the kernel κ. This operation preserves the Poisson p.p. property as stated in the following theorem.

Theorem 2.4. ***(Displacement Theorem).*** *The transformation of the Poisson p.p. of intensity measure Λ by a probability kernel κ is the Poisson p.p. with intensity measure $\Lambda'(A) = \int_{\mathbb{R}^d} \kappa(x, A)\Lambda(dx)$, $A \subset \mathbb{R}^{d'}$.*

[2] $\overline{\mathbb{R}}^{+}$ denotes the extended real number set.

2.2.2 Palm Theory

Palm theory formalizes the notion of the conditional distribution of a general p.p. given it has a point at some location. Note that for a p.p. without a fixed atom at this particular location, the probability of the condition is equal to 0 and the basic discrete definition of the conditional probability does not apply. In this section we will outline the definition based on the Radon-Nikodỳm theorem.

We first define two measures associated with a general point process:

Definition 2.14. The **mean measure** of a p.p. Φ is the measure

$$M(A) = \mathbf{E}[\Phi(A)] \tag{2.17}$$

on $\mathbb{R}^d$. The **reduced Campbell measure** of Φ is the measure

$$C^!(A \times \Gamma) = \mathbf{E}\left[\int_A \chi(\Phi - \delta_x \in \Gamma)\Phi(dx)\right] \tag{2.18}$$

on $\mathbb{R}^d \times \mathbb{M}$, where $\mathbb{M}$ denotes the set of point measures.

Note that $M(A)$ is simply the mean number of points of Φ in A. The reduced Campbell measure $C^!(A \times \Gamma)$ is a refinement of this mean measure; it gives the expected number of points of Φ in A such that when removing a particular point from Φ, the resulting configuration satisfies property Γ. The fact that one measure is a refinement of the other, or more formally, that $C^!(\cdot \times \Gamma)$ for each Γ is absolutely continuous with respect to $M(\cdot)$, allows us to express the former as an integral of some function $P_x^!$, called the Radon-Nikodym derivative with respect to the latter:

$$C^!(A \times \Gamma) = \int_A P_x^! \, M(dx) \quad \text{forall } A \subset \mathbb{R}^d. \tag{2.19}$$

The function $P_x^! = P_x^!(\Gamma)$ depends on Γ. Moreover, if $M(\cdot)$is a locally finite measure, $P_x^!(\cdot)$can be chosen as a probability distribution in $\mathbb{M}$ for each given x.

Definition 2.15. Given a point process with a locally finite mean measure, the distribution $P_x^!(\cdot)$ is called the **reduced Palm distribution** of Φ given a point at x.

The following central formula of Palm calculus, which is called the *Campbell-Mecke* formula, is a mere rewriting of the above definition when $f(x, \nu) = \chi(x \in A, \nu \in \Gamma)$.

Theorem 2.5. ***(Reduced Campbell-Mecke Formula).*** *For all non-negative functions defined on* $\mathbb{R}^d \times \mathbb{M}$

$$\mathbf{E}\left[\int_{\mathbb{R}^d} f(x, \Phi - \delta_x)\Phi(dx)\right] = \int_{\mathbb{R}^d}\int_{\mathbb{M}} f(x, \phi)P_x^!(d\phi)M(dx) \tag{2.20}$$

Corollary 2.2. *The mean measure of a Poisson p.p. is equal to its intensity measure* $M(\cdot) = \Lambda(\cdot)$.

We now state a central result of the Palm theory for Poisson p.p. It makes clear why the reduced Palm distributions are more convenient in many situations.

Theorem 2.6. ***(Slivnyak-Mecke Theorem).*** *Let* Φ *be a Poisson p.p. with intensity measure* Λ. *For* Λ *almost all* $x \in \mathbb{R}^d$,

$$P_x^! = P\{\Phi \in \cdot\} \tag{2.21}$$

that is, the reduced Palm distribution of the Poisson p.p. is equal to its (original) distribution.

Using now the convention, according to which a p.p. is a family of random variables $\Phi = \{x_i\}_i$, which identify the locations of its atoms (according to some particular order) we can rewrite the reduced Campbell formula for Poisson p.p.

$$\mathbf{E}\left[\sum_{x_i \in \Phi} f(x_i, \Phi\{x_i\})\right] = \int_{\mathbb{R}^d} \mathbf{E}[f(x,\Phi)]M(dx) \tag{2.22}$$

2.2.3 Stationarity

Throughout this section we will use the following notation:

$$v + \Phi = v + \sum_i \delta_{x_i} = \sum_i \delta_{v+x_i}$$

Definition 2.16. A point process Φ is **stationary** if its distribution is invariant under translation through any vector $v \in \mathbb{R}^d$; i.e. $P\{v + \Phi \in \Gamma\} = P\{\Phi \in \Gamma\}$.

Proposition 2.5. *A homogeneous Poisson p.p. is stationary*

Corollary 2.3. *Given a stationary point process* Φ, *its mean measure is a multiple of Lebesgue measure:* $M(dx) = \lambda\ dx$

We have that $\lambda = \mathbf{E}[\Phi(B)]$ for any set $B \in \mathbb{R}^d$ of Lebesgue measure l. One defines the Campbell-Matthes measure of the stationary p.p. Φ as the following measure on $\mathbb{R}^d \times \mathbb{M}$:

$$C(A \times \Gamma)\mathbf{E}\left[\int_A \chi(\Phi - x \in \Gamma)\Phi(dx)\right] = \mathbf{E}\left[\sum_i \chi(x_i \in A)\chi(\Phi - x_i \in \Gamma)\right]. \tag{2.23}$$

If $\lambda < \infty$, one can define a probability measure P^0 on $\mathbb{M}$, such that

$$C(A \times \Gamma) = \lambda |A| P^0(\Gamma) \tag{2.24}$$

for all Γ

Definition 2.17. Intensity and Palm distribution of a stationary p.p. For a stationary point process Φ, we call the constant λ described in Corollary 2.3 the intensity parameter of Φ. The probability measure P^0 defined in (2.24) provided $\lambda < \infty$ is called the Palm-Matthes distribution of Φ.

One can interpret P^0 as conditional probability given that Φ has a point at the origin. Below, we always assume $0 < \lambda < \infty$. The following formula, which will often be used in what follows, can be deduced immediately from (2.24):

Corollary 2.4. ***(Campbell-Matthes formula for a stationary p.p.).*** *For a stationary point process Φ with finite, non-null intensity λ, for all positive functions g*

$$\mathbf{E}\left[\int_{\mathbb{R}^d} g(x, \Phi - x)\Phi(dx)\right] = \lambda \int_{\mathbb{R}^d} \int_{\mathbb{M}} g(x, \Phi) P^0(d\phi) dx. \tag{2.25}$$

Remark 2.5. It should not be surprising that in the case of a stationary p.p. we actually define only one conditional distribution given a point at the origin 0. One may guess that due to the stationarity of the original distribution of the p.p. conditional distribution given a point at another location x should be somehow related to P^0. Indeed, using formulae (2.25) and (1.11) one can prove a simple relation between P_x and P^0 . More specifically, taking $g(x, \Phi) = \chi(\Phi + x \in \Gamma)$ we obtain

$$\int_{\mathbb{R}^d} P_x\{\phi : \phi \in \Gamma\} ds = \int_{\mathbb{R}^d} P^0\{\phi : \phi + x \in \Gamma\} \tag{2.26}$$

which means that for almost all $x \in \mathbb{R}^d$ the measure P_x is the image of the measure P^0 by the mapping $\Phi \mapsto \Phi + x$ on $\mathbb{M}$. This means in simple words, that the conditional distribution of points of Φ seen from the origin given Φ has a point there is exactly the same as the conditional distribution of points of Φ seen from an arbitrary location x given Φ has a point at x. In this context, P^0 (resp. P_x) is often called the distribution of Φ seen from its *typical point* located at 0 (resp. at x). Finally, note by the Slivnyak Theorem 2.6 that for a stationary Poisson p.p. Φ, P^0 corresponds to the law of $\Phi + \delta_0$ under the original distribution.

In what follows we will often consider, besides Φ, other stochastic objects related to Φ. Then, one may be interested in the conditional distribution of these objects seen from the typical point of Φ. In these situations it is more convenient to define the Palm-Matthes (or shortly Palm) probability P^0 on the probability space where the p.p. Φ

and all other objects are assumed to be defined, rather than on (some extension of) $\mathbb{M}$ as above. Expectation with respect to P^0 will be denoted by E^0. Thus the Campbell-Matthes formula (2.25) can be rewritten as

$$\mathbf{E}\left[\int_{\mathbb{R}^d} g(x, \Phi - x)\Phi(dx)\right] = \lambda \int_{\mathbb{R}^d} \mathbf{E}^0[g(x, \Phi)]dx. \tag{2.27}$$

2.3 Statistical framework

2.3.1 Sampling mechanism

Now our focus here is on data. What we want to do is to organize the data $\{d_1, \ldots, d_m\}$ available about a given phenomenon in a way suitable for future operations. Thus, we refer to a parametric inference problem that we may summarize as follows. Assume we know that the continuation of the observations log of a given phenomenon is such that the frequency of observations showing a value less than a current x is asymptotically close to a function $F_{X_\theta}(x)$, but we do not know the value of θ. So we say that the log is described by a random variable X with Cumulative Distribution Function (CDF) $F_{X_\theta}(x)$ completely known, except for a vector parameter θ. On the basis of $\{d_1, \ldots, d_m\}$, we want to quantify θ. Moreover, since the parameter concerns an unknown future, we look for a set of θ values that we describe through a random variable (call it random parameter) Θ as well. The sole reason why we can infer Θ from $\{d_1, \ldots, d_m\}$ is that both the latter and X refer to the same variable of a given phenomenon. Hence, we rename the data as the sample $\{x_1, \ldots, x_m\}$ of X . A very suitable ancillary condition is that we may assume the x_is to be realizations of random variables $\{X_1, \ldots, X_m\}$ that have the same CDF as X and are mutually independent. With these assumptions, we may say that both the sample and its continuation – call it population with the further understanding that it is so long that the above frequency practically coincides with $F_{X_\theta}(x)$ – are generated by the same sampling mechanism as follows.

Definition 2.18. (Sampling Mechanism). With reference to the random variable X, a sampling mechanism $\mathcal{M}_X$ is a pair (Z, g_θ),where Z is a completely specified random variable and g_θ is a function that maps from realizations of Z to realizations of X. The scenario is the following. We generate a sample of X starting from a sample of Z, which figures the seeds of the former, and using the function g_θ (called explaining function) which we split into a common part g and a free parameter θ. In turn, we can determine the latter in relation to the global features of X. For the sake of generality, we assume θ to be a vector by default.

An example of a universal sampling mechanism is suggested by the integral transformation theorem [115]. It is represented by $\mathscr{M}_X = (U, \widetilde{F}_{X_\theta}^{-1})$, where U is a $[0,1]$-uniform variable and $\widetilde{F}_{X_\theta}^{-1}$ is a generalized inverse function of the CDF F_{X_θ} of the questioned random variable X. Namely $\widetilde{F}_{X_\theta}^{-1}(u) = \min\{x | F_{X_\theta}(x) \geq u\}$. The above definition reads $\widetilde{F}_{X_\theta}^{-1}(u) = (x | F_{X_\theta}(x) = u)$ for continuous X, with an obvious extension for discrete X. With this scenario, an entire observation history, made of the actual prefix – the sample – and the data we will observe in the future – the population – appears as a sequence of seeds (in its turn partitioned into those that refer to the sample and those that refer to the population) mapped onto the history through the explaining function. Actually, at this point we know almost everything about the constituting parts of the sampling mechanism except for the free parameter θ of the explaining function that, on the other end, it is crucial to being able to generate samples compatible with the phenomenon we are observing. We cannot say in advance the value of θ because we don't know the corresponding seeds associated with θ. Thus what we can do is to transfer the probability mass of the seed from the sample to the parameter value realizing the sample. The key concept of this approach is the *compatibility* of the CDF (hence of the value of θ, since g is known) with the observed sample-a measure we deal with in terms of probabilities and a random parameter Θ. The key operational tool to exploit the concept is a property of the sample to contrast with the unknown parameter which we denote as statistic $\mathbf{s}_\Theta$. [3] More formally:

Definition 2.19. (Master equation). Given a statistic $\mathbf{s}$ as a function ρ of $\{x_1, \ldots, x_m\}$ the *master equation* is the direct relation:

$$\mathbf{s} = h(\theta, z_1, \ldots, z_m) \tag{2.28}$$

between statistic and seeds deriving from $\mathbf{s} = \rho(g_\theta(z_1), \ldots, g_\theta(z_m))$, obtained by plugging the explaining function into the expression

$$\rho(x_1, \ldots, x_m)$$

In order to transfer the probability masses from seeds that generate samples to seeds that generate parameters, we go through a master equation (2.28) that denotes these masses as compatible with the observed property of a sample. The correctness of the transfer lies in the fact that each element of the Θ sample has a definite weight within the sample and the asymptotic probability masses sum to 1. These requisites are reflected in the properties of the statistic that denote it as *well-behaving* w.r.t. the parameter [12].

[3] Let us comment on the notation. $\mathbf{S}$ is a random vector representing a sample property but here he we focus on its realization $\mathbf{s}$ which we actually observe in the drawn sample. Conversely, we expect that exactly one value θ has generated this sample. However, we do not know this value and index $\mathbf{s}$ with the random parameter Θ with which we handle this lack of knowledge.

Definition 2.20. (Well-behaving statistics). Given a random variable X whose distribution law depends on a parameter θ, for a sample $\{x_1,\ldots,x_m\}$, a function $\rho(x_1,\ldots,x_m)$ is a *well-behaving statistic* w.r.t. θ if it satisfies following three properties:

1. *strongly monotonic relationship*. There must exist a unique solution of (2.28) in θ. In the case of scalar parameter θ we simply need a scalar statistic as well, where the uniqueness occurs when a uniform monotonic relation exists between s and θ for any fixed seeds $\{z_1,\ldots,z_m\}$. Otherwise the same relation will concern in a non trivial way the outputs of suitable ordering functions in the two domains $\mathfrak{S}$ and Θ where $\mathbf{S}$ and Θ, respectively span;
2. *well-definition*. On each observed $\mathbf{s}$ the statistic is well defined for every value of θ, i.e., any sample realization $\{x_1,\ldots,x_m\} \in \mathfrak{X}_m$ such that $\rho(x_1,\ldots,x_m) = \mathbf{s}$ has a probability density different from 0;
3. *local sufficiency*. Parameter Θ is the modeling counterpart of exactly the property we observe and nothing else-so that $\breve{\theta}_{\mathbf{i}}$ realizations are independent of the particular sample $\{x_1,\ldots,x_m\}$ which the $\mathbf{S}$ realization is based on. This, in turn, means that the probability of observing the sample depends on θ only through the computed s, so that the probability of the same event for a given $\mathbf{s}$ does not depend on θ – a condition that looks like a local joint sufficiency of the statistic $\mathbf{S}$ as a loose variant of the sufficient statistics introduced by Fisher [52].

Definition 2.21. (Compatible distribution). For a random variable and a sample as in 2.20, a *compatible distribution* is a distribution that has the same sampling mechanism $\mathcal{M}_X = (Z, g_\theta)$ of X with a value $\breve{\theta}$ of the parameter θ derived from a master equation (2.28) which is rooted in a well-behaving statistic $\mathbf{s}$. In turn, $\breve{\theta}$ is a θ value compatible with the sample.

Once the master equations have been set, we have two ways of actually deducing the Θ population: one numerical-which we call *population bootstrap*-and one analytical-which we have customarily referred to as the *twisting argument*.

2.3.2 Population Bootstrap

Starting from the sample $\{x_1,\ldots,x_m\}$, we compute a (possibly well-behaving) statistic $\mathbf{s}$, then we sample Θ realizations compatible with them in a number n as large as desired to compute an empirical distribution of Θ. We achieve this by extracting n samples of size m from the variable Z used for drawing seeds. Referring (2.28) to each seed $(\breve{z}_1,\ldots,\breve{z}_m)$ of the statistic, we solve the master equation in θ, so as to obtain an n-long set of candidate $\breve{\theta}$s, hence the Θ ECDF through:

$$\hat{F}_{\Theta}(\theta) = \sum_{j=1}^{n} \frac{1}{n} \chi_{(-\infty,\theta]}(\breve{\theta}_j) \tag{2.29}$$

where the extension of the characteristic function to vector instances entails $\chi_{(-\infty,\theta]}(\breve{\theta}_j) = 1$ if $-\infty < \breve{\theta}_{ji} \leq \theta_i$, for each i-th component fo the parameter vector.

2.4 Artificial Neural Network

2.4.1 Basic definitions

Actually the concept of Artificial Neural Network is widely known, thus only for completeness we will shortly give just few definitions about the the basic notions concerning artificial neural networks.

Definition 2.22. (Neural Network). An Artificial Neural Network is a graph constituted of v processing units (PE) connected by oriented arcs [4]. The graph is not necessarily fully connected, in the sense that it may lack a connection between some pairs of PE's. The PE's may be either symbolic or sub-symbolic. In case both kind of processors appear in the network, we generally speak of *hybrid neural networks*.

Definition 2.23. (Network state). The state $\tau = (\tau_1, \ldots, \tau_v)$ is the current ordered set of messages in output of each processor, where each message τ_i takes values in a set $\mathfrak{T} \subseteq \mathbb{R}$. We introduce the order to identify the messages of the single PE's, thus we refer to a *state vector*. If we want to specify a time dependence we will denote the state with $\tau(t)$, and a sub-state related to PEs' location L (capital letter) with τ^L.

Definition 2.24. (Free parameters). The free parameters $(\mathbf{w}, \theta)$ consist of the weight vector $\mathbf{w} = (w_{11}, \ldots, w_{ij}, \ldots, w_{vv}) \in D_{\mathbf{W}}$ where w_{ij} is associated to the connection from processor j to processor i ($w_{ij} = 0$ if no connection exists from j to i), and the inner parameters vector $\theta = (\theta_1, \ldots, \theta_v)$, where θ_i is associated to processor i. Depending on its use, typically according to (2.30) later on, when no ambiguity occurs we will refer to the sole vector $\mathbf{w}$ adding a dummy PE piping a signal constantly equal to 1 to the i-th PE through a connection with weight set to θ_i.

Definition 2.25. (Activation function). The activation function vector is the ordered set of functions $\mathsf{h}_i : \mathfrak{T}^v \mapsto \mathfrak{T}$ computing a new state $\tau_i = \mathsf{h}_i(\tau)$ of the i-th PE in function of the current state vector (as a component of the hypothesis h computed by the whole network).

[4] Thus the arc connecting PE i to PE j is different from the one connecting PE j to PE i.

Definition 2.26. (Activation mode). Activation mode is the synchronization order of the single processors. We may distinguish for instance between the following activation modes:

1. parallel: the PE updates its state at the same time as the other PE's synchronized with it.
2. asynchronous: the PE updates its state according to an inner clock.
3. random: the ith PE tosses a die with as many faces as there are randomly activated processors in the network. It renews its state when the die outcomes exactly i).
4. delayed: the PE updates its state a given time after the updating of the afferent PE's. In particular, instantaneous mode means a delay equal to 0.

In the case of sub-symbolic processors (the sole processors considered in this paper) the notation specifies as follows:

- PE $\rightarrow$ neuron;
- arc $\rightarrow$ connection;
- free parameters: $w_{ij} \rightarrow$ connection weight, $\theta_i \rightarrow$ threshold;
- activation function: $\mathsf{h}_i(\tau) = f(\text{net}_i(\tau))$, where

$$\text{net}_i(\tau) = \sum_{j=1}^{v} w_{ij}\tau_j + \theta_i \tag{2.30}$$

is the *net input* to PE i. Hiding the neuron index, the most common expressions of f are the following:

1. the simplest one is a *linear* function:

$$f(\text{net}) = \beta\text{net} \tag{2.31}$$

with $\beta \in \mathbb{R}$;

2. the primary nonlinear one is the *Heaviside* function:

$$f(\text{net}) = \begin{cases} 1 \text{ if net} \geq 0 \\ 0 \text{ otherwise} \end{cases} \tag{2.32}$$

which smooths in two directions described in the following two points;

3. the primary probabilistic one is described as follows:

$$\mathrm{P}(f(\text{net}) = 1) = \frac{1}{1+\mathrm{e}^{-\beta\text{net}}}; \qquad \mathrm{P}(f(\text{net}) = 0) = \frac{1}{1+\mathrm{e}^{\beta\text{net}}} \tag{2.33}$$

with $\beta \in \mathbb{R}^+$, which smooths function (2.32) in terms of random events, coinciding with the original function for $\beta = +\infty$ [5]. Hence the meaning of β is the inverse of a temperature θ of a thermodynamic process determining the value of τ;

4. the primary continuous one is the so-called *sigmoid* function:

$$f(\text{net}) = \frac{1}{1+\mathrm{e}^{-\beta \text{net}}} \tag{2.34}$$

with an analogous smoothing effect, $f(\text{net})$ in (2.34) being the expected value of $f(\text{net})$ in (2.33).

If the connections' grid does not contain a loop the graph shows an orientation. Thus we may interpret the nodes without incoming arcs as *input nodes* and those without outcoming nodes as *output nodes*. We fix the state $\tau^I = \mathbf{x}^I$ of the former, then wait for all the nodes to have updated their states after this external solicitation (we may figure this as a *propagation* of the input through the network). We consider the states of the output nodes as the output τ^O of the function h computed by the network. Training a network is the most common way of fixing their connection weights. We fix a cost function E to penalize the difference between τ^O and the value $\mathbf{x}^O$ you expect in correspondence of τ^I, sum the values of E instantiated on a series of pairs $(\mathbf{x}^I, \mathbf{x}^O)$ constituting the training set, and look for the $\mathbf{w}$ minimizing this sum. Here below we consider two instances of this procedure: i. the archetype of these algorithms aimed to train a single neuron, and ii. the *back-propagation algorithm* that we will use, with some changes and improvements, in this thesis work.

2.4.2 Perceptron

The simplest neural network is called *perceptron*, it computes an indicator function labeling with 1 the points $\mathbf{x} = \tau^I$ having a positive distance $d(\mathbf{x})$ from the hyperplane h described by a constant θ_0 and coefficients w_{0j} (being 0 the index of the output neuron) – let us call them positive points – and with 0 those having negative distance – negative points. Indeed

$$\text{net}(\tau^I) = h(\mathbf{x}) = \sum_{j=1}^{v_I} w_{0j} x_j + \theta_0 \quad \text{and} \quad d(\mathbf{x}) = \frac{h(\mathbf{x})}{\sqrt{\sum_{j=1}^{v_i} w_{0j}^2}} \tag{2.35}$$

[5] Hence $\mathrm{P}(f(\text{net}) = 1) + \mathrm{P}(f(\text{net}) = 0) = 1$.

The following theorem states that we can easily adapt the perceptron's free parameters to divide the unitary hypercube according to any hyperplane. More precisely, identifying concepts and hypotheses with both the indicator functions and the underlying hyperplanes, if a hyperplane c divides a set of points $\mathbf{x}$ into positive points and negative points but we have no record of it apart from the labels $f(c(\mathbf{x}))$ of the points, with little computational effort we can build another hyperplane h equivalent to the former.

Namely, for whatever starting values we give the parameters, we must change these values according to the following simple rule:

1. *pick a point* $\mathbf{x}$ *in the sample and compute a label* $f(h(\mathbf{x}))$ *for it according to the current values of the free parameters, where* σ *is computed according to (2.32).*
2. *If it coincides with the original label* $f(c(\mathbf{x}))$ *then do nothing.*
3. *Otherwise, if* $f(c(\mathbf{x})) = 1$ *and* $f(h(\mathbf{x})) = 0$ *(denoting too low an* $h(\mathbf{x})$*) increase* $\mathbf{w}$*,* vice versa *decrease it.*

If for the sake of uniformity we include θ_0 in the vector $\mathbf{w}$ (by adding a dummy input neuron piping a signal equal to 1 through a connection with weight exactly equal to θ_0), we obtain these changes on $\mathbf{w}$ simply by adding or subtracting the augmented input $(\mathbf{x}, 1)$ to/from the augmented parameter $(\mathbf{w}_0, \theta_0)$.

2.4.3 Gradient descent minimization method

If we have that the cost is a continuous function $s(\mathbf{w})$, then we know its minima lie in points where the first derivative with respect to every w_{ij} is 0. This occurs for instance in a fully connected network of neurons activated as in (2.34). Then, like in many NP-hard search problems, the algorithm for finding an exact solution is well defined, but its implementation might prove computationally unfeasible, for example even only the set of ν^2 equations

$$\begin{cases} \dfrac{\partial s(\mathbf{w})}{\partial w_{11}} = 0 \\ \vdots \\ \dfrac{\partial s(\mathbf{w})}{\partial w_{\nu\nu}} = 0 \end{cases} \tag{2.36}$$

is generally hard to solve analytically. Moreover, to distinguish minima from maxima, more complex conditions must be checked on the second derivatives. Therefore we generally look for incremental methods, where we move along directions in the parameter space where the derivatives of s do not increase, thus denoting a local descent of it.

This strategy has many drawbacks. The first is represented by the local minima traps. Another is that we have no guarantee in general about the running time even to reach so

poor a minimum. All depends on the length of the step we take in the descent direction: if it is too long, it could trespass the minimum; if too short, it could require an unsustainable time to get close to minimum. And then we can invent a lot of degenerate minimum situations where the point in the parameter space runs infinitely along a loop of two or more points, etc. Despite all these considerations, the gradient descent strategy is not only the one most commonly used for training a neural network, but the most successful one as well. The fact is that, in the absence of exploitable formal knowledge on the cost landscape, the more elementary the method the more robust it is.

2.4.4 The back-propagation algorithm

The most popular learning scheme in this framework is the so called *Back-Propagation* algorithm [116] that one may find in many mathematical software libraries and can easily download from the web as well. In its basic version, the method refers to a cost function represented by the quadratic error:

$$E = \sum_{i=1}^{m} l^2(\tau_i^O - \mathbf{x}_i^O) \tag{2.37}$$

where l^2 is the quadratic norm, and a neural network with a multilayer layout (usually referred to multilayer perceptron, MLP, for short) consisting of a set of ordered layers. The neurons of each layer are exclusively connected to those of the subsequent one, hence without any back or self-connection, in the absence of connections between neurons of the same layer. At the bottom we have the input layer whose neurons have states set to external values coding τ^I. These values are piped up through the various layers up to the last one (say R-th) coding the function output. At each cross of a neuron the signal is updated according to the activation rules so that the output neurons code a signal that we compare with the $\mathbf{x}^O$ associated to $\mathbf{x}^I$ in the training set and the quadratic difference between them is added in (2.37). As mentioned before, the learning method consists in the steepest descent along the E landscape in the parameter space. The basic ingredient is the computation of the derivatives of E w.r.t. the single parameters w_{ij}s and θ_is. This is obtained through the derivative chain rule rising a *back-propagation* of the error terms from output to input layer through the intermediate layers, denoted as *hidden* layers, that gives the name to the corresponding algorithms' family. Namely, the recurrent formulas read

$$\frac{\partial E(\mathbf{w})}{\partial w_{jk}} = \delta_j \tau_k, \text{ with} \tag{2.38}$$

$$\delta_j = \begin{cases} \frac{\partial E}{\partial h(\mathbf{x})_j} f'(\text{net}_j) & \text{if } j \text{ denotes an output neuron} \\ f'(\text{net}_j) \sum_{h=1}^{\nu} \delta_h w_{hj} & \text{if } j \text{ denotes a hidden neuron} \end{cases}$$

Hence we have a forward phase, where the input τ_k^I is flown till the output layer through (2.30) and (2.33) to obtain τ_k^O and the corresponding cost $e_k = (\tau_k^O - x_k^O)^2$ on the k-th output neuron, and a backward phase where e_k is piped back to the input layer trough (2.37) and (2.38).

There are many ways to use the gradient. The most elementary is to make a step in the parameters space exactly in the direction of the $E(\mathbf{w})$ steepest descent. This corresponds to updating each parameter according to

$$w_{ij} = w_{ij} - \eta \frac{\partial E(\mathbf{w})}{\partial w_{ij}} \tag{2.39}$$

It represents an algorithmic counterpart of a local independence assumption of the single weight influences. The step width η is denoted *learning rate*. New w_{ij} give rise to new costs e_k and a new run of the bcak-propagation algorithm.

2.5 BICA: Boolean Independent Component Analysis

BICA is a feature extraction procedure oriented to optimize the performances of a classification algorithm. It is framed in the Algorithmic Inference approach and takes its rationale from entropic results.

2.5.1 A peculiar clustering framework

Given a set of m objects that we identify through a set of vectorial patterns $\mathbf{y}^m = \{\mathbf{y}_1, \ldots, \mathbf{y}_m\}$, the clustering problem is to group them into n clusters $\{d_1, \ldots, d_n\}$ in a suitable way. We denote by d_i the i-th class, by Δ_i the decision of attributing a generic pattern $\mathbf{y}$ to it, and by $l(\Delta_i, d_j)$ the comparative cost (loss function) of attributing $\mathbf{y}$ to d_i in place of d_j. In this framework, we identify the problem of establishing clusters of patterns with that of maximizing the cost C of an incorrect attribution of patterns to clusters, i.e. of misclassification once we decide to actually use clusters as classes. Hence, we want to partition the pattern space $\mathfrak{Y}$ into n subsets through a decision rule

$\Delta(\mathbf{y}) : \mathfrak{Y} \to \{d_1, \ldots, d_n\}$ such that

$$\Delta = \arg\max_{\widetilde{\Delta}} \mathsf{C}(\widetilde{\Delta}) = \arg\max_{\widetilde{\Delta}} \left\{ \sum_{\mathbf{y} \in \mathbf{y}^m} \sum_{j=1}^{n} l(\widetilde{\Delta}(\mathbf{y}), d_j) \right\} \tag{2.40}$$

whose solution depends on the shape of l, i.e. we want to sharply discriminate the cluster on the basis of the loss function. For instance, for l identified with the Euclidean distance, i.e. with the l_2 metric

$$l(\Delta(\mathbf{y}), d_j) = \begin{cases} 0 & \text{if } \Delta(\mathbf{y}) = d_j \\ (\mathbf{y} - \mu_{d_j})^T (\mathbf{y} - \mu_{d_j}) & \text{otherwise} \end{cases} \tag{2.41}$$

where μ_{d_j} plays the role of representative of class d_j, then the solution is

$$\Delta(\mathbf{y}) = \arg\min_{j} \left\{ (\mathbf{y} - \mu_{d_j})^T (\mathbf{y} - \mu_{d_j}) \right\} \tag{2.42}$$

$$\mu_{d_j} = \frac{1}{\nu_j} \sum_{i=1}^{\nu_j} \mathbf{y}_{j_i} \tag{2.43}$$

with ν_j the number of objects $\mathbf{y}_{j_i}$ with index i attributed to the j-th cluster. [6]

2.5.2 The representation problem

Denoting by $H(X)$ and $H(X|Z)$ the entropy of X and the conditional entropy of X given Z respectively, for $\mathbf{Y}$ normally distributed around the representative of its cluster the above rule consists in the minimization of conditional entropy $H(\mathbf{Y}|D)$ of the data given the distribution of the clusters, rather, of its sample estimate. Indeed, by definition

$$H(\mathbf{Y}|D) = -\sum_{d_i} p_{d_i} H(\mathbf{Y}|d_i) \tag{2.44}$$

where p_{d_i} is the probability measure of cluster d_i. With this notation

[6] Note that rule (2.42) and templates (2.43) come form the conventional Bayesian approach to clustering [41] as well. Our enunciation of the problem however does not requires the existence *a priori* of *true* classes of data. Rather, they come from the use as a suitable way of organizing data, in the thread of Algorithmic Inference.

$$f_{\mathbf{Y}}(\mathbf{y}) = \sum_{i=1}^{n} p_{d_i} f_{\mathbf{Y}|D=d_j}(\mathbf{y}) = \sum_{i=1}^{n} p_{d_i} \frac{1}{(2\pi)^{n/2}} \exp\left[-(\mathbf{y}-\mu_{d_i})^T(\mathbf{y}-\mu_{d_i})\right] \tag{2.45}$$

Hence

$$H(\mathbf{Y}|D) = \sum_{d_i} p_{d_i} \mathrm{E}[(\mathbf{Y}-\mu_{d_i})^T(\mathbf{Y}-\mu_{d_i})] + a \tag{2.46}$$

where a is a constant and E denotes the expectation operator. By denoting with $I(X,Y)$ the mutual information between X and Y and re-reading in true entropic terms our clustering strategy, the general goal of any useful mapping from $\mathbf{Y}$ to D is to ensure a high mutual information [16]. Now, in the case where the mapping realizes a partition of $\mathbf{Y}$ range, hence in the case of univocal mapping, we have the following expression of the mutual information.

Lemma 2.2. *For any univocal mapping from* $\mathbf{Y}$ *to* D*,*

$$H(\mathbf{Y}|D) = H(\mathbf{Y}) - H(D) \tag{2.47}$$

$$I(\mathbf{Y},D) = H(D) \tag{2.48}$$

Claim (2.48) says that a clustering so more preserves information of patterns the more the entropy of clusters is higher, i.e. the more they are detailed. Claim (2.47) denotes the gap between entropy of patterns and entropy of clusters that is managed by the clustering algorithm. As $H(\mathbf{Y})$ is not up to us, the algorithm may decide to group the patterns so as to increase $H(D)$, in line with the claim (2.48) suggestion.

In a case where the labels of the patterns are given and no other information about them is granted, we are harmless with respect to the $H(D)$ management. Rather, our problem is to find an algorithm that reproduces the given labeling. If the task is hard, we may try to split it into two steps: i) improve the pattern representation so that their label may be more understandable, and ii) find a clustering algorithm with this new input. This corresponds to dividing the gap between $H(\mathbf{Y})$ and $H(D)$ in two steps and, ultimately, looking for an encoding $\mathbf{Z}$ of $\mathbf{Y}$ minimizing $H(\mathbf{Z}|D)$, i.e. the residual gap $H(\mathbf{Z}) - H(D)$. We have two constraints to this minimization. One is to maintain the partitioning property of the final mapping. Hence

Definition 2.27. Consider a set A of $\mathbf{Y}$ patterns, each affected by a label $d \in D$. We will say that an encoding $\mathbf{Z}$ of $\mathbf{Y}$ is correct if it never happens that two patterns of A with different labels receive the same codeword [7].

The second constraint is strategic: as we do not know the algorithm we will invent to cluster the patterns, we try to preserve almost all information that could be useful to a profitable running of the algorithm.

[7] of course we will check this property on the available patterns, with no guarantee as to any future pattern we will meet.

This goal is somehow fuzzy, since *vice versa* our final goal is to reduce the mean information of the patterns exactly to its lower bound $H(D)$. A property that is operationally proof against the mentioned fuzziness is the independence of the components of $\mathbf{Z}$. The following claim is connected to a special way of minimizing $H(\mathbf{Z})$. This is why we speak of operationally proofness. It is however a very natural way in absence of any further constraint on $\mathbf{Z}$.

Lemma 2.3. *Consider a set A of $\mathbf{Y}$ patterns and a probability distribution π over the patterns. Assume that for any mapping from $\mathbf{Y}$ to $\mathbf{Z}$ entailing an entropy $H(\mathbf{Z})$ there exists a mapping from $\mathbf{Y}$ to $\mathbf{Z}'$ with $H(\mathbf{Z}') = H(\mathbf{Z})$ such that $\mathbf{Z}'$ are independent. Then the function*

$$\widetilde{H}(\mathbf{Z}) = -\sum_k p_k \ln p_k - \sum_k (1-p_k)\ln(1-p_k) \tag{2.49}$$

with k spanning the image of A, has minima over the above mappings in $\mathbf{Z}$s having independent components.

2.5.3 The Boolean option

Finally, with the aim of lowering the vagueness of our entropic goal, we want to have Boolean $\mathbf{Z}$, as a way of forcing some reduction of data redundancy and information as well, in the direction of taking a discrete decision, ultimately, the clustering. This produces the side benefit of a concise description of both patterns and clustering formulas as a nice premise for a semantic readability of them. To this end, and limiting ourselves to a binary partition of the patterns, we assume as cost function of the single pattern s in our coding problem the following Schur-concave function [72] which we call *edge pulling* function:

$$E_s = \ln\left(\prod_{k=1}^{n} z_{s,k}^{-z_{s,k}}(1-z_{s,k})^{-(1-z_{s,k})}\right) \tag{2.50}$$

where $z_{s,k}$ is the k-th components of the encoding of the pattern s. In line with the general capability of Schur-concave functions of leading to independent component located on the boundaries of their domain [113], we may prove that minimizing over the possible encodings the sum $\breve{H}$ of the logarithm of E_s over all patterns leads us to a representation of the patterns that is binary and with independent bits, which we call BICA representation.

Lemma 2.4. *Any $\mathbf{Z}$ mapping that is correct according to Definition 2.27 and meets assumptions in Lemma 2.3 while minimizing the* edge pulling *function (2.50) is expected to produce Boolean independent components minimizing (2.44) as well.*

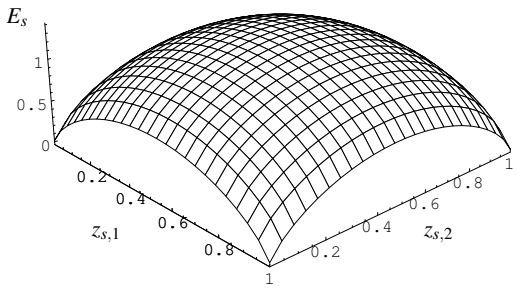


Fig. 2.1 Graph of the function E_s with $n = 2$.

Chapter 3
Capturing aggregation dynamics in space and time

We introduce our mobility model using a social community as a leading scenario. This allows us to focus on the agent intentionality as the key feature differentiating it from a particle in a gas boule. The latter is the archetype of total random community of non intelligent agents whose interaction is just uniformly ruled by the rigid body elastic collision laws. On the contrary, a social community is a crasis between social networks and virtual communities, hence a set of persons grouped together in a community because they share a common interest (the social core of the community) and need a communication network to cultivate this interest (the structural counterpart) [43].

For the members of this community we introduce a *wait and chase scheme* modeling their inter-contact times starting from the metaphor of a duel between dodgem cars. This calls for a mobility process that is stochastic but with memory at the same time, with the former component subtended by Brownian motions and the latter by the chaser capability of maintaining the correct angle of the line connecting him to the target. In this way, we add a further mobility model to the vast literature on the field, giving rise to a distribution law of the inter-contact times merging features of both negative exponential and Pareto distribution laws. Of this law we:

1. discuss the probabilistic features,
2. isolate a few free parameters and related almost sufficient statistics,
3. estimate them in order to fit experimental data,
4. provide a constructive model for simulating data,
5. regress statistical parameters into mobility physical parameters,
6. contrast results with the state-of-the-art models in the literature.

The experimental data are partly produced by ourselves through expressly featured devices within an accurate experimental campaign, and partly drawn both from specific benchmarks in the field of opportunistic networks [121], and from wider scope

databases recently collected through GPS devices within members of more or less explicit location-based social communities [112, 103].

3.1 The dynamics of a social community

Social community is the common attractor of the evolutionary trajectory of many recent ICT (Information Communication Technology) ventures, from pervasive computing [59] to ensembles of agents [136], collaborative computational intelligence [5], emerging functionalities [10], and so on. While from a static perspective, the main emphasis in social networks, such as LinkedIn [81] or Facebook [49], is placed on the relationship graph between the agents (the *role assignment* problem [79]), as a result of sophisticated studies (in terms, for instance, of linguistic/Bayesian analysis of direction-sensitive messages sent between agents [89], statistical mechanics [4], or solution of ecological graph coloring problems [19]), the time variability of these graphs is commonly dealt with in terms Dynamic Network Analysis (DNA) by combining previous approaches with multi-agent simulation techniques [27], or even through more abstract dynamical structures such as in [83].

However, new paradigms connected with the empowered telecommunication technologies, such as the mobile wireless ad hoc network (MANET) [101], stress a richer management of the communications, now under the decision of the agents, which bring back the connection timing, and the updating timing consequently, at a meso-scale that cannot be ignored [54]. It is a different approach to the network evolution, for it is rooted on the individual behavior of the single communicating members having the network as a corollary rather than the reference gear of the dynamics. In this, we expect the results of the two complementary approaches to be suitably integrated into various DNA branches [23]. Epidemic processes [22] and opportunistic networks [129, 94] are two instances of this focus on the dynamics of a social community where the scientific community has focused research efforts in recent years. The main ingredients are: a) the fact that the members of the community travel inside it; b) they have a limited infection raw, so that a message must rely on many hops from one member to another in order to cross the entire community or even a subset of it; and c) each member may manage the message transmission by himself – for instance to decide whether or not to receive a message, whether or not to forward it to another member.

The presented scenario produces a new form of proactive connectionist system whose primary engine of information exchange lies in the mobility habits of the agents. We keep our focus on the connection timing, pursuing the dry technical task of understanding the *intercontact times* distribution between members of the same community [70] – i.e. the distribution law of the random variable reckoning the elapsed times between

contacts stated within the community. The sole assumption we make is that if a member has something to transmit to others, he looks for an individual within the community who is reachable and chases him until he succeeds in transferring the message. This is the atomic motion of the community, consisting of a *wait phase* (until the member has something to transmit and someone to whom transmit it) and the above *chase phase*. Plugging these atomic motions into a stationary dynamics, we deduce the inter-contact times distribution law. This distribution has been investigated for about ten years in the frame of opportunistic networks, but interest in it dates much further back in analogous frames, such as animal foraging patterns [106] or human mobility at the basis of infectious diseases [32]. With these patterns we move from Brownian motion [46] – which is the favorite framework in physics to study inanimate particles ensemble dynamics – to Lévy flights [24, 112] – which denote a definite bias of animal patterns toward food locations. We identify this bias with the *intentionality* of the animal, say of the agent in general, and give it a constructive model as a modification of the original Brownian motion. As Palm probability theory explains [14], this leads to a timing of events that may diverge from the negative exponential distribution, still allowing for an equilibrium distribution of the mobility process, which leads to the family of Pareto distribution laws.

3.2 Processes with memory

In very essential terms, we speak of memory if we have a direction along which to order the events. Now, for any *ordered* variable T, such that events on their sorted values are of interest to us, the following *master equation* holds

$$\mathrm{P}(T > t|T > k) = \mathrm{P}(T > q|T > k)\mathrm{P}(T > t|T > q) \quad ,\forall k \leq q \leq t \tag{3.1}$$

It comes simply from the fact that in the expression of the conditional probability

$$\mathrm{P}(T > t|T > k) = \frac{\mathrm{P}(T > t)}{\mathrm{P}(T > k)} = \frac{g(t)}{g(k)} \tag{3.2}$$

we may separate the conditioned variables from the conditioning ones. While (3.1) denotes the time splitting in the fashion of the Chapmann-Kolmogorov theorem [74] as a general property of any sequence of data, equation (3.2) highlights that events $(T > t)$ and $(T > k)$ are by definition never independent. What is generally the target of the memory divide in random processes is the time $t - k$ elapsing between two events. In this perspective, the template of the memoryless phenomena descriptor is the (homogeneous) Poisson process, whose basic property is $\mathrm{P}(T > t) = \mathrm{P}(T > q)\mathrm{P}(T > t - q)$, if

$t > q$. It says that if a random event (for instance a hard disk failure) did not occur before time q and we ask what will happen within time t, we must forget this former situation (it means that the disk did not become either more robust or weaker), since your our question concerns whether or not the event will occur at a time $t - q$. Hence our true variable is $\mathtt{T} = T - q$, and the above property is satisfied by the negative exponential distribution law with

$$\mathrm{P}(T > t) = 1 - F_T(t) = \mathrm{e}^{-\lambda t} \tag{3.3}$$

for constant λ, since with this law (3.1) reads

$$\mathrm{e}^{-\lambda(t-k)} = \mathrm{e}^{-\lambda(q-k)}\mathrm{e}^{-\lambda(t-q)} \tag{3.4}$$

and the property that $\frac{g(t)}{g(k)}$ in (3.2) equals $g(t-k)$ is owned only by the exponential function.

On the contrary, we introduce a memory of the past (q-long) if we cannot separate $T - q$ from q. Here we consider very simple cases where this occurs because the time dependence is of the form $\mathtt{T} = (T/q)^{\beta}$. The simplest solution of (3.1) is represented by

$$\mathrm{P}(T > t | T > k) = \left(\frac{t}{k}\right)^{-\alpha} \tag{3.5}$$

so that the master equation reads

$$\left(\frac{t}{k}\right)^{-\alpha} = \left(\frac{t}{q}\right)^{-\alpha}\left(\frac{q}{k}\right)^{-\alpha} \tag{3.6}$$

Note that this distribution, commonly called Pareto distribution [102], is defined only for $t \geq k$, with $k > 0$ denoting the true time origin, where α identifies the distribution with the scale of its logarithm (more details will be presented in Subsection 3.2.1). The main difference w.r.t. the negative exponential distribution is highlighted by the LogLogPlots of $\overline{F}_T$ in Fig. 3.1: a line segment with a Pareto curve (see picture (a)) in contrast to a more than linearly decreasing curve with the exponential distribution (Fig. 3.1(b)).

The difference between the graphs in Fig. 3.1 shows that, for a same mean value of the variable, we may expect this occurrence in a more delayed time if we maintain memory of it as a target to be achieved, rather than if we rely on chance.

Let us introduce a local time $\tilde{t}$ as a companion of the universal time t measured by any standard clock system. Locality stands for the specificity of the time features that are relevant for a given phenomenon. In the exponential model the two time scales coincide (with the sole exception of the two time origins). In the Pareto model the local time is measured in multiples ($\frac{t}{k}$ in (3.5)) of the last time you know the questioned event has not yet occurred. Moreover, you pass from a Pareto T to an exponential variable Θ just by reckoning the time in logarithmic scale rather than linear scale, which means that

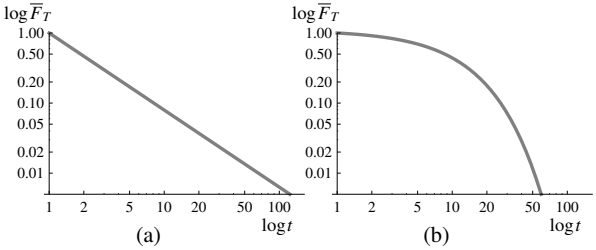


Fig. 3.1 CCDF LogLogPlot when T follows: (a) a Pareto law with $\alpha = 1.1$ and $k = 1$; (b) a negative exponential law with $\lambda = 0.09$. Parameters were chosen to have the same mean.

$\Theta = \log T$, so that

$$\mathrm{P}(\Theta > \theta) = \mathrm{P}(T > \mathrm{e}^{\theta}) = \left(\frac{\mathrm{e}^{\theta}}{k}\right)^{-\alpha} = \mathrm{e}^{-\alpha(\theta-\theta_0)} \tag{3.7}$$

with $\theta_0 = \log k$, inviting us to read θ in terms of the logarithm of a corresponding linear time t. Hence the homogeneity of the process, in the sense of a constant rate with time in (3.4), is just a matter of considering a suitable time metering. In addition, on the one hand the time homogeneity of exponential distribution gives rise to a stationary counting process ruled by a stationary Poisson distribution.

3.2.1 The Pareto family

The Pareto distribution is named after an Italian-born Swiss professor of economics, Vilfredo Pareto (1848-1923). Pareto's law, as formulated by him (1897), dealt with the distribution of income over a population and can be stated as follows:

$$N = kx^{-\alpha} \tag{3.8}$$

where N is the number of persons having income greater that x, and k, α are parameters: α is the *shape* parameter and k is the *scale* parameter. During years the Pareto distribution has maintained its original form

$$\overline{F}_X(x) = P(X \geq x) = \left(\frac{k}{x}\right)^{\alpha}, \quad k > 0, \alpha > 0, x \geq k \tag{3.9}$$

where $\overline{F}_X(x)$ is the complementary cumulative distribution function and represents the probability that the income is equal to greater than x and k represents some minimum income. Then we can write the CDF of X as

$$F_X(x) = 1 - \left(\frac{k}{x}\right)^{\alpha}, \quad k > 0, \alpha > 0, x \geq k \tag{3.10}$$

and the PDF

$$f_X(x) = \alpha k^{\alpha} x^{-(\alpha+1)}, \quad k > 0, \alpha > 0, x \geq k \tag{3.11}$$

The (3.10) is known as *Pareto distribution of the first kind*. Two other forms of this distributions were proposed by Pareto. The first one, now referred to as *Pareto distribution of the second kind* (also called Lomax distribution) has the form

$$F_X(x) = 1 - \left(\frac{C}{(x+C)}\right)^{\alpha}, \quad x \geq 0 \tag{3.12}$$

This was used by Lomax in 1954 for the analysis of the business failure data. The Pareto of type II in its standard form has $C = 1$ so that its PDF reads:

$$f_X(x) = \alpha(1+x)^{-\alpha-1}, \quad x > 0, \alpha > 0 \tag{3.13}$$

and the survival function is

$$\overline{F}_X(x) = (1+x)^{-\alpha}, \quad x > 0, \alpha > 0 \tag{3.14}$$

The third distribution proposed by Pareto - the *Pareto distribution of the third kind* - has the CDF:

$$F_X(x) = 1 - \frac{Ce^{-bx}}{(x+C)^{\alpha}}, \quad x > 0 \tag{3.15}$$

The original form and its variants found many rationales during the years. For Instance, in his work, Harris [60] has pointed out that a mixture of exponential distributions, with parameter θ^{-1} having a Gamma distribution, and with origin at zero, gives rise to a Pareto distribution. Indeed if

$$P(X \leq x) = 1 - e^{-x/\theta} \tag{3.16}$$

with $\mu = \theta^{-1}$, then

$$
\begin{aligned}
P(X \leq x) &= \frac{1}{\beta^{\alpha}\Gamma(\alpha)} \int_0^{\infty} t^{\alpha-1} e^{-t/\beta}(1-e^{-tx})dt \\
&= 1 - \frac{1}{\beta^{\alpha}\Gamma(\alpha)} \int_0^{\infty} t^{\alpha-1} e^{-t(x+\beta^{-1})}dt \\
&= 1-(\beta x+1)^{-\alpha}
\end{aligned}
\tag{3.17}
$$

More in general the Pareto family is a wide scope family that proves capable of describing many phenomena in the field of social communities, from various internet communication aspects (such as node outdegrees [64], byte flow per request [135], number of crawled links [25] and so on) to traditional communication volumes in terms of telephone calls [1, 2], physical travels of community members [125], or citations of scientific papers [39], etc., taking theoretical roots on various paradigms, such as *rich gets richer* i.e. preferential attachment [137, 84], transmission entropy minimization [86], log-returns [85, 70], phase transitions and self-organized criticality [40], etc. For short, in our perspective, facing two random times, distributed respectively according to a negative exponential and to a Pareto distribution, if they have the same average, the latter has a longer tail denoting the possibility of having a contact even after a long period. We may interpret this persistency as an expression of the agent intentionality, in turn viewed as a consequence of the fact that the underlying random process has a memory. Said in other words, it is easy to recognize that we move from exponentially distributed times – denoting intervals between sorted points uniformly drawn in a large segment – to Pareto times just by increasing at any exponential rate the scale of these intervals, like in (3.7). On the one hand this time expansion captures the heavy tailed feature of mobility processes that *run faster* than diffusive ones [90] (often associated to Lévy flights [123]). On the other hand, it denotes in our vision an intrinsic memory of these processes.

While in the rest of this thesis we will concentrate on the *Pareto type II* distribution, in the next section we will investigate from an analytical perspective the above time stretching which characterize the Pareto type I distribution.

3.3 Reasoning about scalable processes

Until now we considered the processes with and without memory as separated and very different entities. Actually they are, but we are interested in understanding more about their connection and more precisely their generative procedure, in the aim of finding relevant statistics for a deep inference of their sampling mechanisms.

The generative model

Let $(0,a)$ be a segment on the real axis $\mathfrak{u}$ and $\{u_1,\dots,u_m\}$ the sorted coordinates of points uniformly randomly shot on it. The random variable $T_i/a = (U_{i+1}-U_i)/a$ is distributed as a Beta distribution with parameters $\alpha = 1, \beta = m$. In the limit $a/m \to 0$, T_i follows a negative exponential distribution law. Namely

$$f_{T_i}(t) = \lambda e^{-\lambda t} I_{[0,\infty)}; \quad \forall i \tag{3.18}$$

with $\lambda = m/a$. Note that:

- under the above limit hypothesis, we may reformulate the joint distribution of (U_i, U_{i+1}) as follows:

$$f_{U_i,U_{i+1}}(u_i,u_{i+1}) = f_{U,T}(u_i,u_{i+1}-u_i) = f_U(u_i)f_T(u_{i+1}-u_i) \tag{3.19}$$

- for $Y = \exp(T)$ we have

$$F_Y(t) = P(T < \log t) = 1 - \exp(-\lambda \log t) = 1 - \left(\frac{1}{t}\right)^{\lambda} \tag{3.20}$$

  Hence we pass from a process without memory whose times are distributed according to a negative exponential distribution of parameter λ to one with memory whose times are distributed according to a Pareto distribution of parameters $1, \lambda$ just after a change of scales.

It is a notably change of scale the one considered in the previous point, which largely extend the time elapsed between two events of our point process. We will investigate on it in the next subsection in order to capture its property and the one of similar processes.

Sampling mechanism

As mentioned in Section 2.3.1 the seed of a sampling mechanism may be anyone, provided its distribution law is completely known. Here we focus on $U_{i+1} - U_i$ (and possibly U_i as well) as the seed of the random variable T_i according to various explaining functions. Let us start with the following equalities which partly hold only approximately for small $u_{i+1} - u_i$.

$$(u_{i+1} - u_i) \approx \exp(u_{i+1} - u_i) - 1 = \exp(u_{i+1} - u_i) - \exp(0) \quad (3.21)$$

$$(u_{i+1} - u_i) \approx \exp\left(\int_{s=u_i}^{u_{i+1}} ds\right) - 1 = \int_{s=0}^{u_{i+1}-u_i} \exp(s) ds \quad (3.22)$$

$$t_i \approx \prod_{i=1}^{n_{t_i}} \exp(\Delta_s) - 1 \approx \sum_{s=0}^{n_{t_i}} \prod_{i=1}^{n_s} \exp(\Delta_s) \quad (3.23)$$

The former one roots on the first terms of the Taylor expansion of e^x, namely:

$$e^x \approx 1 + x \quad (3.24)$$

The second line translates these results in terms of local operations done on the infinitesimal segments ds joining u_i to u_{i+1}. The latter rewrite them in terms of finite differences Δs.
Remarks:

1. Equation (3.23) highlights a non uniform process as the basis of the random process filling the segment (u_i, u_{i+1}) of terms that are progressively decreasing. This is an interpretative key of our sampling mechanism.
2. Equation (3.24) denotes that if T follows and exponential distribution, $T+1$ follows a Pareto distribution of parameters $1, \lambda$. Actually $T+1 \approx \exp(T)$, as seen in (3.20). The approximation holds for $a/m \to 0$. But the shape of the distribution is not dependent on a liner scale, so that this remains true for any a provided we adopt the correct parameters of the Pareto distribution.
3. With a small linear scale factor c, such that $c(U_{i+1} - U_i) \ll 1$ so that (3.24) still holds, we have :

$$ct \approx \exp[ct] - 1 = \exp\left(\int_{s=u_i}^{u_{i+1}} c\, ds\right) - 1 = c \int_{s=0}^{u_{i+1}-u_i} \exp(cs) ds \quad (3.25)$$

$$t_i = \prod_{i=1}^{n_{t_i}} \exp(c\Delta_s) - 1 = \sum_{s=0}^{n_{t_i}} \prod_{i=1}^{n_s} \exp(c\Delta_s) \quad (3.26)$$

Since $c\Delta_s \ll 1$, we remain with an exponential distribution, namely:

$$F_T(t) = 1 - \exp\left(\frac{ca}{m} t\right) \quad (3.27)$$

4. If we increase c in (3.25), so that $c\Delta_s > 1$, or even we adopt a non linear scale, i.e with a scale factor $c(s)$ increasing with s, then (3.27) does not hold longer. An interesting family of $c(s)$ is represented by $c(s) = \log[h(s)]$ with $h(s)$ moderately increasing with s. Reasoning with finite elements we have:

$$t_i = \prod_{i=1}^{n_{t_i}} h(s)^{\Delta_s} - 1 = \sum_{s=0}^{n_{t_i}} \prod_{i=1}^{n_s} h(s)^{\Delta_s} \tag{3.28}$$

And passing to the infinitesimal calculus

$$\exp\left(\int_{s=u_i}^{u_{i+1}} \log(h(s))\, ds\right) - 1 = \int_{s=0}^{u_{i+1}-u_i} \exp\left(s \log(h(s))\right)\, ds \tag{3.29}$$

So that, in particular:

- with $h(s) = s/\log(s)$ – rather with its extension

$$h(s) = \begin{cases} 1 & \text{if } s < e \\ \log\left(\frac{s}{\log(s)}\right) & \text{otherwise} \end{cases}$$

  it results:
  – the sampling mechanism

$$t_i = \exp\left(\int_{s=u_i}^{u_{i+1}} \log(h(s))\, ds\right) - 1 \tag{3.30}$$

  does no longer generates a negative exponential variable. Meanwhile,
  – the sampling mechanism

$$t_i = \exp\left(\int_{s=u_i}^{u_{i+1}} \log(h(s))\, ds\right) \tag{3.31}$$

  generates a Pareto variable.
- Idem for large $h(s) = c$, for instance with $c = 10$.
- With still more fast increasing functions, for instance $h(t) = t^{0.5}$, both sampling mechanisms give rise to a Pareto distribution, denoting that the offset -1 becomes inessential.

3.3.1 The sampling mechanism benefits

On the one side, we easily estimate the parameters of a Pareto distribution on the basis of the joint sufficient statistics $\min_i t_i$ and $\sum_i \log(t_i)$. A more challenging task is to entering the sampling mechanism in order to estimate for instance the parameters (c, α) as well of $h(s) = c$ and $h(s) = s^\alpha$, respectively. The aim is to get these estimates on the basis of simple statistics again, possibly regardless of the sequence of observed times. Actually, exploiting the Palm calculus we can change the reference frame of t from the generic u_i to the fixed origin 0:

Proposition 3.1. $\mathbf{E}[W(t)] = \mathbf{E}^0\left[\int_{s=0}^{T} W(s)ds\right]$ *when t is computed from the origin. The notation on left hand side of the equation represents the mean $E[W(s)]$ cumulated on all the trajectories of values of $U_i < s < U_{i+1}$, averaged on T and normalized by $1/\lambda$.*

The stationarity requires $W(t) = W(t - u_i)$, in this way we can figure all traits (u_i, u_{i+1}) to be dragged back to 0 so becoming $(0, u_{i+1} - u_i)$. This is both the rational of the above integral and the way of computing $\mathbf{E}[W(t)] = \mathbf{E}[W(0)]$ from a sample coming from the above mechanism. Thanks to the ergodicity of the process we can

- either integrate $W(t)$ along the whole segment $(0, a)$ and divide by a,
- or integrate $W(t)$ separately along the different traits (u_i, u_{i+1}), then take the sample mean of these values, and finally divide by the sample mean of the traits length.

This statement finds a companion on the following equality that is true for any c

$$\exp\left(\int_{s=u_i}^{u_{i+1}} c\, ds\right) - 1 = c\int_{s=0}^{u_{i+1}-u_i} \exp(c\, s)\; ds \tag{3.32}$$

We look for analogous formulas also for non constant $h(s)$. For instance, we are looking for β mediating the $h(s)$ values, such that:

$$t_i = \exp\left(\int_{s=u_i}^{u_{i+1}} \log(h(s))ds/h(u_i)\right) - 1 \approx \alpha\int_{s=0}^{u_{i+1}-u_i} \exp(s\log(h(s))^\alpha)\, ds \tag{3.33}$$

In this way we try to pair a non homogeneously scaled process (we normalize the integral in the exponent of the first member by a function of one of the extremes), on the one side, with a function of the sole observed times, on the other, in view of discovering parameters of the original process.

The matter of this section is still an open problem. Rather, in the next section we work with a specific generative model which proved very suitable in many operational fields.

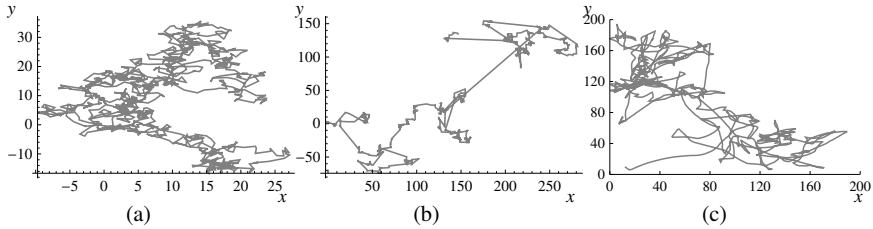


Fig. 3.2 Trajectories described by: (a) Brownian motion, (b) Lévy flights, and (c) the proposed mobility model.

3.4 A Model to maintain memory in a time process

Brownian motion has for long time been the favorite process to describe ensembles of particles within a system in equilibrium [93], for instance a gas in a cylinder, or even a crowd at an expo. In the second case we consider people behaving no differently from inanimate particles: each individual moves along a series of steps, giving each of them a constant size and no preference re the direction options, so that the symmetry of the starting scenario is maintained during its evolution (see Fig. 3.2(a)). On the contrary, a distinguishing feature of animate agents within a well structured ensemble – such as a social community – is the causality of their actions, say their intentionality. In turn, intentionality introduces local asymmetries in the motion that have been variously studied. Following [71], we may extend the reproducibility property of the Brownian motion – so that, in analogy with fractal systems [87], the large scale dynamics reproduces the low scale one, in that both are ruled by Gaussian distribution laws – using Lévy flights [56] in place of random walks [68] as elementary traits of the motion. The theoretical framework of this motion is described in Chapter 2, Section 2.1. A typical picture of a sequence of Lévy flights is shown in Fig. 3.2(b). It denotes trajectories that may describe the foraging patterns of animals such as monkeys [106], albatrosses [44], and targeted human walking as well [112]: namely, local stays in place (to eat and rest) plus sudden jumps here and there (to chase the food). The drawback of this dynamics is its high dispersion, so that the central limit theorem does not hold in its classical form (hence we do not come to a Gaussian distribution in spite of the fact that we are tracking the sum of equally distributed movements). Rather, while the scaling of the Brownian motion is $\sqrt{t}$ (since the variance is proportional to the elapsed time t), Lévy flights (with concentration parameter $\alpha < 2$) have no scaling factor with time as a consequence that the single step has no finite variance [30].

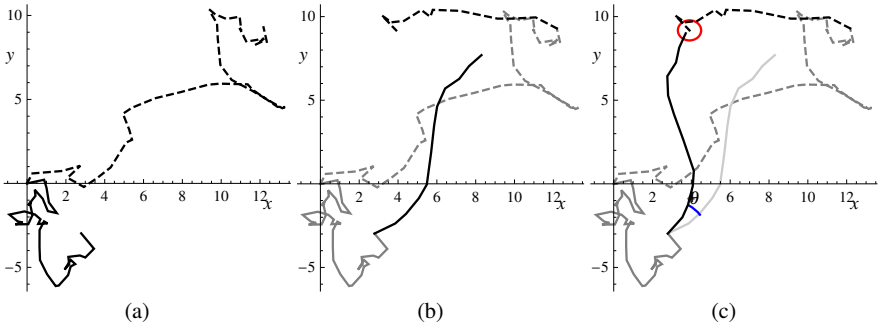


Fig. 3.3 Joint traces of two cars (plain and dashed curves respectively) when: (a) both move according to a Brownian motion behavior; (b) the former moves only in one quadrant (absolute value of the Brownian motion components) from a trigger time on; and (c) an oracle rotates this trajectory toward the other car with some approximation (quantified by the ray of a proximity circle).

Here we present a model having the same two-phase pattern as animal foraging, but with a greater prominence of local stays in place and therefore a less dispersed general behavior (see Fig. 3.2(c)), at some expense of the reproducibility property. To introduce it, think of the dodgem cars at an amusement park.

Assume you are playing with dodgem cars. You drive around until, from time to time, you decide to bang into a given car which is unaware of your intent. For the sake of simplicity, we may assume the trajectory of each car to be a plane Brownian motion before the chase triggering. Thus, with the reference frame in Fig. 3.3(a), indexing with $i = 1,2$ the cars whose stories we are following, we have

$$X_i(t) \sim \mathscr{N}_{0,\sqrt{t}} \qquad Y_i(t) \sim \mathscr{N}_{0,\sqrt{t}} \tag{3.34}$$

where $\mathscr{N}_{\mu,\sigma}$ is a Gaussian variable of mean μ and standard deviation σ. Then you, sitting in the first car, decide at time w to reach and crash into the second car. The questioned variable records the instant $T > w$ when you succeed. In the case study, where cars are points in the plane, in order to identify this instant we must specify: i) an operational definition of the cars' clash since the probability of exact matching is 0, and ii) the symmetry break introduced by the chase intention. The chase effectiveness depends on the ability to orient your motion in the direction of the target, which corresponds to converting a part of the motion along the cars' connecting line from symmetric to oriented moves. Mathematically, orientation corresponds to taking the absolute value of the elementary steps in that direction, so as to work with Chi distributed addends in place of Gaussian ones (see Fig. 3.3(b)).

In order to overcome analytical complications and fulfill point i) as well, we propose this simple scheme. As the difference between two Gaussian variables is a Gaussian variable too, we may use (3.34) also to describe the components of the distance Δ between the two cars before w. We just need to multiply them by $\sqrt{2}$ so as $X_\Delta(t) \sim \mathcal{N}_{0,\sqrt{2t}}$ and similarly for $Y_\Delta(t)$. Moreover, if we move to polar coordinates (r,θ) with $x = r\cos\theta$ and $y = r\sin\theta$, the density function f_Δ of Δ becomes

$$f_\Delta(r,\theta) = \frac{1}{4\pi t} r \mathrm{e}^{-\frac{r^2}{4t}} \tag{3.35}$$

which looks for the joint density function of (R,Θ), with R a Chi variable with 2 degrees of freedom scaled by a factor $\sqrt{2t}$, and Θ a variable uniformly distributed in $[0,2\pi)$ independently of R. Our assumption about the pursuit is that, with reference to the distances D_1 and D_2 of the two cars from the position of the first one at time w, you are able to maneuver Θ_1 from w on, so that when $D_1 = D_2$ also $\Theta_1 = \Theta_2$ (see Fig. 3.3(c)). As mentioned before, *per se* the probability of a match between two points representing the cars is null. Thus your task is unrealistic. However, intentionality recovers feasibility thanks to the fact that in practice it is enough that the angles are sufficiently close to entangle the two cars. The actual correspondence with the pursuit dynamics is facilitated by some free coefficients which will be embedded in the model.

With this assumption we are interested in the time t when $D_1 = D_2$. Given the continuity of the latter we may measure only a probability density with t. In other words, at any change of the sign in the difference $D_1 - D_2$ with the running of the two cars, there will correspond a matching time as a specification of a continuous variable T. Since both D_1 and D_2 scale with the square root of time, expressing their dependence on the *trigger time* w and the *pursuit time* τ, we have

$$D_1(t) = \sqrt{\tau}\chi_{2_1}; \qquad D_2(\tau) = \sqrt{2w+\tau}\chi_{2_2} \tag{3.36}$$

where χ_2 denotes a two degrees of freedom Chi variable whose density function is: $f_{\chi_2}(z) = z\mathrm{e}^{-\frac{z^2}{2}}$. Thus, after equating $D_1(\tau)$ with $D_2(\tau)$ we obtain

$$1 = \frac{D_2(\tau)}{D_1(\tau)} = \frac{\chi_{2_2}}{\chi_{2_1}} \frac{\sqrt{2w+\tau}}{\sqrt{\tau}} \tag{3.37}$$

under the condition $\chi_{2_2} \geq \chi_{2_1}$. Denoting with $\mathscr{T}$ the random variable with specifications τ and W with specifications w, this equation finds a stochastic solution in the random variable

$$V = \frac{\mathscr{T}}{W} = 2\left(\frac{\chi_{2_1}^2}{\chi_{2_2}^2} - 1\right)^{-1} \tag{3.38}$$

It follows the same distribution law of the ratio between two unconstrained Chi square variables, i.e. an F variable with parameters $(2,2)$ [52], whose cumulative distribution function (CDF) reads

$$F_V(v) = 1 - \frac{1}{1+v} I_{[0,\infty)}(v) \tag{3.39}$$

where $I_{[a,b]}(x)$ is the indicator function of x w.r.t. the interval $[a,b]$, thus being 1 for $a \leq x \leq b$, 0 otherwise.

Since $\tau = vw$, to have the car pursuit time $\mathscr{T}$, we need to have a convolution of the above distribution with the one of the trigger times. Let f_W be the probability density function (PDF) of the latter, defined in a range $(w_{\text{inf}}, w_{\text{sup}})$. Since $\tau + w = (v+1)w$, we obtain F_T, with $T = \mathscr{T} + W$, by computing

$$F_T(t) = \int_{w_{\text{inf}}}^{\min\{t, w_{\text{sup}}\}} F_V\left(\frac{t}{w} - 1\right) f_W(w) \mathrm{d}w \tag{3.40}$$

The dependence of the convolution integral extreme on t induces a tangible dependence of the final distribution on the trigger time's. Nevertheless, having in mind some experimental results we will discuss in the next sections, we look for a general shape of T distribution that is capable of recovering the mentioned dependences in a wide range of operational fields. To this aim, we first generalize the form (3.39) into

$$F_V(v) = 1 - \frac{1}{1+v^{2\alpha}} I_{[0,\infty)}(v) \tag{3.41}$$

obtained by changing $\sqrt{t}$ into t^α in (3.34). In this way we extend the scaling of the stochastic dynamics from the $\frac{1}{2}$ power – used in the Brownian motion – to a generic power α – in analogy to Lévy flights. Then, we approximate and further generalize this form through

$$F_V(v) = 1 - \frac{b+1}{b + (\frac{v}{c} + 1)^a} I_{[0,\infty)}(v) \tag{3.42}$$

whose template shape is reported in Fig. 3.4 in terms of both $F_T(t)$ in normal scale (see picture (b)), and $\overline{F}_T(t) = 1 - F_T(t)$, i.e. the complemetary cumulative distribution function (CCDF), in LogLog scale (see picture (b)) – thus representing both abscissas and ordinates in logarithmic scale. We call it a *shifted-Pareto* distribution since its typical elbow shape in the latter representation may be recovered from a Pareto distribution [96] just by shifting the time origin, i.e. through a law

$$\overline{F}_X(x) = \frac{b}{b + x^\alpha} I_{[0,\infty)}(x) \tag{3.43}$$

Though somewhat structurally different from (3.41), (3.42) coincides exactly with

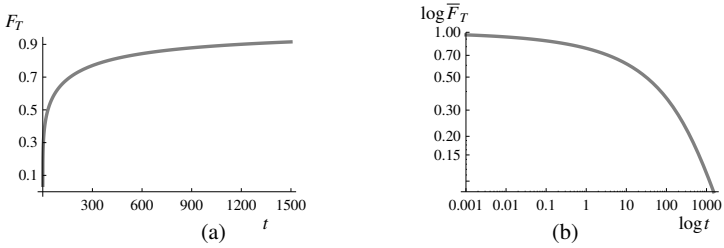


Fig. 3.4 (a) CDF Plot of a shifted-Pareto distribution, and (b) LogLogPlot representation of its complement .

(3.39) when $a = c = 1$ and $b = 0$. Indeed, the main benefit we draw from (3.42) is the gain in model generality and flexibility: thanks to the three free parameters, we may get satisfactory approximations not only of (3.41) but also of (3.40) in a wide range of operational frameworks. Actually, by plugging (3.41) or (3.42) and the chosen trigger time distribution in (3.40), we obtain expressions whose analytical form is generally not easily computable. For instance, an approximate expression of the above convolution integral for a Pareto trigger time distribution, such as

$$F_W(w) = 1 - \left(\frac{w}{k}\right)^{-\nu} I_{[k,\infty)}(w) \tag{3.44}$$

under the condition $\tau \gg w$ is:

$$F_T(t) = \frac{\nu\left(1 - \left(\frac{k}{t}\right)^{2\alpha}\right) - 2\alpha\left(\left(1 - \frac{k}{t}\right)^{\nu}\right)}{\nu - 2\alpha} \tag{3.45}$$

with an analogous power dependence on t. A better understanding of the interpolating role of (3.42) can be appreciated by referring to numerical integrations of (3.40), whose typical curves are reported in Figs. 3.5(a) to (c). They refer to different trigger time distributions – reflecting in slight changes in the length of the initial plateau, in the slope of the linear trend, and also in the smoothness of their unions – which are well recovered by the parameters of (3.42).

Thus we will refer to the random variable T as a generic success time concerning the wait and chase process, expressed either as a ratio V between chase $\mathscr{T}$ and wait time W, or as pure chase time $\mathscr{T}$, otherwise as the sum $\mathscr{T} + W$ of the two times. In any case we expect this variable to be described by the CDF (3.42) with suitable parameters.

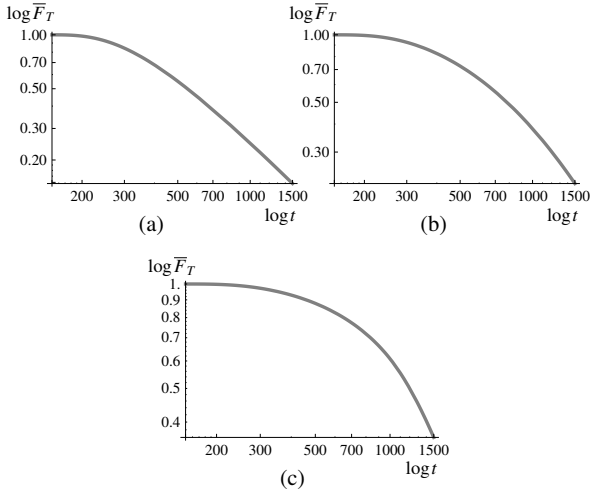


Fig. 3.5 CCDF LogLogPlot of contact times with a trigger time varying according to distribution law: (a) Pareto; (b) negative exponential; and (c) uniform. Parameter $\alpha = 0.9$ in (3.41) for all distributions, while parameters specific to the various trigger distributions are set in order to have the same expected value for the latter.

3.4.1 Completing the mobility model

A second component we must take into account to model the clash times is represented by non intentional encounters interleaving with the desired ones. By their nature, we may expect them to be biased toward short values, which may be ruled by a negative exponential distribution law. A further matter is that we are generally interested in the difference between subsequent clash times, as an instance of intercontact times in the mobility models. While this is precisely the *true* variable T we deal with through the exponential distribution of the non intentional clashes, with the chase target we have no strict theoretical results. However, basing on loose reproducibility properties deriving from both Brownian and Lévy processes ours lies between [55], we rely on the same distribution law (3.42), with proper parameters, also for these differences. We recover these aspects numerically. For instance, denoting with $\widehat{\overline{F}}_T(t)$ the empirical complementary cumulative distribution function (ECCDF) for a sample $t = \{t_1, \ldots, t_m\}$ drawn from T, i.e.

$$\widehat{\overline{F}}_T(t) = 1 - \frac{1}{m}\sum_{i=1}^{m} I_{(-\infty,t]}(t_i) \tag{3.46}$$

in Fig. 3.6 we see two ECCDFs LogLogPlot referring to a T sample obtained by drawing a trigger time uniform in $[1,150]$ and reckoning intervals between subsequent clash times whose ratio w.r.t. the former is simulated through (3.41). To these points we add further samples coming from an exponential distribution with a set parameter λ. In both figures we see a plateau analogous to those in Fig. 3.5, having the abscissa of its right end at around 100, followed by a linear slope that may be recoverd through a shifted-Pareto distribution (3.42). The effect of the exponentially drawn points is the small hump over the sloping course that variously characterizes the experimental curves with respect to their interpolations through the mentioned distribution. Thus from this and similar graphs, we may recognize a general trait of the figures that we will use henceforth, where: i) the length of the plateau plays the twofold role of order of magnitude of the mean trigger time, as for the constructive model, and of $b^{\frac{1}{a}}$, as for the interpolating law; and ii) the slope of the linear trend is close to a. To sharpen our intuition, in very broad terms we will refer to the first part of the curve as gathering almost exclusively the non intentional encounters, while the second describes almost entirely the intentional ones. On the contrary, the difference in the graphs in Fig. 3.6 is related both to the number m_e of exponentially drawn points w.r.t. the size of the sample connected to distribution (3.42), and to the respective values of parameters λ and α. As we may appreciate from the CDF course, these additional times have the effects of globally delaying the times within the plateau and of incrementing their number after its end. While in both cases $\alpha = 0.9$, the former has $\frac{m_e}{m} = 0.5$ and $\lambda = 0.1$, the latter $\frac{m_e}{m} = 0.4$ and $\lambda = 0.0001$. Shifted-Pareto parameters read $a = 1.85, b = 11,819$ and $c = 1.27$ in the first settings, and $a = 3.61, b = 7.55 \times 10^{10}$ and $c = 0.85$ in the second one.

3.5 Real world benchmarks and artificial datasets

While the paradigms of social networks and virtual communities date to over forty years ago [105, 95, 133, 134], social communities, in the dynamics connotation we explained in our introduction, represent a social and technological phenomenon whose features are still in progress [127, 43, 27]. We acquainted them from the narrow technological perspective of communication protocols, and adopted a loose opportunistic networks scenario [129, 94] in order to stress the communication features we expect from advanced social communities. In a strict sense, they are networks of mobile radio devices carried by community members, equipped with a short-range antenna and limited batteries yet with a topology continuously evolving as a function of mobility. With these devices, which you may imagine as a further enhancement of your mobile (as in smartphones, netbooks, etc.), you must abandon any real-time ambitions and reliability requirements. It works because you and other members want to communicate through a message that

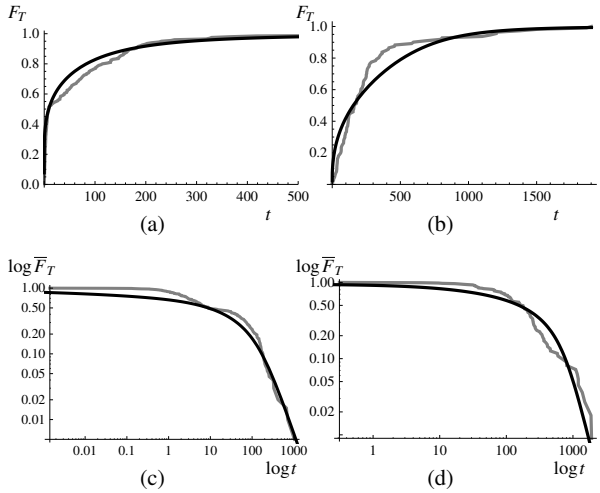


Fig. 3.6 Recovering the intercontact ECCDF shape through our mobility model. First row: CDF, second row: CCDF. Columns: different mobility parameters. Gray curves: experimental distributions; black curves: their interpolations.

reaches you because someone in the immediate past decided to send it and others to transpond it. Since the atomic action of this protocol is a contact between (at least) two members to transfer a message – a *hop* in the message trip from the origin to the destination – we study exactly the distribution of the time elapsing between one contact and the next. In a looser sense, we may consider as potential hop all occasions where two people of a same community are close enough.

Of these intercontact times we have collected a huge amount of data, partly by ourselves and partly from either public or proprietary repositories available on the web. We produced our own experimental data as well, so as both to compare ours with the former's features and to check the generality of the proposed model. In addition, we have simulated a huge dataset as an efficient *trait-d'union* between probability model and truth, with the aim of checking the appropriateness both of the former with the mobility model and of this model with the virtual community we are exploring. The essentials of the benchmarks are reported in Table 3.1, while a short description of them is reported here below.

Dataset		Location	Technology	# Agents	Mean Trace Length	# Processed Agents	Beaconing Time (sec.)
Artificial		Intel Core 2 Duo	–	39	275	39	1
PTR		Computer Science Dept. in Milan	PTR	39	1876	39	1.5
Crawdad	CH_1	Intel Research in Cambridge	iMote	9	145	9	125
	CH_2	Cambridge Univ. Computer Lab	iMote	12	342	12	125
	CH_3	IEEE INFOCOM 2005 Conf. in Miami	iMote	41	510	41	125
HMTD	Orlando	Disney World	Garmin GPS 60CSx	37	68	18	30
	NCSU	Raleigh (North Carolina) Univ. campus	Garmin GPS 60CSx	30	34	15	30
	NY	New York City	Garmin GPS 60CSx	35	25	18	30
	KAIST	Daejeon (Korea) Univ. campus	Garmin GPS 60CSx	91	308	12	30
Nokia	Helsinki	Helsinki	GPS equipped mobile	522	212	50	1
	London	London	GPS equipped mobile	199	233	50	1

Table 3.1 Description of real world and synthetic benchmarks.

3.5.1 Benchmarks from the WEB

We check our methods on two categories of benchmarks: a former expressly connected with opportunistic networks, the latter to virtual communities which could implement analogous communication protocols thanks to the contacts between their members.

3.5.1.1 The Cambridge/Haggle (CH) datasets

The Cambridge/Haggle (CH) datasets represent four of the earliest benchmarks available on the web, stored in the Crawdad database [121]. These data concern contact times between mobile agents whose companion inter-contacts, of a single agent versus the remaining ones, may be appreciated as usual through CCDF curves as in Fig. 3.7. The first dataset (CH_1) refers to eight researchers and interns working at Intel Research in Cambridge. The second (CH_2) records data from twelve doctoral students and faculty comprising a research group at the University of Cambridge Computer Lab. The third (CH_3) concerns contacts collected from 41 attendees of the IEEE INFOCOM 2005 4-day conference held in Miami. We skipped the fourth dataset because it is less homogeneous as for both the people exchanging messages and the devices carrying them. During the former experiments, very short messages are exchanged between the agents through the iMote platform [66], an embedded device equipped with ARM processor, Bluetooth radio, and flash RAM, all fed by CR2 battery. Optimized algorithms deduce both contact – the period when two agents are in range of one another – and intercontact

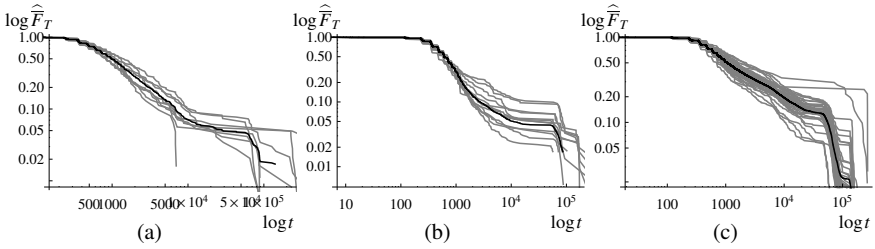


Fig. 3.7 LogLogPlot of ECCDF intercontact times for datasets CH_1 (a), CH_2 (b) and CH_3 (c). Gray curves: individual intercontacts; black curves: merge of the former.

times – the period between the contact times, when data are not directly transferrable between the two agents – through a Bluetooth base-band layer inquiry, a sort of beaconing strategy where a five seconds "enabled inquiry mode" alternates with a 120 seconds "sleep mode" to save batteries.

Here we focus on intercontact times, where the usual basic inspection tool is the ECCDF of the time log registered on a single individual getting in contact with any other of the agents participating in the experiment. We jointly visualize these curves for all devices in LogLog scale in Fig. 3.7. Namely, the time logs constitute the *traces* of the single agents in the time domain as a companion of their *trajectories* in the space domain. The corresponding ECCDF LogLog representations are the agent *tracks* from a statistical perspective. In the following we will consider them both as a curve sheaf and through single representatives. While when referring to the interpolating parameters central values such as medians will be suitable, the overall shape of the tracks looks to be better preserved by the *merge* of the curves, i.e. the ECCDF of the merged traces.

3.5.1.2 GPS trajectories

In the last few years we had a wide proliferation of personal devices, mainly smartphones, endowed with GPS facilities. This, on the one hand, stimulated the devising of many location-based services, mainly shared by members of a same social community. On the other hand, it made a plenty of mobility GPS trajectories available, as a result of current activities and a necessary basis for planning new ones. The exchange of information, and actions in general, between the involved mobile agents constitutes the core point of many of these ventures, where the exchange occurs after the encounter between agents, who may have different meeting modalities. To check extreme instances, we analyze two benchmarks where in the former – the Human Mobility Trace Dataset (HMTD) collected at the NC State University of Raileigh (NC) [111] – encounters occur

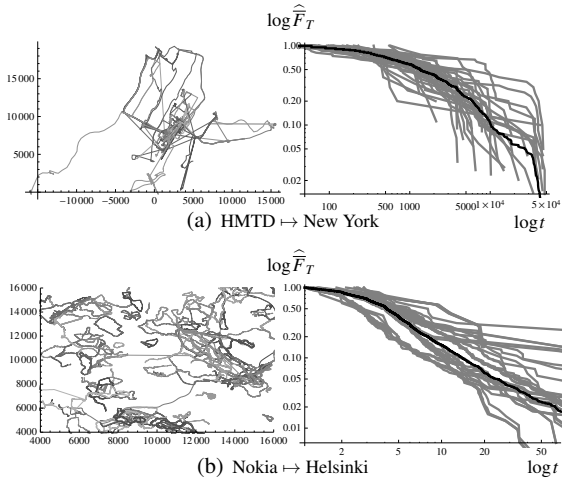


Fig. 3.8 Agent trajectories and tracks for HTMD and Nokia datasets, one per benchmark in Table 3.1. Gray ECCDF curves: individual inter-contact times; black ECCDF curves: merge of the former ones.

when people enter a relatively large interaction neighborhood re topological distances but at the same time instant. We consider the complementary situation in the second benchmark, coming from Nokia's Sports Tracker project [98]. It concerns people who virtually interact with one another because they cross the same site though at (possibly) different times.

The trajectories in the HMTD dataset, collected worldwide on various cities/ campuses through Garmin GPS devices, are at the basis of many research activities concerning human mobility modeling, routing, content search and distribution in delay tolerant networks (DTN) and ad hoc (MANET) environment [111]. From these signals, after a preprocessing phase (for instance isolating only the logs recorded within a radius of 10 km from the center of each site), and focusing on a single area where they have been collected, for instance New York City, we obtain the cartesian coordinates of the walkers position, as shown on the left in Fig. 3.8(a-b). Stating that a contact occurs when two people are less than 250 meters from each other, we obtain the companion inter-contact times ECCDF as in Fig. 3.8(a-b) on the right. In Table 3.1 we distinguish between the number of available agents for each location and the number of them processed to get statistics. This denotes that some trajectories have been discarded in that tangibly anomalous (for instance 2 or less inter-contacts) w.r.t. the phenomenon we are considering.

The second dataset is definitely larger and increases continuously. It is the follow-out of the Nokia Sports Tracker service [98] to which any person may apply by running a specific software on his own GPS-equipped smartphone. Now more than 125,000 trajectories are available, collected at a rate of one beacon per second from many regions in the world. They reduce to 9,000 circa when we focus on people walking (neither running nor cycling or anything else) and to 522 and 199, respectively in the cities of Helsinki and London, after having discarded bugged trajectories. With reference to the pic on the left in Fig. 3.8(b), we isolated 236 trajectories spanning mainly at the bottom left corner. Then we sampled variously 50 tracks re the walker we monitor and the number of other walkers (10 to 100) whose trajectories we consider to reckon contacts. We did analogously for the London tracks. In Table 3.2 we will distinguish between less crowded (sparse: fewer than 60 crossing walkers) and more crowded (dense) trajectories.

3.5.2 The ground truth

The puzzling point of the above interpretations is that all found a tangible number of authors sustaining them with theoretical and numerical arguments. Thus, in order to have a clearer perspective of the phenomenon, we decided to essentially replicate the experiment but with a finer time scale and, in any case, with perfectly known environment conditions. We achieved this by both developing portable radio devices, denoted as Pocket Traces Recorders (PTRs), and deploying the test bed [53].

Requirements

The design of the PTR has functional as well as architectural requirements. The former are related to traces collection, recording and transferring to a server station for off-line analysis. The primary focus of the PTR design is the collection of data that describe the contacts among encountering devices. The main architectural requirement is to enable experiments to last 3-4 weeks with limited human intervention as to battery changes. Consequently, we too base the communication protocol on beaconing transmission with a limited frequency. After sending its beacon, a PTR enters a sleep mode and wakes up whenever it receives a beacon from the neighborhood. Whenever a beacon is received from a given encounter, the device creates a new entry in the local *contact-log*. The beacon contains the following items: i) local ID and ID of the encounter PTR; ii) the timestamp of the first contact; and iii) the time stamp of the contact closing event. As for the latter, an entry in the contact-log is *closed* whenever no beacons are received from

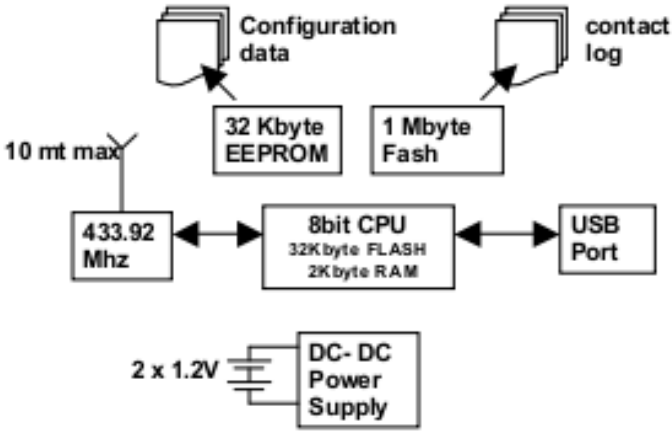


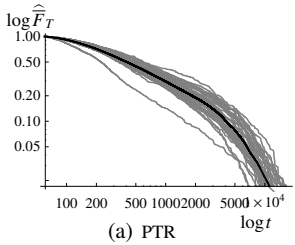
Fig. 3.9 PTR architecture.

the encounter device for more than t seconds, with $t = 60$ seconds in our experiments. The local memory size has been dimensioned to store traces of experiments lasting up to 3 weeks. As a matter of fact, our test beds have generated on average 1900 contacts per device, with beaconing time set to 1.5 seconds.

The Architecture

The overall PTR architecture is described in Fig. 3.9. It uses the Cypress CY8C29566 micro-controller and the radio module AUREL, model RTX-RTLP. The radio range has been limited to 10 meters in order both to maintain a sparse PTR distribution even in an office area and to limit power consumption. In particular, the experiments may last for more than a week with common batteries NiMh, AA $1.2V$. Each PTR is equipped with 1 MB flash memory, capable of storing more than 50.000 contacts.

The PTR firmware implements the first two layers of ISO-OSI model [73]: Manchester coding is used at the physical layer, while a CSMA non-persistent MAC protocol that regulates access to the $2400^{b}/_{s}$ channel implements the data-link layer. The local time is set at the configuration time. The clock drift in 3-week experiments has been evaluated in 10-12 seconds and therefore we have not executed any run-time clock synchronization. Each PTR uses an USB interface to communicate with the Pocket Viewer, the Desktop application software, which has been used to configure the devices, collect



(a) PTR

Fig. 3.10 Tracks from the PTR datasets. Gray ECCDF curves: individual inter-contact times; black ECCDF curves: merge of the former ones.

the recorded data at the end of the experiment, and support data analysis and device monitoring.

The collected data

The data were collected through a set of 50 PTRs circa in two experimental campaigns between February and October 2008 [11]. Each device was enabled to send and collect signals for 18 days with a couple of battery recharges. The PTRs were distributed to students and administrative / teaching staff within the Computer Science Department of the University of Milano. At the conclusion of the campaign their logs were gathered in a single server and remodulated so as to remove artifacts. In particular, we eliminated idle periods represented by the time intervals where people were expected to be far from the campus. Namely, we contracted to 0 the time intervals between 7 p.m and 8 a.m. of workdays and all during the weekend. We also clamped to 0 the last 60 seconds of contacts artificially generated by the above beaconing control rule.

After this preprocessing, we computed for each PTR a log of its inter-contact times with any other of the PTRs participating in the experiment 3.10.

3.5.3 The artificial dataset

We have numerically implemented the mobility model introduced in Section 3.4 on a simulated field. Namely, replacing the agents of the PTR campaign with dodgem cars, we consider a field of 200×200 square meters with 39 individuals uniformly located inside it as our initial configuration. Each agent has two mobility modes: random way-

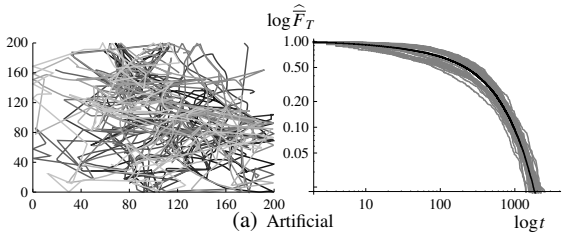


Fig. 3.11 Agent trajectories and tracks from the Artificial datasets. Gray ECCDF curves: individual intercontact times; black ECCDF curves: merge of the former ones.

point [67] up to trigger time w, and the mentioned pursuit strategy after it. In the first mode, an agent randomly selects a direction that follows for a time length θ uniformly drawn in $[0, 2000]$ steps with a mean velocity of $\overline{v} = 1.47$ meters per second (mean pedestrian velocity). This is simulated by tossing a positive random number less than or equal to 2000, as for θ, a uniform value between 0 and 2π, as for direction, and a random number drawn from a Chi distribution with 2 degrees of freedom scaled by $1.17t$ (to maintain the mentioned mean velocity), to sample the distance $D(t)$ covered by the agent at time t. At the completion of time θ, it selects a new random direction and so on. When the trigger time w expires, it shifts to the second mode: the above Chi step is now coupled with a suitable angle rotation directing the agent toward the chosen target. A match occurs when, for any reason, an agent gets closer than 10 meters to another one. We remark that we do not impose any constraint on the agent location (i.e. rebounds handling, etc.) since the chase features automatically maintain an overall attractor within the original 200×200 square. Fig. 3.11(a) reproduces a story of this model when the trigger time is drawn, for a mere choice of convenience, according to Pareto distribution (3.44) (see Fig. 3.5). In particular, dynamics parameters α (the exponent in (3.41) modulating the agent mean speed versus time in this phase) and ν (describing the rate of the trigger time distribution (3.44)) were set to 0.9 and 1.5, respectively, and suitable values were chosen for ancillary parameters, such as k or the chased target distribution. The contact times are collected along the entire experiment log corresponding to 18 days of simulated time.

3.6 Fitting the model traces

Our final goal is to propose our mobility model as an adequate tool for interpreting mobility tracks observed in some relevant social communities. To this aim, denoting with $t = \{t_1, \ldots, t_m\}$ the sample of intercontact times of a given trace, we solve two orders of inference problems:

1. fitting of t through form (3.42), by identifying *statistical parameters* a, b, c (*s-parameters*, henceforth) for each agent, with the aim of proving suitability of the proposed model to describe mobility;
2. regression of the s-parameters versus the *mobility parameters* (for short *m-parameters*), which accounts for understanding the main traits of human mobility. Here we focus on α and ν (the parameters of (3.41) and (3.44) respectively), whose acquaintance allows us to better appreciate the departure of human mobility from pure Brownian motion, as we will show in the next section. In turn, these parameters may be thought as critical ingredients in the efficient implementation of any optimized message forwarding algorithm, a task to be the subject of further research conducted by the authors.

To be most convincing, we solve these problems in two steps, in terms of: i) a reconstruction problem, by working with the artificial dataset introduced in Section 3.5.3, and ii) true inference problems over real world data described in Sections 3.5.1 and 3.5.2.

Having an efficient automatic procedure to draw the interpolating curves is a first matter. Actually, inferring a shifted-Pareto distribution is not a standard task *per se*. In addition, we must consider that, besides the hump discussed earlier, empirical data are affected by many artifacts, linked for instance to seasonal phenomena such as user habits during a particular week and/or on a particular day of the week, special tasks shared exclusively by some pairs of users, etc. Thus, rather than expecting a perfect fitting, we look for tight regions, such as confidence regions [7] where the experimental curves lie completely with a good probability. The identification of these regions is a favorite task of Algorithmic Inference paradigm [12], which we exploit here as follows.

3.6.1 The Statistical Bases

Looking at curves as in Fig. 3.11, we may consider our estimation problem in terms of drawing a regression curve through the set of pairs $\left(t_i, \widehat{\overline{F}}_T(t_i)\right)$, coupling the observed intercontact time with the ECCDF computed on it. According to our model, this regression curve depends on three s-parameters: a, b, c. Following our stated goal, we look for a suitable region containing this curve that we consider as a specification of

a random function, in principle. Thus, in analogy with the usual notion of confidence interval [114], we may define a *confidence region* as follows.

Definition 3.1. For sets $\mathfrak{X}, \mathfrak{Y}$ and a random function $C : \mathfrak{X} \mapsto \mathfrak{Y}$, denote by abuse $c \subseteq \mathfrak{D}$ the inclusion of the set $\{x, c(x); \forall x \in \mathfrak{X}\}$ in $\mathfrak{D}$. We define a confidence region at level γ to be a domain $\mathfrak{D} \subseteq \mathfrak{X} \times \mathfrak{Y}$ such that

$$\mathrm{P}(C \subseteq \mathfrak{D}) = 1 - \gamma. \tag{3.47}$$

Within the Algorithmic Inference framework [9], we infer this region via a bootstrap procedure in a slightly different version of the Efron paradigm [45]. The lead idea is that, starting from the observed data $\left\{\left(t_i, \widehat{\overline{F}}_T(t_i)\right)\right\}$, we generate a huge set of curves that *could fit* them. They represent replicas of a random curve (i.e. a curve with random parameters) at the basis of these data, where the bootstrap generation method allows us to attribute a probability to each curve whose reckoning identifies the confidence region. With this perspective we devise a procedure running through the following steps [8].

1. *Sampling mechanism.* According to Definition 2.18, the explaining function for T distributed according to (3.42) is

$$t = F_T^{-1}(u) = g_{a,b,c}(u) = c\left(\left(\frac{bu+1}{1-u}\right)^{\frac{1}{a}} - 1\right) \tag{3.48}$$

2. *Master equations.* The actual connection between the model and the observed data is exploited in terms of a set of relations between statistics on the data and unknown parameters that come as a corollary of the sampling mechanism. With these relations we may inspect the values of the parameters that could have generated a sample with the observed statistic from a particular setting of the seeds. Hence, if we draw seeds according to their known distribution – uniform in our case – we get a sample of compatible parameters in response. As mentioned in Section 2.3.1, in order to ensure this sample clean properties – so as to have clear connection between the model and the observed data – it is enough to involve sufficient statistics w.r.t. the parameters [126] in the master equations. Unluckily, because of the shift terms, the parameters are so embedded in the density function of T that we cannot identify such statistics for them. Rather we may rely on statistics that are *well behaving* with analogous benefits [9].
   Namely, denoting by $t_{(i)}$ the i-th element of the sorted intercontact times and by $\overline{m}$ the quantity $\left\lfloor \frac{(m+1)}{2} \right\rfloor$, we use the well behaving statistics

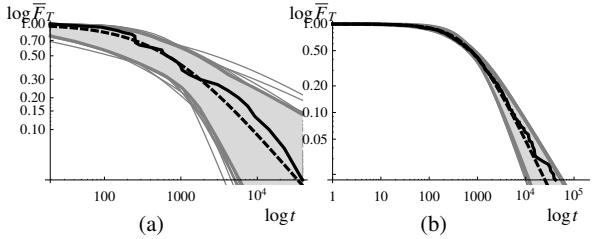


Fig. 3.12 Curves fitting with compatible parameters. (a) Sample size: 30; (b) sample size: 500. Thick plain curves: sample ECCDF; gray curves: 200 population replicas; thick dashed curves: median of the replicas. Light gray region: 0.90 confidence region.

$$s_1 = t_{(\overline{m})} \tag{3.49}$$

$$s_2 = \frac{1}{m}\sum_{i=1}^{m} t_i - s_1 \tag{3.50}$$

$$s_3 = \sum_{i=\overline{m}}^{m} \log t_{(i)} \tag{3.51}$$

Thanks to the sampling mechanism (3.48) relating a realization of the uniform random variable U to a T's, we obtain the master equations

$$s_1 = g_{a,b,c}(u_{(\overline{m})}) \tag{3.52}$$

$$s_2 = \frac{1}{m}\sum_{i=1}^{m} g_{a,b,c}(u_i) - g_{a,b,c}(u_{(\overline{m})}) \tag{3.53}$$

$$s_3 = \frac{\xi m}{2}\log c + \frac{1}{a}\sum_{i=\overline{m}}^{m} \log\left(\frac{bu_i+1}{1-u_i}\right) \tag{3.54}$$

As usual, we solve them in the s-parameters in correspondence to a large set of randomly drawn seeds $\{u_1,\ldots,u_m\}$. In this way we obtain a sample of fitting curves, as in Fig. 3.12, which we statistically interpret to be *compatible* with the observed data. The two pictures differ only in the size of the sample generated through (3.48), sharing the same s-parameters $a = 1.1$, $b = 1000$ and $c = 1.2$. The free parameter ξ is set to a value slightly greater than 1 in order to compensate the bias coming both from computing the last statistic only on a part of the observed sample, and, in the case of simulated/real tracks, from the truncation at the last intercontact, as a direct consequence of the finiteness of the campaign duration. In the figure we also report the 0.90 confidence regions for these curves. We obtain these regions through a standard

peeling method [82, 7]. We remark that in these experiments we prefer referring to the median as a central value, rather than the mean, because of the estimation method we use, as explained in [6]. But different choices may be done as a function of specific statistical goals, as will be shown in Section 3.8. The pictures highlight the tight influence of the sample size on the width of the confidence region, which in any case completely contains the whole ECCDF uniformly over its support in both situations. We also note a sort of indeterminacy – to be read as non univocity – in the triples fitting the observed ECCDF. We may attribute this both to the high variance of the sample data and to the intricate interrelations among the trend of the curves and the s-parameters per se. Namely, though the large sample allows us to infer s-parameters closer to the original ones ($\check{a} = 1.14$, $\check{b} = 1800$ and $\check{c} = 1.05$, with $\check{\theta}$ denoting the median estimate of parameter θ using the above extreme peeling procedure), with the smaller sample we have acceptable interpolation as well, despite the great difference between the inferred parameters and their true values ($\check{a} = 0.87$, $\check{b} = 320$ and $\check{c} = 0.87$).

3.6.2 Testing the inference algorithm

First of all we tested the procedure on a controlled environment represented by the artificial dataset introduced in Section 3.5.3. Fig. 3.13(a) shows the fitting curves obtained through our procedure. More in detail, from the tracks of the single agents we get the confidence region at the top of the picture. Fitting intercontact times obtained by merging individual trajectories, we get the dashed curve at the bottom of the picture which proves very close to the merge ECCDF curve.

We did not have the same success with real tracks. This motivated us to do a deeper analysis of the data described in Sections 3.5.1 and 3.5.2, suggesting that the hypothesis of their independence should be removed. While with intercontact times derived from simulation this hypothesis is true (within the limit of our ability to generate independent seeds in the sampling mechanisms ruling the simulation), with real people we may expect non independent performances. With the exception of *cold* individuals who allow no influence on their attitude by external events, most people are more reactive to both the environmental conditionings and their own personality traits. So we may expect that with busy, sociable and/or even anxious people, if a task required a quick contact with colleagues, a subsequent one will do the same with a high probability; idem for encounters that do not last very long. On the contrary, a different attitude may induce a greater duration after a rushed task and *vice versa*. In our context, this calls essentially for respectively positive and negative autocorrelation among intercontact times [37]. Far from enunciating a general theory on non independent samples, we may within our sampling

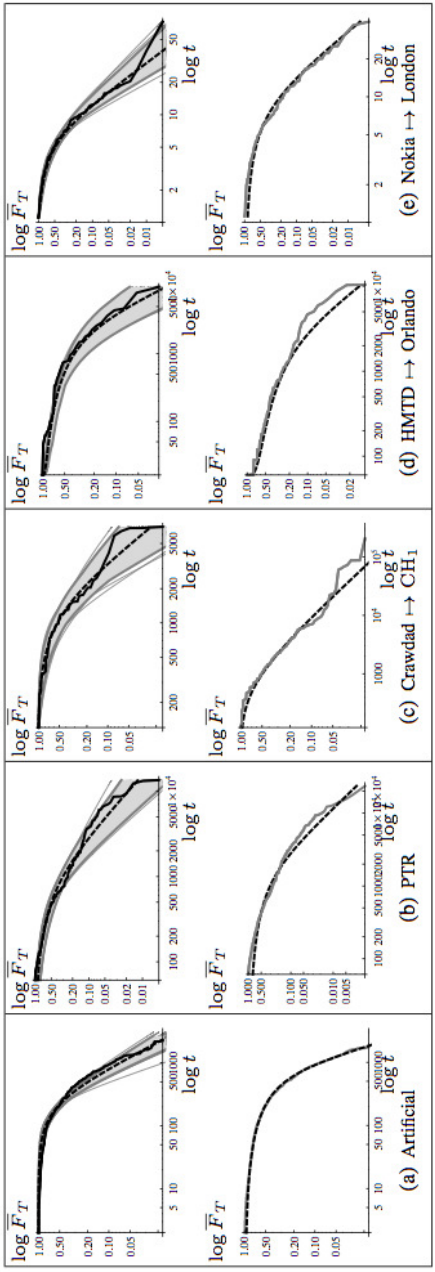


Fig. 3.13 Fitting agent tracks drawn from the dataset in Table 3.1 through our Shifted-Pareto distribution. First row → 0.90 confidence region and median curve for single agents; same notation as in Fig. 3.12. Second row→ merge track of all agents from the same dataset (gray curve) and its fit (black dashed curve).

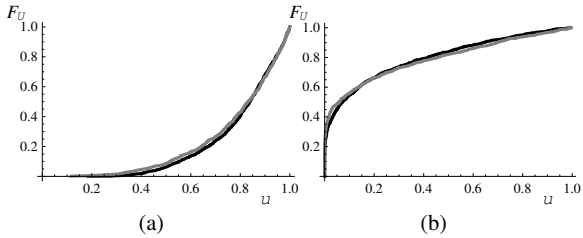


Fig. 3.14 ECDF of samples drawn according to the sampling mechanism $u_i \equiv u_i^{\left(\frac{u_{i-1}}{k}\right)^r}$ (gray curves) and $u_i \equiv u_i^d$ (black curves) when: (a) $r = -1, h = 1, d = 4, \rho_{U_i,U_{i+1}} = -0.24$; and (b) $r = 1, h = 3, d = 0.25, \rho_{U_i,U_{i+1}} = 0.37$.

mechanism reverberate this dependence directly on the observation seeds u_is entering the sample mechanism (3.48). We may look directly at a Markov process on the seeds, so that their conditional distribution depends on certain batches of previous seeds. If we are so lucky that some standard conditions are satisfied [74], we may rely on an equilibrium distribution from which to pick more suitable seeds of the sampling mechanism of the observed non independent intercontact times. Using the special typed symbol U (resp. u) to distinguish the new seed from the uniform variable U (or its realization u), we have a very preliminary hypothesis on its CDF as follows:

$$F_U(u) = u^{1/d} \tag{3.55}$$

with $d > 0$. It is definitely a gross hypothesis, relying just on some similitude between the ECDF of samples generated by the mechanism $u_i \equiv u_i^d$ (hence from the random variable U^d having exactly the CDF (3.55)) and the sampling mechanism $u_i \equiv u_i^{\left(\frac{u_{i-1}}{h}\right)^r}$ (reproducing a Markovian dependence of the current u_i from the previous one u_{i-1}), for suitable d as a function of r, with h a suitable tuning parameter (see Fig. 3.14). As for the autocorrelation $\rho_{U_i,U_{i+1}}$, the value $d = 1$ denotes independence between sample items, whereas $d < 1$ corresponds to $r > 0$ and $\rho_{U_i,U_{i+1}} > 0$, and $d > 1$ to $r < 0$ and $\rho_{U_i,U_{i+1}} < 0$.

On the one hand with known d nothing changes on the above statistical procedures, apart from the new seed generation, since the sampling mechanism now reads

$$t = c\left(\left(\frac{bu^d + 1}{1 - u^d}\right)^{\frac{1}{a}} - 1\right) \tag{3.56}$$

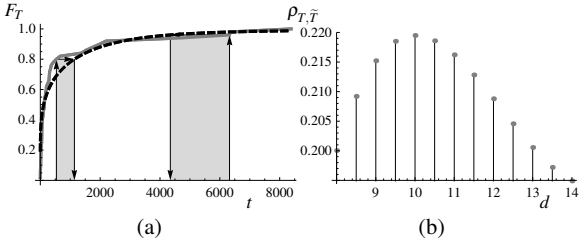


Fig. 3.15 Discovering the seed bias from a correlation rooted on reconstructed data. (a) The reconstruction mechanism. (b) Course of correlation with the biasing exponent.

leading to the CDF

$$F_T(t) = \left(1 - \frac{(b+1)}{b + \left(\frac{t}{c}+1\right)^a}\right)^{\frac{1}{d}} \tag{3.57}$$

On the other hand, adding d within the set of s-parameters would unbearably complicate the procedure. Thus we decide to use d as an external parameter that we infer by inspection with the help of the correlation $\rho_{T,\widetilde{T}}$ between actually observed times t_is and reconstructed times $\widetilde{t_i}$s. The latter are obtained by inverting the inferred CDF $F_{\widetilde{T}}$ on the ECDF $\widehat{F}_T$ specifications computed on t_is. Namely, with reference to Fig. 3.15(a),

$$\widetilde{t_i} = (t : F_{\widetilde{T}}(t) = \widehat{F}_{\widetilde{T}}(t_i)) \tag{3.58}$$

This corresponds to relating t_i to the $\frac{i}{m}$-th quantile $\widetilde{t_i}$ of $F_{\widetilde{T}}$, with $i = 1, \ldots, m$.

We choose the d maximizing the above correlation, with the further expedient of selecting a proper scale for the input data. In this respect, we experimentally found that the best indication comes from $\rho_{T,\widetilde{T}}$ when the times are recorded in logarithmic scale and possibly truncated to their minimum value (see Fig. 3.15(b)).

In this way we obtain the other pictures in Fig. 3.13, to complete the analogous ones in Fig. 3.11, yet referring to different datasets within the same benchmarks so as to widen the inspection on the data. We see that the confidence regions satisfactorily include the empirical curves, whereas the median of the compatible curves absorbs the hump commonly occurring in the right part of the curve (after the plateau). As discussed earlier, we may attribute it to a superposition of casual encounters which go in parallel to the intentional ones. While with Crawdad and HMTD benchmarks the bending of the ECCDF around the estimated median curve suggests the presence of further local phenomena to generate these balanced shifts, the more regular course of Nokia tracks may depend on the different way of collecting intercontacts. With the former benchmarks

		Single tracks				Merge track			
		$\check{a}$	$\check{b}$	$\check{c}$	$\check{d}$	$\check{a}$	$\check{b}$	$\check{c}$	$\check{d}$
Artificial		1.834 (0.215)	30584.6 (28461.6)	1.177 (0.046)	1. (0.)	2.97367	4.46×10^8	1.02844	6
PTR		1.484 (0.115)	44920.6 (39575.6)	1.097 (0.042)	2.6 (0.6)	1.752	709781	0.959	4
HMTD	Orlando	1.018 (0.185)	31.881 (27.445)	1.261 (0.142)	0.2 (0.)	2.005	2.08×10^7	0.959	8.
	NCSU	1.084 (0.262)	8336.36 (8334.31)	0.959 (0.388)	6. (2.)	1.329	151719.	1.531	6.6
	NewYork	1.099 (0.304)	10048.3 (9611.62)	1.133 (0.183)	2.6 (0.)	1.305	244650.	1.337	5.8
	KAIST	0.794 (0.17)	5.376 (4.204)	1.012 (0.21)	0.2 (0.)	1.819	1.73×10^7	0.798	11.
Nokia	$\text{Helsinki}_{\text{dense}}$	1.318 (0.324)	2.118 (1.517)	0.71 (0.067)	0.4 (0.2)	1.092	1.738	0.63	0.6
	$\text{Helsinki}_{\text{sparse}}$	1.524 (0.334)	11.45 (10.881)	0.835 (0.094)	0.6 (0.4)	1.328	0.933	0.758	0.2
	$\text{London}_{\text{dense}}$	1.883 (0.897)	32.886 (31.314)	0.863 (0.026)	1.2 (0.6)	1.646	1.985	0.829	0.2
	$\text{London}_{\text{sparse}}$	2.922 (0.217)	715.551 (318.767)	0.869 (0.014)	1.7 (0.1)	2.991	1441.7	0.858	2.4
Crawdad	CH_1	0.934 (0.166)	118.228 (98.693)	0.939 (0.128)	0.2 (0.)	0.879	106.163	0.926	0.3
	CH_2	0.849 (0.122)	41.172 (26.771)	0.762 (0.111)	0.2 (0.)	0.977	530.91	0.724	0.8
	CH_3	0.872 (0.082)	54.245 (32.943)	1.68 (0.24)	0.2 (0.)	0.813	31.488	1.725	0.2

Table 3.2 Synopsis of the parameters fitting the benchmark tracks. Cell values: single track column $\mapsto$ median and MAD (in brackets) of the estimated parameters within the dataset; merge track column $\mapsto$ parameters fitting this track.

we reckon the time differences between one contact and the next one for one agent versus another specific one, and subsequently gather the differences referring to the same agent. With the latter, for a same agent we reckon the time difference between contacts with any other agent.

As for the merge curves, we observe again a good fitting of the inferred median parameters.

3.6.3 An overall evaluation

In Table 3.2 we sum up the above inference on the benchmarks listed in Table 3.1. As mentioned before, our main statistic is the median of the parameters of the compatible curves computed for each agent. In turn, of these values we report in the first column the median and the median absolute deviation (MAD) [58] to capture, respectively, the central trend and the dispersion over the agents. In the second column we refer directly to the merge traces of the various benchmarks for which we analogously report the compatible curve median parameters. A first insight arising from an overall outlook is that with our model we cover a vast variety of targeted short-range walking situations, as for walking mode (relaxed, jogging), common goals (work, entertainment, sport, shopping, etc), geographic location (from Europe to U.S.A. to Asia) and recording method (merge of specific pair contacts, one trajectory crossing a variable number of other trajectories, merge tracks). This reverberates in a parameter variety, though with some structural commonalities.

While we will contrast a general good fitting performance with other methods in the next section, focusing on the specific features of our model, from a modeling perspective we note that the weak reproducibility property of our shifted-Pareto distribution hypothesized in Section 3.4.1 reflects in the parameters as well (with some exceptions). Indeed, taking note that the plateau parameter to be compared is $b^{\frac{1}{a}}$, we see the parameters of the merge track close to the medians of the single agent parameters enough, with the main exceptions for Orlando and KAIST, plus discrepancies on a restricted number of parameters in three other sites. There is no clear relationship between these discrepancies and the dispersion of the parameters drawn from the single agents. Rather, we note that they never occur alone, but on at least a couple of parameters per benchmark. Paired with the variability of the solutions of the inversion problem (3.52-3.54) we mentioned in Section 3.6.1 and the chimeric effects due to tracks mixing, the above fact might suggest attribute discrepancies to numerical artifacts rather than to statistical reasons. In this sense paradigmatic is the high correlation described by parameter d, which takes values much greater than 1 in the merge curve of the artificial benchmark, in spite of the true intercontact independence (by construction) in the single agent traces. The benefit of our approach, in any case, is that from a possible unconstrained number of compatible curves we may select the best fitting one according to the fitness criterion we decide, in order to for example preserve reproducibility, stress some centrality index, or follow statistical reasonings like those we will use in Section 3.8. As for the parameter dispersion, which we appreciate through MAD, we actually do not expect moderate values as an indicator of good estimates. Rather, on the one hand their high values are again a consequence of the different experimental condition each track refers to, where the almost coincidence with median for b denotes an exponential distribution law of this parameter, while other minor peculiarities emerge for the others. On the other hand, we cumulated a set of around 300 traces for a total of $140,000$ observed intercontacts circa that we will exploit as a whole in Section 3.8 to have very meaningful fitness tests. For the moment we get rid of the experiment variations through the structural coherence between data and use the different values as a cue to characterize the experiments.

From an operational perspective, in Table 3.2 we note that the slopes $\breve{a}$s have a value slightly less than 1 with the Crawdad datasets and notably greater than 1 with the others. Now, values of a outside of the interval $(0,1)$ stands for a favorable feasibility indicator. Indeed, in [29] it is shown that this interval gathers unfavorable values of the slope ν of a Pareto distribution describing intercontact times, since with these slopes none of the algorithms developed up till now for MANET routing protocols can guarantee a finite message delivery expected delay [138]. This suggests that the feasibility of an opportunistic communication protocol is not a free lunch in social communities. Rather, it may occur with a well featured mobility model in conjunction with a good practice level, like with our experiment. On the contrary, Crawdad experimental settings seem not to be destined to support these protocols, at least not at this level of investigation.

3.7 Understanding the mobility model

In the previous section we are left with a satisfactory match between experimental data and their modelization made up of three ingredients: 1) the ability of the dodgem car model to capture the main features of the agent mobility; 2) the robustness of the expression (3.42) to synthesize these dynamics even in the presence of many additional effects drifting the dynamics from the model; and 3) the adequacy of the Algorithmic Inference statistical tools to draw probabilistic scenarios compatible with the observed data, even in the presence of variations in the mobility parameters. Now we want to exploit these statistical benefits to invert the model, i.e. to deduce the m-parameters from the s-parameters (recall their definition at the beginning of Section 3.6). It accounts for a regression problem that we solve using artificial datasets like in Section 3.5.3 as training set. Namely, we kept α and ν exponents as free parameters in a range compliant with the PTR opportunistic network. In detail, we let α vary in the range $[0.35, 1.05]$, so as to appreciate sensible deviations from pure Brownian motion (i.e. $\alpha = 0.5$), and ν in $[0.1, 20]$, spanning a wide range of Pareto rates to cover both finite and non finite moments of this distribution (see rightmost picture in Fig. 3.16). Note that the d parameter is out of the question, since it is constantly equal to 1 in the artificial setup. To learn the regression function of the remaining s-parameters a, b, c versus m-parameters α and ν, first we identify the median as a template of the CCDF curves, then we regress its parameters through a polynomial in the m-parameters. In Fig. 3.16 we see the best fitting we obtain separately on a, $b^{\frac{1}{a}}$ and c. The interpretation of these curves is far from simple. As a matter of fact, with this kind of mobility phenomena we theoretically rely mainly on asymptotic results in space and time domains, and on tighter results only in the Fourier transform framework [21]. Hence, here we just venture some guesses, declaring in advance that they are partial and need serious validation. With these caveats, we note that the first graph shows a complex trend of a with α that we interpret in this way. On the one hand, the comparative inspection of curves as in Fig. 3.17 shows that an a increase ($a\uparrow$) in (3.42) has the effect of shifting the elbow between the non-Pareto and Pareto trends back (as for turning time) and down (as for the corresponding probability). This produces the twofold effect of reducing both the distribution time scale ($t\downarrow$) and the rate of contact times ($r\downarrow$) falling in the second trend (call them the intentional times according to our broad curve interpretation in Section 3.4.1). On the other hand, we see that a has a parabolic trend with α having the top in the surrounding of $\alpha \approx 0.5$, a value that calls for the Brownian motion as the basic component of the model. Moving far from this value, we see a decreasing of a that we alternatively relate to the two effects $t\downarrow$ and $r\downarrow$. Namely, since α is a mobility speed-up factor, on the lefthand side of the trend we relate the increase of a with α to a decrease in the time scale ($t\downarrow$). This effect is contrasted by the rarefaction of the random encounters when α becomes still higher, since the probability of crossing a same 10 meter raw transmitting coverage di-

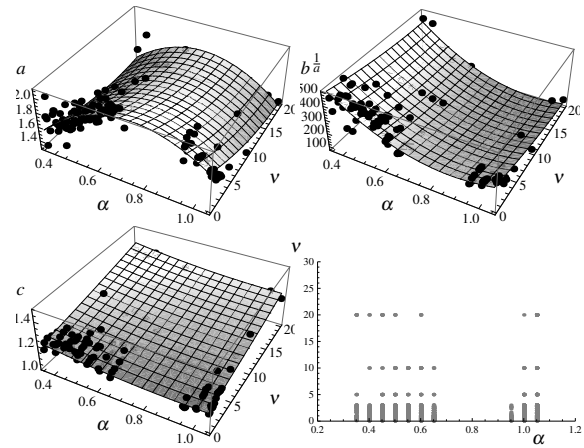


Fig. 3.16 Relation between s-parameters and m-parameters in the artificial dataset. Surfaces: best fitting curves; points: s- and m-parameters.

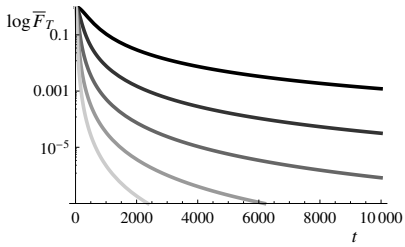


Fig. 3.17 CCDF LogPlot of shifted-Pareto distribution (3.42) with a ranging from 2 (black curve) to 3.6 (light gray curve).

minishes with the velocity due to the low agent density. Under these conditions we have an overwhelming amount of intentional contacts (belonging to the Pareto trend) ($r\uparrow$).

We may similarly explain the second graph, where we broadly relate the $b^{\frac{1}{a}}$ parameter to the scale of the non intentional encounter times. In principle, this scale decreases with ν – since the average of the related Pareto does so – and increases with α – because of the aforementioned spreading effects of this parameter. However, in this case too we have a saturation effect, so that very small νs equalize the trigger times. As a consequence, the number of (now almost purely intentional) contacts follows a Poisson distribution that is analogous to the one of the almost purely non intentional encounters reckoned in the opposite corner. Likewise, we have seen that short as in correspondence

	Simil-Pareto Best	Simil-Pareto Median	Exponential	Pareto	LogNormal	Tapered Pareto	Truncated Pareto
PTR	0.154 0.949	0.077 0.923	0.821 0.	0. 0.	0.026 0.026	0. 0.	0.077 0.
Crawdad	0.508 0.441	0.169 0.101	0.067 0.	0.051 0.	0.034 0.288	0.017 0.	0.068 0.271
HMTD	0.593 0.656	0.468 0.469	0. 0.	0.25 0.	0.468 0.	0.562 0.062	0.781 0.281
Nokia	0.937 0.531	0.796 0.281	0. 0.	0.312 0.218	0.641 0.141	0.437 0.	0.328 0.109

Table 3.3 Statistical comparison between competitor models. Rows: benchmarks; column: models; cells: CvM test acceptance rate (upper line) and Akaike criterion winning rate (lower line).

to short αs may reduce the number of non intentional encounters (since $r\uparrow$) contributing to the definition of the scale of the non Pareto trend.

The third parameter, c, looks like a fine tuning factor indirectly affected by the m-parameters.

Moving on the experimental datasets, we want to discover both the mean velocity and mean waiting time of the people who wear the beaconing device using the above regression curves. Namely, having computed s-parameter replicas compatible with the collected experimental datasets through master equations (3.52-3.54), as in Section 3.6, we look for the corresponding m-parameters α and ν that minimize the relative error between the computed a, b, c and the triple obtained through the regression curves. We depict all of them (merged re the benchmark subsets) in Fig. 3.18.

Omitting for a moment the third column, we see that the former two denote a notable generalization of the regression curves on the new points, despite their location in regions that may be far from the ones spanned by the training set. The clouds of points refer to the union of numerous curves (hence the triplets of parameters specifying them) that are compatible with the single agent tracks. For instance, we have $73,180$ curves related to the PTR benchmark, $20,000$ to the Nokia benchmark, and so on. For all these curves, on the one hand we obtain values in line with the overall trend of both a and b with α and ν, as modeled in the previous sections. On the other hand, these values are compatible with the physics of the people's dynamics and reflect the two polarizations of the dynamics (before and after $\alpha = 0.5$) discussed in the previous section.

To get these results we had to do a small preprocessing of the GPS benchmarks. Namely we had to suitably enlarge b through a scale factor that depends on the single dataset. As this parameter is a gross indicator of the mean span of the waiting phase, we are not surprised by the need for this correction in order to adapt the regression curves drawn on the ideal model to track peculiarities, such as the time difference vanishing with which people cross each other in the Nokia trajectories. However, this rescaling appears to be consistent. Still within the Nokia datasets, we must enlarge more the b values referring to dense traces rather than to sparse ones. Moreover, we cannot guarantee

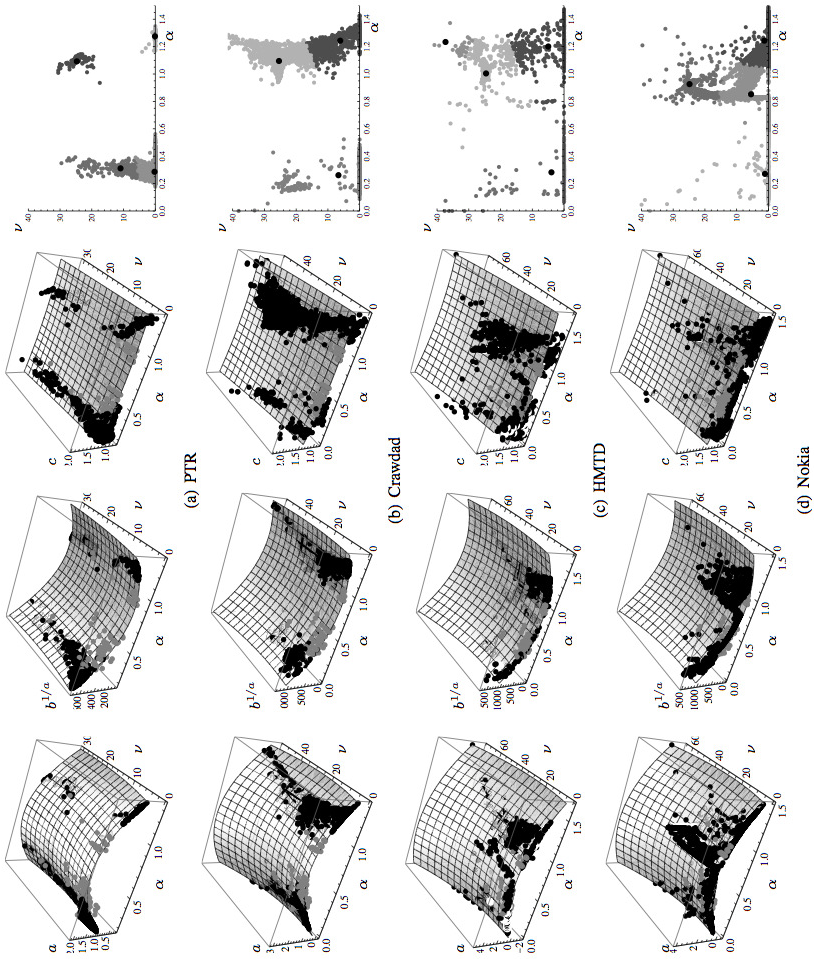


Fig. 3.18 Mobility parameters emerging from smearing the s-parameters of our experimental benchmark tracks on the surfaces in Fig. 3.16. First three columns → gray points: same simulated parameters as in Fig. 3.16; black points: replicas compatible with the processed dataset; white points: median parameters among the replicas (graphically hidden in general by the former ones). Last column → gray points: mobility parameters colored according to the cluster they belong to; bullets: cluster centroids.

trigger times to be Pareto distributed. But we have also realized a broad independence of the final distribution on the specific trigger one, so that we may appreciate the inferred ν as an indicator of the first and second moment values.

With c things go worse. But this is to be expected given the tuning role of this parameter. We note, however, that taking into account its value in back-regressing α and ν (through the minimizing procedure) diminishes the spread of these parameters.

Fig. 3.18, fourth column reinforces the hypothesis of a common phenomenon at the basis of the different community mobilities. Indeed, it highlights the great similarity between the mobility parameters underlying the different benchmarks. This comes through in the shape of the clouds gathering them and even in the location of the centroids of the clusters emerging from a naive k-means algorithm [61] computed on the whitened data [41] (to take into account the different dispersion of the mobility features).

3.8 Contrasting the literature

The common features emerging from the various tracks drawn from the different experimental setups include: a prominent elbow, separating the plateau from the slope, and a linearity of the tail. Actually, the elbow is an artifact of the CCDF LogLog representation of a vast variety of distribution laws, from uniform distribution to Gaussian, and even to exponential one. Among them, the following distributions are adopted in the literature as candidates to cope with a prominence of this feature: exponential, Pareto, lognormal, tapered Pareto, truncated Pareto. In this section we will use these as competitors of our shifted-Pareto distribution. An exception w.r.t. this feature is precisely represented by the power law, here identified with the Pareto distribution, i.e. the most simple version of the generalized power laws enunciated in the literature [124], whose graph, on the contrary, fully meets the second feature of our tracks (see Figs. 3.11 and 3.13).

The time distribution we propose captures both features (prominent elbow and linear slope) by introducing a shift in the coordinate of the power law distribution as done by Pareto himself over a century ago [102]. Then, in analogy with the 3-parameter generalized Pareto described by the formula

$$\overline{F}_T(t) = \left(1 + \frac{a(t-c)}{b}\right)^{-\frac{1}{a}} I_{[c,+\infty)}(t) \tag{3.59}$$

we introduce in (3.42) further scale parameters to render it adaptive to many experimental situations, but in a slight different way than in (3.59) in order to get more efficient statistical tools for estimating its parameters as in Section 3.6. By contrast, the random

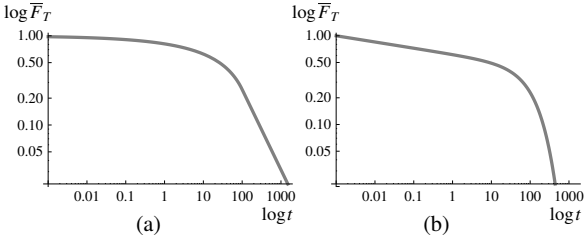


Fig. 3.19 CCDF LogLogPlot of a T randomly varying according to: (a) a double Pareto; (b) a tapered Pareto.

process we consider, is quite different from those to which the Pareto family usually applies [96], as mentioned in our introduction. Thus, the wait and chase model has no direct connection with the *rich get richer* paradigm used both in economics [110] and in web consensus phenomena [25]. The same holds for the dynamics of exceptional events in nature, such as earthquakes [104], which are far from the dynamics of our agents.

From a strictly analytical perspective, with our curves we have a course that is close to be exponential before the elbow and definitely power law after it. They cope respectively with the mentioned plateau and slope we observe in the experimental tracks. Wanting to explicitly maintain this dichotomy, other authors gave different readings of these courses, for instance in terms of: i) a *double Pareto* curve (a lower power curve followed by a greater power one) [109], or, in alternative, ii) a temporal sequencing of a Pareto trend proceeding with an exponential distribution that quickly nears 0 [29]. Now, while extensively studied in growth phenomena [91, 110], the double Pareto not only has been adapted to mobility studies with feeble results; it also misses the real course of the first part of the experimental curves (see Fig. 3.19(a)). *Vice versa*, the tapered Pareto distribution [69]

$$\overline{F}_T(t) = 1 - \left(\frac{k}{t}\right)^{\alpha} \mathrm{e}^{\frac{k-t}{b}} I_{[k,+\infty)}(t) \tag{3.60}$$

and similar ones [26, 66] recently brushed out to cope with the questioned phenomena [29] in line with alternative ii), have the exponential decrease in the rightmost part of the curve as their main drawback (see Fig. 3.19(b)). This is somehow explained in terms of nomadic motion [76], and find a good fitting only with specially devised traces [112]. As a matter of fact, researchers working on Nokia datasets also [103] lean toward an analytical description of these data through a tapered Pareto distribution, though admit that other types of mobility patterns, and consequent distributions, could equally serve their purpose.

Other authors prefer concentrating their analysis on the most populated part of the traces to gain simplicity. Thus they [26, 29] bind the analysis near the plateau, lowering the relevance of the remaining times with the twofold effect of shading off times that are exceedingly long and exceedingly costly to process re parameter estimation. Then, they analyze the surviving data according to the candidate distributions mentioned in this section, using various goodness-of-fit (GoF) tests to decide the ultimate model. Aiming to show the benefits of our model, we both repeat part of these tests and make specific theoretical considerations as a further preference argument.

We use the same statistics as in [33] and [65], i.e. the Cramer-von-Mises (CvM) test [35] and the Akaike criterion [3], respectively. Thus in Table 3.3 we index the rows with the experimental benchmarks considered, and head the columns with the candidate distributions. In each cell we report both the percentage of traces not rejected by the CvM test having the column distribution as null hypothesis, and the percentage of traces whose Akaike statistic computed on the column distribution proves to be the best, i.e. the lowest, re the other candidates. To be compliant with analogous tables in the cited papers, we used the maximum likelihood estimators of the competitor distribution laws, mentioning that their computation deserves some numerical instability when referred to the tapered distribution [69].

We remark the general superiority of our distribution, which is partly due to the recalled fact that for each trace we have concretely available a huge set of *compatible* distributions from which to select the one minimizing the test statistic. Fig. 3.20 shows this point with respect to one of these curves. Of the compatible distributions, we reported both the median one (the usual reference term of the other pictures) and the optimal one re the adopted test. From their comparison, two different inference targets clearly emerge. The median curve aims to fit the entire experimental track well with special regard to the tail, due to the special features of heavy tailed distributions on the one hand and the involved statistics on the other one. The optimal curve focuses on the mean score of the fitted points, thus tending to sacrifice the tail points at the entire benefit of the remaining bulk. This is particularly true with the CvM test, where the contribution of each point to the statistic is at most $\frac{1}{m}$, no matter how far the point is from the hypothesized track. Actually, while the Akaike criterion is always in our favor, the CvM test promotes the truncated Pareto distribution on the HMTD benchmark, and the exponential distribution on the PTR benchmark. Note that on the other two benchmarks we beat the other candidates also on the basis of the median curves. The difference between the two inference targets mainly determines the beneficial difference between ours and most alternative approaches. Actually, the various artifices to embed an elbow in the Pareto distribution, such as by multiplying it with an exponential distribution (the origin of tapered distribution), or truncating the former to its maximum observed value (the origin of the truncated Pareto), have the main effect of canceling the heavy tailed feature. As a consequence we miss the fitting of the tail observations, which neverthe-

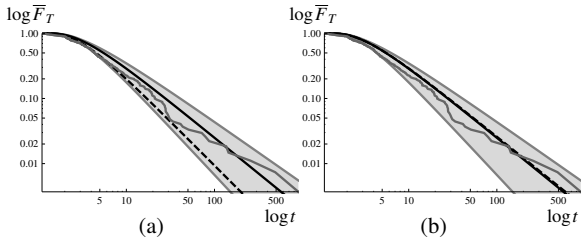


Fig. 3.20 Comparison between median curves (thick black) and the best fitting ones (dashed black) computed according to: (a) CvM test, and (b) Akaike criterion, for a Helsinki$_{\text{dense}}$ track. Same notation as in Fig. 3.13.

less constitute however almost one third of the entire sample. Thus, losing the CvM test in the mentioned benchmarks, despite we win w.r.t. the Akaike criterion, may denote the unsuitability of this test to our inference goal rather than a failure of the inference. In any case, since these statistics are distribution free, we may state that, over a sample of 303 elements, the goodness of fitting a general walker mobility through our model passes the CvM test in 81.52% of the cases and beats the competitor models on the basis of Akaike statistic 100% of the time.

Summarizing the comparison with state-of-the-art models, we can say that ours gives rise to an intercontact distribution law having a strong rationale in the dynamics of human agents within a social community and that it enjoys efficient statistical tools to infer its free parameters. These tools are so powerful that they meet different fitting criteria. We often beat competitor models as for standard GoF tests, with the additional ability to earn comparable scores even under the unprecedented constraint of preserving the heavy tailed feature of the observed data.

3.9 Concluding remarks

In view of discussing the genuine roots of the non Brownian motions we toss in this chapter the *non symmetry* features of the involved random phenomena. Reading these features in terms of *intentionality* driving the members of a social community far from a simple random walk, we focus on elementary processes where trajectories cannot be considered as the accumulation of a symmetric noise. To this aim we have introduced an extended Pareto distribution law with which we analyzed some intentional trajectories.

Actually, discovering a law ruling Nature requires managing features that are common to its application field. Without any philosophical pretension, we may identify them with properties with a certain degree of uniformity. It may refer to *invariance* of physical laws [97], uniformity of organic systems [120] or constancy of living systems' morphology [47], and so on. Under a logical perspective all of them share the minimal feature of symmetry as both a fairness guarantee and a universality prerequisite. Moving from static laws to algorithms, their correctness has been customary tossed in terms of homogeneity: the algorithm is correct since it produces a correct output for whatever input, with the understatement that an extended form of symmetry now concerns the tool for processing data and not the phenomenon generating them. Recently, the granular computation framework [9] moved a step ahead in the consideration of symmetry constraints versus benefits. Within this framework, an algorithm is satisfactory if it works well on a certain set of inputs, not necessarily coinciding with the input universe. To be useful, however, we expect that this set covers a wide family of instances that we may encounter in the real world. Now, with this strategy algorithms are tailored on data, thus producing statistics, by definition. The study of statistics, however, is tightly connected to the study of probabilistic models, in turn heavily biased by symmetry constraints as antidotes to lack of knowledge. This opens a hiatus in the above sequence toward the symmetry release. So, in this chapter, on the one side we introduce a very elementary model to explain the clean, and complex as well, phenomena connected to the bumps of a dodgem car, in the ambition of having them as a template of a vast family of intentional processes. On the other side, to empower the model, we have been pushed to fill the mentioned gap between model and statistics for its identification.

Facing a new kind of processes we found beneficiary using a new statistical framework represented by the Algorithmic Inference, getting some satisfactory results. Far from considering exhausted the matter, in the next chapter we introduce this mobility model in a very complex process implementing a subsymbolic learning procedure. Learning indeed looks us to be the archetype of the intentional processes.

Chapter 4
Information driven dynamics

One of the most relevant topic of the computer science is the computational learning, i.e. to design algorithms enabling a computer system to (almost) autonomously add a new program in its software library just after analyzing relevant data. This looks us to be the archetype and the abstraction as well of any intentional behavior. Relating the degree of autonomy to the absence of any structured knowledge transmitted by other systems, an extreme stretching of the above goal is to reach it through a sub-symbolic computational tool, say a neural network. With these premises, in this chapter we toss the augmentation of a neural network through the addition of the motion facility of its neurons. For short: *What's better than a neural network? A network of mobile neurons.* We interpret this empowering as an addition of a further intentionality expression. Hence we finalize the neuron motion to exploit a greater cognitive performance, like it happens in Nature. This is why we start this chapter with some biological recalls. Then we move to our artificial implementation, by introducing a new paradigm of neural network by coupling the well assessed evolution of the network parameters through the back-propagation algorithm with a dynamics of its neurons. They move according to Newtonian laws which are parametrized on cognitive quantities and finalized to reaching a ground state denoting a firm stability of the acquired knowledge. This entails a new training algorithm that we toss on two common benchmarks for regression and classification tasks, respectively. Besides the canonical efficiency indicator, such as convergency speed and test error, we discuss of the structural features emerging from the dynamics of this network.

4.1 A biological insight of intentionality

The macro-scale intentional behavior we caught by the use of processes with memory can be found also in the micro-scale domain. Moving for a while our discussion to biology, we can find a very similar representation of the previously discussed intentionality, though intended in a little different way. We are referring to what happens during the process of the human brain morphogenesis, here the first step in wiring the nervous system together is the generation of neurons. Consider for example the striate cortex. In the adult, there are six cortical layers, and the neurons in each of these layers have characteristic appearances and connections that distinguish striate cortex from other areas. Neuronal structure develops in three major stages: cell proliferation, cell migration, and cell differentiation. We will use the central visual system as an example of this process [15].

Cell proliferation

The brain develops from the walls of the five fluid-filled vesicles. These fluid-filled spaces remain in the adult and constitute the ventricular system. Very early in development, the walls of the vesicles consist of only two layers: the ventricular zone and the marginal zone. The *ventricular* zone lines the inside of each vesicle, and the *marginal* zone faces the overlying pia mater[1]. Within these layers of the telencephalic vesicle, a cellular ballet is performed that gives rise to all the neurons and glia of the visual cortex. The process can be summarized as follows:

1. A cell in the ventricular zone extends a process toward the pia mater that reaches the upward zone of the cortical plate.
2. The nucleus of the cell migrates upward from the ventricular surface toward the pial surface; the cell's DNA is copied.
3. The nucleus, containing two complete copies of the genetic instructions, settles back to the ventricular surface.
4. The cell retracts its arm from the pial surface.
5. The cell divides in two.

The fate of newly formed *daughter cell* depends of a number of factors. As a matter of fact, a ventricular zone precursor cell that is cleaved vertically during division has different fate than one that is cleaved horizontally. After vertical cleavage, both daughter

[1] The surface of the central nervous system is covered by three membranes called meninges: the *dura mater*, the *arachnoid membrane* and the *pia mater*. The pia mater is the innermost of these three meninges.

cells remain in the ventricular zone to divide again and again. This mode of cell division predominates early in development to expand the population of neuronal precursors. Later in development, horizontal cleavage is the rule. In this case, the daughter cell lying farthest away from the ventricular surface migrates away to take up tis position in the cortex, where it will never divide again. The other daughter remains in the ventricular zone to undergo more divisions. Ventricular zone precursor cells repeat this pattern until all the neurons and glia of the cortex have been generated (see Fig. 4.1).

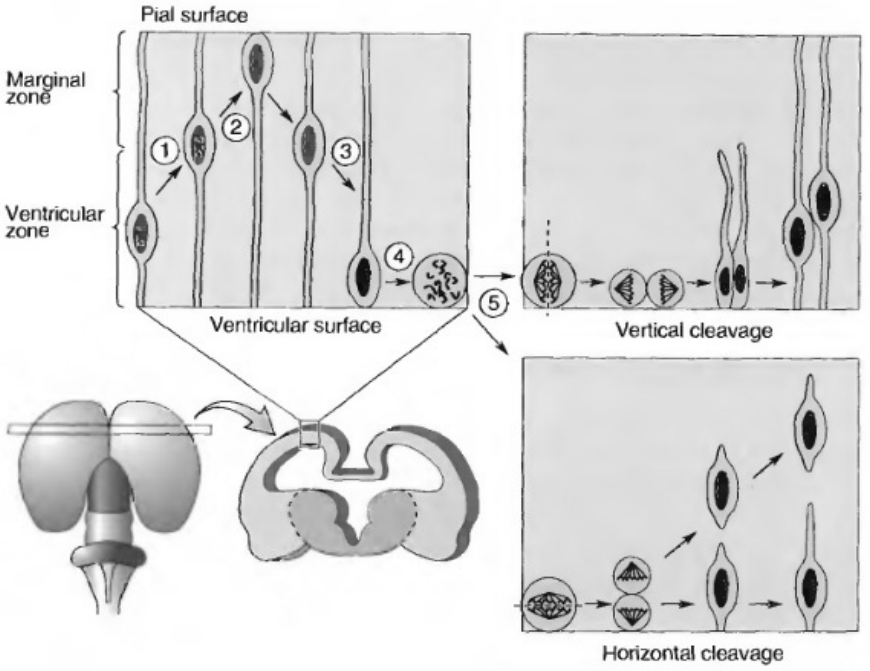


Fig. 4.1 (Up-left) The wall of the brain vesicles Initially consists of only two layers, the marginal zone and the ventricular zone. Each cell performs a characteristic ”dance” as it divides, shown here from left to right. The circled numbers correspond to the five ”positions” described in the text. The fate of the daughter cells depends On the plane of cleavage during division. (Up-right) After cleavage in the vertical plane. both daughters remain in the ventricular zone to divide again. (Down-right) After cleavage in the horizontal plane, the daughter farthest away from the ventricle ceases further division and migrates away.

In humans, the vast majority of neocortical neurons are born between the fifth week and the fifth month of gestation, peaking at the astonishing rate of 250,000 new neurons per minute. Recent findings suggest that although most of the action is over well before birth, the adult ventricular zone retains some capacity to generate new neurons. However, it is important to realize that once a daughter cell commits to a neuronal fate, it will never divide again.

Mature cortical cells can be classified as glia or neurons, and the neurons can be further classified according to the layer in which they reside, their dendritic morphology, and the neurotransmitter they use. Conceivably, this diversity could arise from different types of precursor cell in the ventricular zone. In other words, there could be one class of precursor cell that gives rise only to layer VI pyramidal cells, another that gives rise to layer V cells, and so on. However, this is not the case. Multiple cell types, including neurons and glia, can arise from the same precursor cell. Because of this potential to give rise to many different types of tissue, these precursor cells are also called *neural stem cells*. The ultimate fate of the migrating daughter cell is determined by a combination of factors, including the age of the precursor cell, its position within the ventricular zone, and its environment at the time of division. Cortical pyramidal neurons and astrocytes derive from the dorsal ventricular zone, whereas inhibitory interneurons and oligodendroglia derive from the ventral telencephalon. The first cell to migrate away from the dorsal ventricular zone are destined to reside in a layer called the *subplate*, which eventually disappears as development proceeds. The next cells to divide become layer VI neurons, followed by the neurons of layers V, IV, III and II.

Cell migration

Many daughter cells migrate by slithering along thin fibers that radiate from the ventricular zone toward the pia. These fibers are derived from specialized *radial glial cells*, providing the scaffold on which the cortex is build. The immature neurons, called *neuroblasts*, follow this initial radial path from the ventricular zone toward the surface of the brain. Recent studies indicate that some neurons actually derive from radial glia. In this case, migration occurs by the movement of the soma within the fiber that connects the ventricular zone and pia. When cortical assembly is complete, the radial glia withdraw their radial processes. Not all migrating cells follow the path provided by the radial glia cells, however. About one-third of the neuroblasts destined to become sub-plate cells are among the first to migrate away from the ventricular zone. Neuroblasts destined to become the adult cortex migrate next. They cross the sub-plate and form another cell layer called the *cortical plate*. The first cells to arrive in the cortical plate are those that will become layer VI neurons. Next come the layer V cells, followed by layer IV cells,

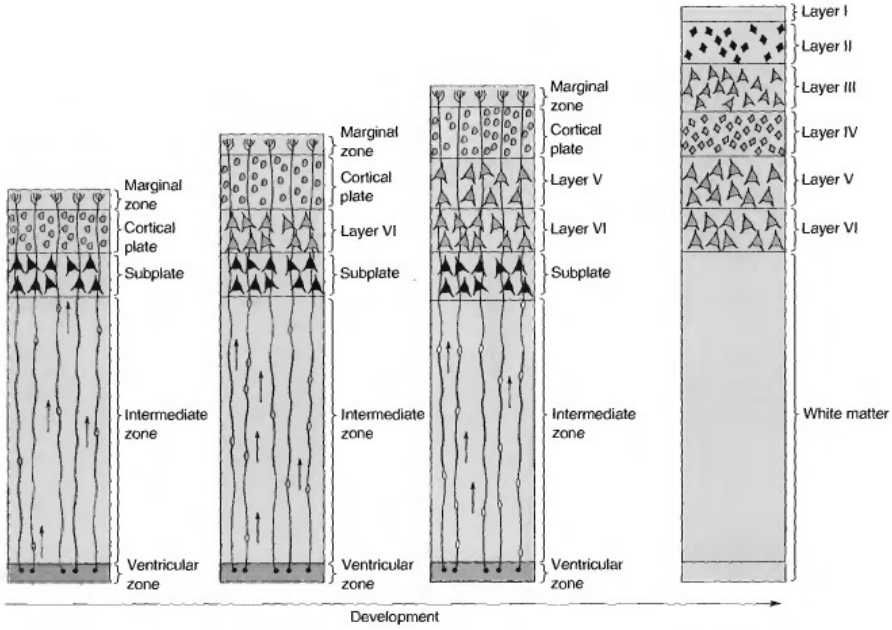


Fig. 4.2 The first cells to migrate to the cortical plate are those that form the sub-plate. As these differentiate into neurons. the neuroblasts destined to become layer VI cells migrate past and collect in the cortical plate. This process repeats again and again until all layers of the cortex have differentiated. The sub-plate neurons then disappear.

and so on. Notice that each new wave of neuroblasts migrates right past those in the existing cortical plate. In this way, the cortex is said to be assembled *inside-out* (see Fig. 4.2). This orderly process can be disrupted by a number of gene mutations.

Cell differentiation

The process in which a cell takes on the appearance and characteristics of a neuron is called *cell differentiation*. Differentiation is the consequence of a specific spatiotemporal pattern of gene expression. Neuroblasts differentiation begins as soon as the precursor cells divide with the uneven distribution of cell constituents. Further neuronal differentiation occurs when the neuroblast arrives in the cortical plate. Thus, layer V and VI

neurons have differentiated into recognizable pyramidal cells even before layer *II* cells have migrated into the cortical plate. Neuronal differentiation occurs first, followed by astrocytes differentiation that peaks at about the time of birth. Oligodendrocytes are the last cells to differentiate.

Differentiation of the neuroblast into a neuron begins with the appearance about the same, but soon, one becomes recognizable as the axon and the other dendrites.

4.2 Moving from biological neural networks to Artificial Neural Networks

After the completion of the morphogenesis process, the resulting structure is a complex network spread on almost seven layers representing one of the fundamental part of the whole abstraction and learning mechanism. Few years ago, in the artificial neural network field there were the first attempt to pass from the usual $1,2$ or 3 levels architecture to a deeper one with $5,6$ or 7 layers, like the human brain really is. The passage was not so easy due to the fact that the common learning algorithms used to train artificial neural network, when applied to this deeper architecture present a great deterioration in term of performances. For example the well known back-propagation learning algorithm run on this *deep architectures* used to get stuck in poor local minima of the objective function [75]. Thus, new ways to train such deep networks have been crafted: the seminal work on this was done by Hinton et al. who introduced the so called Deep Belief Networks (DBN) [62] with a learning algorithm that greedily trains one layer at a time, exploiting an unsupervised learning algorithm for each layer, named Restricted Boltzmann Machine (RBM) [119]. These kind of networks have been applied successfully to a huge number of problems: classification tasks [18, 107, 78], but also in regression [118], dimensionality reduction [117], modeling textures [100], modeling motion [128], object segmentation [80], information retrieval [130], natural language processing [92], etc . However the question remains: is it possible to profitably train a deep neural network with standard algorithms like back-propagation so that to maintain a simple learning procedure without having to pay in terms of performances ?

4.3 An Artificial Neural Network with mobile neurons

Looking at biology we can state that the success of a complex biological neural network is determined by two main factors [88]:

1. an effective mobility of neurons during the brain morphogenesis

2. a selective formation of synaptic connections

In the field of Artificial Neural Networks we can find a huge quantity of techniques inspired on the second point, we can mention for example the ART algorithms [28] or the whole family of growing and pruning methods [20]. While the second point has been deeply investigated, the first one has never been taken into account explicitly, indeed many author preferred to consider directly the entire neural architecture to evolve toward an optimal configuration to be trained [99].

To fill this lack we introduce both a mobility paradigm for the neurons of a multilayer neural network and a methodology to train this new kind of network so that all neurons, though acting as single entities, cooperate together bringing out a collective behavior. The lead idea is that neurons move toward the most informative mates to better learn how to fulfill their part in the overall functionality of the network. Thus, on the one hand neurons from the upper layer attract those lying on the lower layer with the strength of the information they pipe (the back-propagated delta term). On the other hand, the latter repulse one another whenever they do the same job, i.e. have similar weights connecting them with upwards neurons. If we associate a potential field to these attraction/reaction forces, and put neurons in this field as particles ruled by a Newtonian dynamics, we obtain a neuron dynamics that we plan to be modulated by the learning process. With this aim we have to add the Hamiltonian of this motion into the cost function of the training process thus obtaining a great synergy between the physical and information components of an extended Lagrangian ruling the entire dynamics of the network.

Thanks to this contrivance we are able to work well with deep neural network architectures on different benchmarks bypassing the mentioned drawbacks (as seen before) and getting satisfactory learning results, though not yet stretched to compete with the best ones available in the literature. Rather, we use these benchmarks to capture some features of the above physics-cognition synergy.

4.3.1 The model

Let us start with a standard model of an r-layer neural network where all neurons of a layer are located in a Euclidean space $\mathfrak{X}$ which is two-dimensional by default. The activation of the generic neuron is ruled by:

$$\tau_j = f(\mathrm{net}_j); \quad \mathrm{net}_j = \sum_{i=1}^{v_\ell} w_{ji}\lambda_{ji}\tau_i; \quad \lambda_{ji} = \mathrm{e}^{-\mu d_{ji}}; \tag{4.1}$$

where τ_j is the state of the j-th neuron in the $\ell+1$-th layer, w_{ji} the weight of the connection from the i-th neuron of the lower layer ℓ (with an additional dummy connection

in the role of neuron threshold), ν_ℓ the number of neurons lying on layer ℓ, and f the activation function, as usual. By contrast, net_j is a new expression of j-neuron net-input where λ_{ji} is a penalty factor depending on the topological distance d_{ji} between the two neurons, or from another point of view it can be read as the scaling factor that influences the contribution coming from the neurons belonging to the layer ℓ. This reads in the following rule: the closer is a neuron on the layer ℓ to one lying on the $\ell+1$ the greater is its contribution to the state of the latter. Namely, $d_{ji} = \|\mathbf{x}_j - \mathbf{x}_i\|$ [2], where $\mathbf{x}_j$ denotes the position of the neuron within its layer. Note that the distance formula does not take into account the value ℓ of the layer where the single neurons lie. Thus, if they belong to two different layers we may figure them as projected into the same $\mathfrak{X}$ plane. The penalty factor is a distinguishing feature of our model linking the *cognitive aspects* to the neuron dynamics determining the distance d_{ji} – a feature that we ascribe to the *physical aspects* of the network. As for the latter, we assume the neurons to be embedded into a potential field smeared in a single $\mathfrak{X}$ plane into which we project contiguous layers. The potentials are linked to the mentioned attraction/repulsion forces which move the neurons according to a Newtonian dynamics (see Fig. 4.3).

4.3.2 Potentials

Let us fix the notation:

- with the subscript j we refer to the neurons belonging to the layer $\ell+1$;
- with the subscript i,i' we identify the neurons belonging to the layer ℓ;
- ℓ goes from 1 to r.

For each neuron of the network, let us say the i-th neuron, we have:

1. an attraction force A by every j-th neuron of the upward layer expressed by the formula:
$$A = G\frac{m_j m_i}{\zeta_{ji}^2} \tag{4.2}$$
where G is the gravitational constant and ζ_{ji} is the distance between the two neurons in their role of particles of masses m_i, m_j. The distance is considered in a three-dimensional space, where the third coordinate refers to the distance between layers. We assume it to be a constant h so high that it allows us to resume in it both the contribution of the components in the $\mathfrak{X}$ plane and the square root of G, for the sake of simplicity;

[2] With $\|\cdot\|_i$ we denote the L_i-norm, where $i = 2$ is assumed wherever not expressly indicated in the subscript.

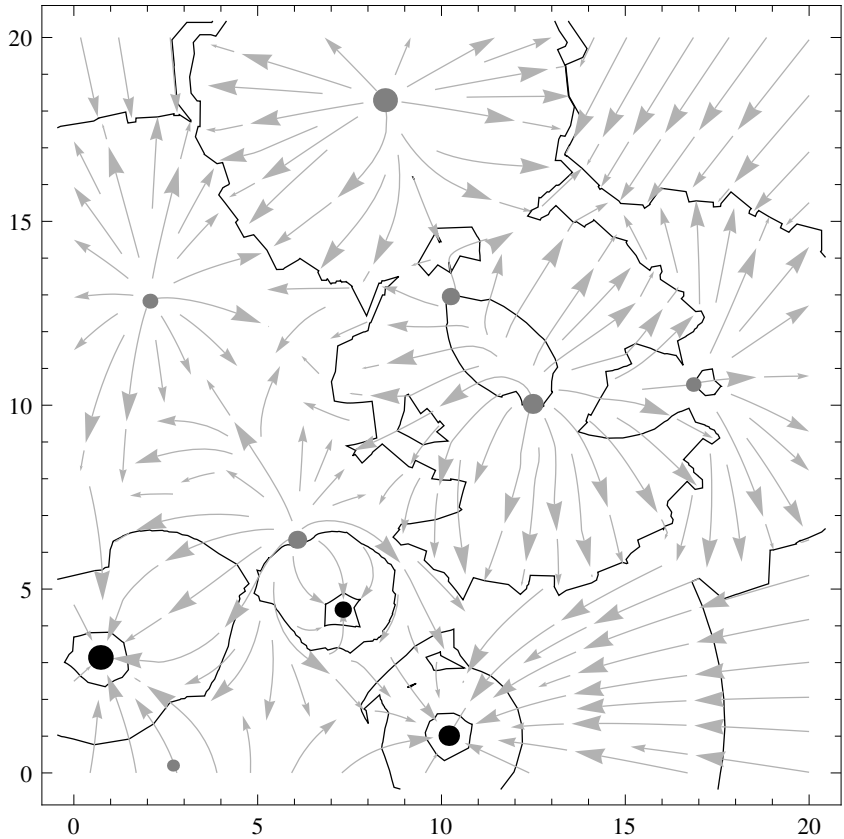


Fig. 4.3 Potential field generated by both attractive upward neurons (black bullets) and repulsive siblings (gray bullets). The bullet size is proportional to the strength of the field, hence either to the neuron mass (black neurons) or to the outgoing connection weight averaged similarity (gray neurons). Arrows: stream of the potential field; black contour lines: isopotential curves.

2. an l-repulsive elastic force R between particles of the same layer which are closer than l, expressed by:
$$R = k_{ii'} \max\{0, l - d_{ii'}\} \tag{4.3}$$
where $k_{ii'}$ is the elastic constant between particles i and i'. The force is linearly dependent on the compression $l - d_{ii'}$ between them.
3. a mass m expressed in term of the information content of each i-th neuron identifiable in our back-propagation training procedure by the error term δ:

$$m_i = \frac{\delta_i}{\|\delta\|_1} \tag{4.4}$$

where in the standard formulation δ is defined as:

$$\delta_i = \begin{cases} f'(\text{net}_i)(\tau_i - o_i) & \text{if } \ell = r \\ f'(\text{net}_i)\sum_j \delta_j w_{ji} & \text{otherwise.} \end{cases} \tag{4.5}$$

In the next paragraphs there will emerge that when $\ell < r$ the formula is quite different due to the introduction of an entropic term we will motivate later.

Let now combine the attraction force A with both the repulsive force R and another term expressing the kinetic energy in correspondence to the neuron velocities v's, obtaining the energy of the network by the formula:

$$\mathscr{P} = \xi_{P_1} P_1 + \xi_{P_2} P_2 + \xi_{P_3} P_3 \tag{4.6}$$

where $\xi_{P_1}, \xi_{P_2}, \xi_{P_3}$ are suitable coefficient in the role of tuning parameters and

$$P_1 = \frac{1}{h}\sum_{i,j} m_i m_j \tag{4.7}$$

$$P_2 = \frac{1}{2}\sum_{i,i'} k_{ii'} max(0, l - d_{ii'})^2 \tag{4.8}$$

$$P_3 = \frac{1}{2}\sum_{i} m_i \|\mathbf{v}_i\|^2 \tag{4.9}$$

Equation (4.7) state the gravitational potential according to (4.2) whereas eq. (4.8) identifies the l-repulsive elastic energy where the elastic constant $k_{ii'}$ is a function of the angular distance between the weight vectors w_i,$w_{i'}$ connecting the i, i'-th neuron to those lying on the layer $\ell + 1$:

$$k_{ii'} = \left| \frac{\langle \mathbf{w}_i \cdot \mathbf{w}_{i'} \rangle}{\|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|} \right| = \left| \frac{\sum_j w_{ij} w_{i'j}}{\sqrt{\sum_j w_{ij}^2}\sqrt{\sum_j w_{i'j}^2}} \right| \tag{4.10}$$

Finally, P_3 is the kinetic energy of the network where v_i is the velocity of the i-th neuron calculated using the following formulas relating together the position x_i, the velocity v_i and the acceleration a_i of each computational moving unit:

$$\mathbf{x}^{(n)} = \mathbf{x}^{(0)} + \sum_{h=1}^{n} \mathbf{v}^{(h)} \boldsymbol{t}^{(h)} = \mathbf{x}^{(n-1)} + \mathbf{v}^{(n)} \boldsymbol{t}^{(n)} \tag{4.11}$$

$$\mathbf{v}^{(n)} = \sum_{h=1}^{n} \mathbf{a}^{(h)} t^{(h)} = \mathbf{v}^{(n-1)} + \mathbf{a}^{(n)} t^{(n)} \tag{4.12}$$

where the superscript refers to h-th discrete time instant. Now we can substitute (4.12) in (4.11) so that we are able to compute $\mathbf{x}^{(n)}$ using the formula:

$$\mathbf{x}^{(n)} = \mathbf{x}^{(n-1)} + \mathbf{v}^{(n-1)} t^{(n)} + \mathbf{a}^{(n)} t^{2(n)} \tag{4.13}$$

so it remains to calculate the acceleration $\mathbf{a}^{(n)}$ at time n. Through the potential $\mathscr{P}$ we are able to calculate the global acceleration of the system: on the one side the potential energy moves the i-th neuron toward the j-th neuron with higher δ, according to the formula

$$\sum_j m_j \mathrm{sign}(\mathbf{x}_j - \mathbf{x}_i) \tag{4.14}$$

on the other side the elastic repulsion moves away the neurons on the same layer with an acceleration i) in inverse proportion to the angular distance between the synaptic weights of the neurons lying on the same layer and ii) in inverse proportion to their distances.

By design we decided to introduce a rest position through the constant ℓ beyond which there is no repulsion, so the complete formula is

$$-\sum_i k_{ii'} \mathrm{m}ax(0, l - d_{ii'}) \mathrm{sign}(\mathbf{x}_{i'} - \mathbf{x}_i) \tag{4.15}$$

and finally we obtain the acceleration formula for the generic neuron i:

$$a_i = \xi_1 \sum_j m_j \mathrm{sign}(\mathbf{x}_j - \mathbf{x}_i) - \xi_2 \sum_i k_{ii'} \mathrm{m}ax(0, l - d_{ii'}) \mathrm{sign}(\mathbf{x}_{i'} - \mathbf{x}_i) \tag{4.16}$$

for proper ξ_is. Actually, for the sake of simplicity we embed the mass of the accelerated particle into ξ_2. Since this mass is not a constant, it turns out to be an abuse which makes us forget some sensitivity terms during the training of the network. Moreover, in order to guide the system toward a stable configuration, we add a viscosity term which is inversely proportional to the actual velocity, namely $-\xi_3 \mathbf{v}_i$, which we do not reckon within the Lagrangian addends for analogous simplicity reasons. It is worth noting that the ξ_h, $h = 1, 2, 3$ multipliers are different from those introduced in (4.6), moreover here the superscript indicating the time instant is not reported for a better readability.

4.3.3 Error term

The missing part that completes our model creating in the meanwhile a cost function comprehensive of both dynamics and cognitive aspects, can be resumed in:

- a customary quadratic error function

$$E_c = \frac{1}{2}\sum_o (\tau_o - z_o)^2 \tag{4.17}$$

  where o indexes the output neurons, τ is the network output and z is the target output.
- an entropic term devoted to promoting a representation of the neuron state vector through independent Boolean components via a Schur-concave function as discussed in Section 2.5. According to the notation of this section, the edge pulling function reads now:

$$E_b = \ln\left(\prod_i \tau_i^{-\tau_i}(1-\tau_i)^{-(1-\tau_i)}\right) \tag{4.18}$$

As a conclusion, we put together both the Lagrangian of physical part of the neuron motion and the cognitive parts regarding the error terms, thus obtaining an Hamiltonian $\mathscr{H}$ of the form:

$$\mathscr{H} = \xi_{e_c}E_c + \xi_{e_b}E_b + \xi_{p_1}P_1 + \xi_{p_2}P_2 + \xi_{p_3}P_3 \tag{4.19}$$

and substituting with have:

$$\mathscr{H} = \xi_e \frac{1}{2}\sum_o (\tau_o - z_o)^2 + \xi_b \ln\left(\prod_i \tau_i^{(-\tau_i)}(1-\tau_i)^{-(1-\tau_i)}\right) + \\ \xi_1 \sum_{i,j} m_i m_j + \xi_2 \frac{1}{2}\sum_{i,i'} k_{ii'} \mathrm{m}ax\{0, l - d_{ii'}\}^2 + \xi_3 \frac{1}{2}\sum_i m_i \|\mathbf{v_i}\|^2 \tag{4.20}$$

We want to minimize (4.20) in order to find the configuration of lower energy that better approximates the function we are trying to learn.

4.4 Training the network

In spite of recent trends to look for different strategies for training the multilayer perceptron, specially if designed with more than one hidden layer, once again we commit back-propagation (Chapter 2) to perform this task. The idea is that the dynamic we introduce on the neurons' layout, along with the augmentation of the cognitive aspects,

will be sufficient to get rid of some of the standard implementation drawbacks [17]. *Per se*, the classical back-propagation algorithm comes directly from the functional minimization of a Lagrangian related exactly to the quadratic error E_c [36]. Rather, taking for granted the Newtonian dynamics that comes from the extended Lagrangian as well, we focus on the Hamiltonian of our system and specifically look for its parametric minimization. This leads to the identification of the system *ground state* which according to quantum mechanics denotes one of the most stable configurations of the system equilibrium. Actually, the framing of neural networks into the quantum physics scenario has been variously challenged in recent years [57]. However these efforts mainly concern the quantum computing field, hence enhanced computing capabilities of the neurons. Here we consider conventional computations and classical mechanics. Rather, *superposition* aspects concern the various endpoints of learning trajectories. Many of them may prove interesting (say, stretching valuable intuitions). But a few, possibly one, attain stability with a near to zero entropy – a condition which we strengthen through the additional viscosity term.

Looking for the minimum of the cost function (4.20) requires only a careful computation of the derivatives of all its addends w.r.t. each connection weight w_{ji}. In particular, as for the cognitive part, we must consider the dependence of λ_{ji} on w_{ji} which passes through the dependence of the position $\mathbf{x}_i$ and $\mathbf{x}_j$ on the same parameter. As for the physical part, we must also consider the dependence of the acceleration $\mathbf{a}_i$, velocity $\mathbf{v}_i$ and the elastic constant k_{ji} on w_{ji}.Thus the main expressions related to the above derivatives are summarized in Table 4.1.

The interested reader could find all the details about the derivatives for the back-propagation in Appendix A.

4.5 Numerical Experiments

Adding degrees of freedom to a neural network makes sense only if we are able to govern its evolution. The failure of this attitude is the common reason why we generally abandon the implementation of both recurrent neural networks [63] and the most ingenuous versions of deep architectures as well [48]. Hence we stress our network of mobile neurons – which copes both with the recursiveness through the neurons physical motion and with the deepness of the architecture through a relatively high number of its layers – on two typical benchmarks as a specimen of a regression problem and a classification task, respectively.

1. The Pumadyn benchmark pumadyn8-nm. It is drawn from a family of datasets which are synthetically generated from a Matlab simulation of a robot arm [34]. It contains $8,192$ samples, each constituted by 8 inputs and one output. The former record the

error	$\frac{\partial(E_c+E_b)}{\partial w_{ji}} = \left(1-\frac{w_{ji}}{d_{ji}}(\mathbf{x}_j-\mathbf{x}_i)\frac{\partial \mathbf{x}_i}{\partial w_{ji}}\right)\lambda_{ji}\tau_i\delta_j;$ $\delta_j = f'(\mathrm{net}_j)\left(-\gamma\ln\left(\frac{\tau_j}{1-\tau_j}\right)+\sum_k \delta_k\lambda_{kj}w_{kj}\right); \quad \gamma=\frac{\xi_{e_b}}{\xi_{e_c}}$
potential	$\frac{\partial P_1}{\partial w_{ji}} = m_j\frac{\mathrm{sign}(\delta_i)}{\lVert\delta\rVert_1}(1-m_i)f'(\mathrm{net}_i)\delta_j$ $\frac{\partial P_2}{\partial w_{ji}} = \sum_{i'}\frac{1}{2}\max\{0,l-d_{ii'}\}^2\frac{\partial k_{ii'}}{\partial w_{ji}}+\sum_{i'}\frac{k_{ii'}}{d_{ii'}}\max\{0,l-d_{ii'}\}(\mathbf{x}_{i'}-\mathbf{x}_i)\frac{\partial \mathbf{x}_i}{\partial w_{ji}}$ $\frac{\partial P_3}{\partial w_{ji}} = \frac{1}{2}\lVert\mathbf{v}_i\rVert^2\frac{\mathrm{sign}(\delta_i)}{\lVert\delta\rVert_1}(1-m_i)f'(\mathrm{net}_i)\delta_j+m_i\mathbf{v}_i\frac{\partial \mathbf{v}_i}{\partial w_{ji}}$
dynamics	$\frac{\partial \mathbf{a}_i^{(n)}}{\partial w_{ji}} = -\xi_2\left(\sum_{i'}(\max\{0,l-d_{ii'}\})\mathrm{sign}(\mathbf{x}_{i'}-\mathbf{x}_i)\frac{\partial k_{ii'}}{\partial w_{ji}}+ -\sum_{i'}k_{ii'}\mathrm{sign}(\mathbf{x}_{i'}-\mathbf{x}_i)\frac{\partial d_{ii'}}{\partial w_{ji}}\right)$ $\frac{\partial \mathbf{v}_i^{(n)}}{\partial w_{ji}} = \frac{\partial \mathbf{v}_i^{(n-1)}}{\partial w_{ji}}+t_n\frac{\partial \mathbf{a}_i^{(n)}}{\partial w_{ji}}$ $\frac{\partial \mathbf{x}_i^{(n)}}{\partial w_{ji}} = \frac{\partial \mathbf{x}_i^{(n-1)}}{\partial w_{ji}}+t_n\left(\frac{\partial \mathbf{v}_i^{(n-1)}}{\partial w_{ji}}+t_n\frac{\partial \mathbf{a}_i^{(n)}}{\partial w_{ji}}\right)$
	$\frac{\partial k_{ii'}}{\partial w_{ji}} = \mathrm{sign}\left(\frac{\langle\mathbf{w}_i\cdot\mathbf{w}_{i'}\rangle}{\lVert\mathbf{w}_i\rVert\cdot\lVert\mathbf{w}_{i'}\rVert}\right)\frac{w_{ji'}\lVert\mathbf{w}_i\rVert\cdot\lVert\mathbf{w}_{i'}\rVert-\langle\mathbf{w}_i\cdot\mathbf{w}_{i'}\rangle w_{ji}\frac{\lVert\mathbf{w}_{i'}\rVert}{\lVert\mathbf{w}_i\rVert}}{(\lVert\mathbf{w}_i\rVert\cdot\lVert\mathbf{w}_{i'}\rVert)^2}$ $\frac{\partial d_{ii'}}{\partial w_{ji}} = -\frac{\mathbf{x}_{i'}-\mathbf{x}_i}{d_{ii'}}\frac{\partial \mathbf{x}_i}{\partial w_{ji}}$

Table 4.1 Gradient expressions for the backward phase from last-but-one layer down.

angular positions and velocities of three arm joints plus the value of two applied torques. The latter is the resulting angular acceleration of one of the joints. This acceleration is a nonlinear function of the inputs which is affected by moderate noise as well.

2. The MNIST benchmark. This handwritten digits benchmark [77] consists of a training and a test set of 60,000 and 10,000 examples, respectively. Each example contains a 28×28 grid of 256-valued gray shades (between 0 and 255, with 0 meaning black and 255 white). It is a typical benchmark for classification algorithms, given the high variability of the handwritten representation of the same digit.

With these highly demanding datasets, at the moment we have no ambition to win a competition with the many methods which have been tossed on them. Rather, we want to use these data as different-size/different-task instances on which to carry out a test lap of the new learning machinery we have set up.

4.5.1 Assessing the new training procedure

At the start of the lap, we had to solve three kinds of partly nested problems:

1. designing the architecture,
2. assessing the particle dynamics,
3. tuning the learning parameters.

The solution we assessed for the moment are the following:

1. As for the former, we strove to stress the capability of working with deep networks. Therefore we devised a 5-layer network in each experiment. We distribute the neurons on the crosses of a plane square grid in each layer, apart from the output layer. We standardize the grid edge to a size equal to 100 and centered each grid in $(0,0)$. Then, the number of neurons per layer is determined by the benchmarks as for input and output, while is decided by us in order to be *sufficient* to manage the piped information, as for the intermediate layers. In particular,

   - for Pumadyn dataset we have 8 input neurons in a grid of 3×1 cells and only one output neuron. This results in a $8\times100\times80\times36\times1$ perceptron.
   - For the MNIST benchmark, facing the 28×28 raster of the digits, we reduce the number of pixels by 4, just by periodically taking the average of contiguous values on both axes. Hence we have an input layer of 196 neurons spread in a grid of 13×13 cells. Concerning the output, we opted for a unary representation; hence we commit a single neuron to answer 1 and the others 0 on each digit. We distributed these neurons on a circle of ray equal to 50 centered in the axes origin like the other layers. In conclusion, we have a $196\times120\times80\times36\times10$ network.

   This is for the initial layouts which we may see in Fig. 4.1e. Then the neurons will find their proper positions by themselves.
2. The physics of the particle motion is characterized by a time step t_n equal to 0.02 per state update. This is a critical parameter tuning the velocity of the physical particles within the Euclidean space w.r.t. the velocity of the same particles in the weight space. We assume that particles are able to find the best reciprocal positions. However, an exceeding velocity could thwart any training effort because of the continually changing configuration of the network. In Fig. 4.7 we highlight this crucial aspect which we may synthesize by claiming that we need a gentle particle motion to avoid hard training backtracking which frustrates the overall training.
   Our neurons are cognitive particles which need a proper metering system. We take care of it in a tight empirical way through the ξ coefficients with the practical aim of balancing the cost terms in the Hamiltonian (4.20). It is suggestive the fact that we use the same coefficients in the two benchmarks, in spite of the very different goals of the two experiments.

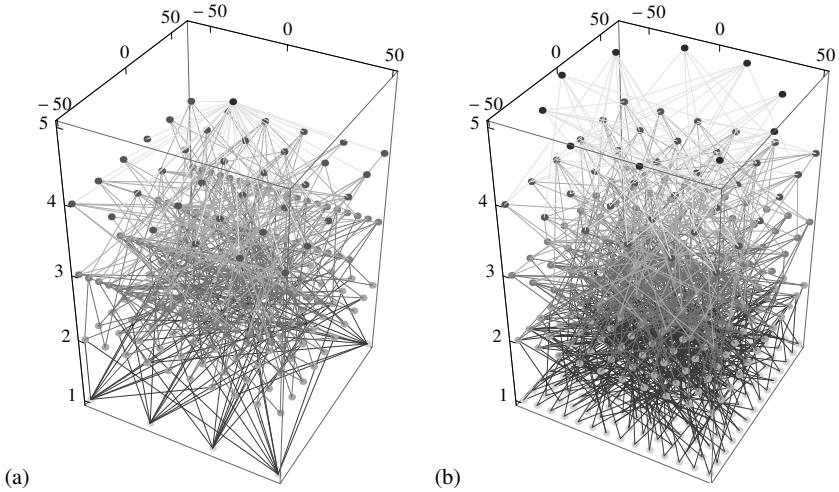


Fig. 4.1e The initial network layouts for: (a) Pumadyn, and (b) MNIST.

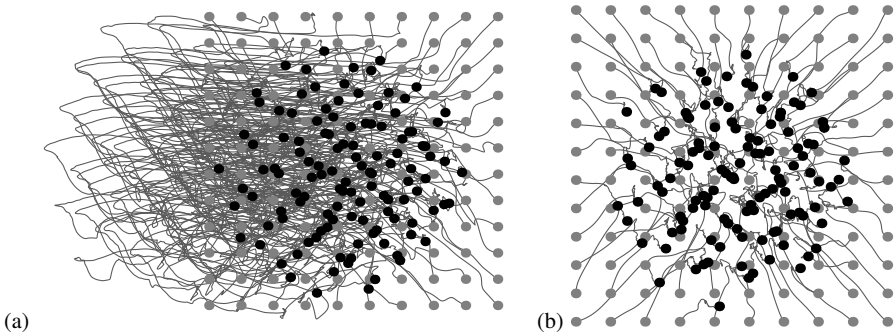


Fig. 4.7 Neuron trajectories of: (a) a not well tuned training story, and (b) a successful one where the dynamic is more gentle. Bullets: neurons of the second layer in the MNIST benchmark (grey/black: initial/final position).

3. As for the cognitive aspects, the effect of neuron motion on the training process is linked to two factors. An indirect one is represented by the enriching of the cost function with physical energy terms, so that its minimization entails that the connection weights must change also in order to *freeze* the dynamics of the ground state. The more direct effect comes from the penalty term λ_{ji} which reduces the influence

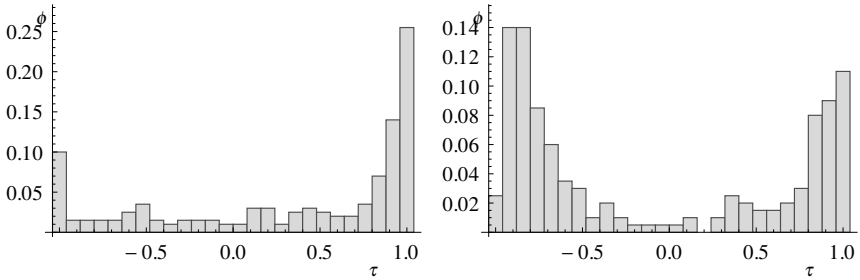


Fig. 4.9 Histograms of the state values of two typical hidden layer neurons at the end of training.

of farther neurons in the net-input accumulation. This plays a dual role of: 1) modulating the load of the Hamiltonian gradient cumulated over the reversed outgoing connections, hence the amount of weight changes along the training steps, and 2) determining the specialization rate of the various neurons w.r.t. the facing mates in the contiguous layers. We experimented on several alternative time strategies aimed at either reducing or increasing the radius of influence of the single back-propagating neurons, attesting at end that the better choice is to keep the radius at fixed value – it could be a different one layer per layer – along the training.

A second key aspect in the training process is the effect of the BICA term (4.18). As aforementioned, this cost term is devoted to promoting a meaningful representation of the network state vector τ at each layer, in terms of tendentially binary independent components. At the beginning of the training, specially if we initialize the weights to very small values in order to prevent bias, this term may prove exceedingly high, and thus requires a proper calibration of the coefficient γ. As a matter of fact, we want this stretching of the state vector components toward the extremes to occur at the end of the training, so as to saturate the output of the activation functions to suitable values. Moreover, to increase the sensitivity of the system, we adopt the usual back-propagation trick of normalizing the state range to $(-1,1)$ through the hyperbolic tangent activation function. The sole exception is the output neuron's of the regression task, which is linear. Thus a typical histogram of the component states is the one depicted in Fig. 4.9.

In essence, we venture the conjecture that motion plus BICA goal induce a preprocessing of the state vector in the same direction as more explicit techniques like Restricted Boltzmann machines [51] in the role of autoencoders [132], but in a simpler and quicker way, without straying from the root of the back-propagation method. The curves in Fig. 4.10 confirm our conjecture. Namely, the descent of the error term E_c summed over the training set – the training Mean Square Error (MSE) – with the

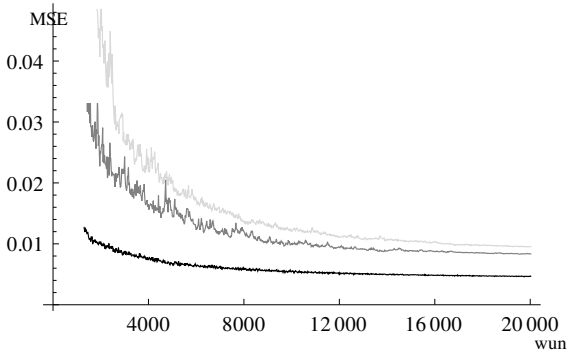


Fig. 4.10 Course of training MSE with weight updates' number (wun) for the regression problem. Same architecture different training algorithms: light gray curve → standard back-propagation, dark gray curve → back-propagation enriched with the BICA term, black curve → our mob-neu algorithm.

iterations improves when we add the BICA term in the cost function of the standard backpropagation, and improves still more when we add motion to the neurons of the same architectures.

To complete the setup of the training procedure we have to fix the learning rate parameters. We differentiate a coefficient concerning the contribution to weight adaptation deriving from the weight sensitivity of the cognitive part ($E_c + E_b$) from another linked to the analogous sensitivity of the dynamic part ($P_1 + P_2 + P_3$). Both are exponentially decaying of a pair of orders with the number of elapsed steps, whereas their fixing depends on the benchmark features and on batch size, as usual.

4.5.2 Discussing some results

4.5.2.1 The regression task

In Table 4.2 we compare the test-set MSE of our procedure on the Pumadyn benchmark with the ones obtained with other methods in the literature. To have a fair comparison we use a completely standard framework. Namely, according to the Delve testing scheme [108] applied to the pumadyn-8nm dataset, we generate five groups of training sets, of size $64, 128, 256, 512$ and 1024, respectively. Each group is made up of 8 different subsets (4 subsets for the 1024 one) of the downloaded dataset, each paired with a different test set of fixed dimension 512 (1024 for the last one) drawn from the same

dataset. Each pair training/test-set is the basis for one experiment. To have comparable results to be statistically considered for the same training set size, targets are normalized around their median value using the Median Absolute Deviation as a scale factor. MSE results are computed on the original feature set removing the above normalization. Namely, on this framework we compare the mean and standard deviation of the MSEs obtained by our network of mobile neurons (mob-neu) with those of other methods drawn from the Delve repository. For small sizes (64 and 128), our method outperforms the others based on back-propagation even when they benefit from both test and validation sets, while it loses out w.r.t. regression methods based on different strategies (for instance hidden Markov models, Bayesian methods, and so on). Things become less sharp with higher sizes, where a few trespasses occur between the methods as for the means and tangible overlaps as well of the confidence intervals. Admittedly, the considered methods date at least 20 years, yet offering a completely definite comparison framework. Looking at the most recent results we find notably progresses in this regression problem. For instance, support vector regression methods improve these results of even two orders when they are based on very specialized kernels [38], while model selection and specific error functions may reduce the MSE of the back-propagation procedure by a factor ranging from 2 to 3 [122]. Rather, we remark that our implementation was very naive, in the sense that we did not spend much time searching for the best updating strategy and parameter tuning, nor did we stretch out the procedure for long time. Moreover, we did not miss the general-purpose features of the training scheme. We simply enriched the structure and the functionality of the underlying neural network.

The typical MSE descent of the training error (in the normal scale) with the number of weight updates is reported in Fig. 4.11(a) for a batch size equal to 20. We appreciate the generalization capability of the network in Fig. 4.11(b), where we represent in gray the sorted test set targets and in black the corresponding values computed by our network for one of the 8 replicas of the 512 experiment. In Fig. 4.11(c) we also plot the sorted lists of the absolute errors obtained by our method and a couple of competitors. Since the latter refer to the first and last rows of Table 4.2 we may realize how close the proposed method works to the best one.

Lastly, in Fig. 4.12(a) we report the final layout of the neural network after training, as a companion to the initial layout reported in Fig. 4.1e(a). Even though we do not have yet a functional interpretation of the new layout, its features denote some persistency along the Delve replicas, which put them far from a random generation.

4.5.2.2 The classification task

Moving on the classification problem, Table 4.3 shows the confusion matrix of the classifier on the entire MNIST test set we get directly from the output of a well trained neural

network (i.e. assigning to a test pattern the label of the output neuron which computes the highest state value). Postponing a more deeply statistical analysis to a further work, we note in the matrix a fair distribution of the errors on the digits, though reflecting the widely recognized greater difficulty of classifying digits like '9'.

The cumulative error rate is about 3.26%, that is definitely worse than the best results, which attest at around one order less [31], yet without suffering a bad ranking w.r.t. a plenty of methods given the amount of tuning efforts (2 days), the number of weight updates (50,000) and the length of the running time (128 minutes) we devote to this task. For instance, in Fig. 4.13 we report a few example of hard handwritten digits that have been correctly classified, together with another set of such digits on which the network did not succeed. Considering their shape we wander if it makes sense looking for a network which recognizes the second picture of the first line to be a 9. Oreationally, a recognition percentage of 99.7% is better than 97.0%, of course. But it sounds like to give label to random noise. Hence a question arises on the replicability of the experiment on another dataset.

Fig. 4.14 shows the error decreasing of the 10 digits classification. We use a batch size of 30 examples randomly drawn from the 60,000 long training set, whereas the error reported in Fig. 4.14 is measured on 200 *new* examples – actually a generalization error causing some ripples on the trajectories of the single digits.

From Fig. 4.12(b) we capture the different displacement of the nodes w.r.t. the original layout, due to a motion which preserves some symmetries but generates some pe-

	64	128	256	512	1024
mlp-mc-1	1.96726 (0.276)	1.61052 (0.154)	1.25143 (0.0613)	1.13004 (0.0397)	1.07711 (0.0355)
gp-map-1	2.00875 (0.132)	1.55786 (0.0506)	1.31765 (0.0363)	1.13152 (0.0319)	1.0794 (0.0312)
mob-neu	5.25507 (0.541)	2.24522 (0.154)	1.81821 (0.096)	1.28294 (0.036)	1.21273 (0.034)
mlp-mdl-3h	6.40296 (0.620)	3.12531 (0.475)	2.82356 (0.575)	2.35649 (0.417)	2.1694 (0.614)
mlp-mc-2	8.93908 (2.53)	1.43405 (0.0412)	3.21679 (1.85)	2.35757 (1.22)	1.08938 (0.0322)
mlp-ese-1	9.36373 (0.776)	3.81698 (0.225)	1.73787 (0.0455)	1.37629 (0.0378)	1.2133 (0.0340)
mlp-wd-1	19.3383 (1.84)	6.78868 (1.76)	3.56734 (0.252)	2.71222 (0.228)	2.14455 (0.364)

Table 4.2 Comparison of regression performances. Raw: method; column: size of the pumadyn8-nm training set; cell: MSE average and standard deviation (in brackets)

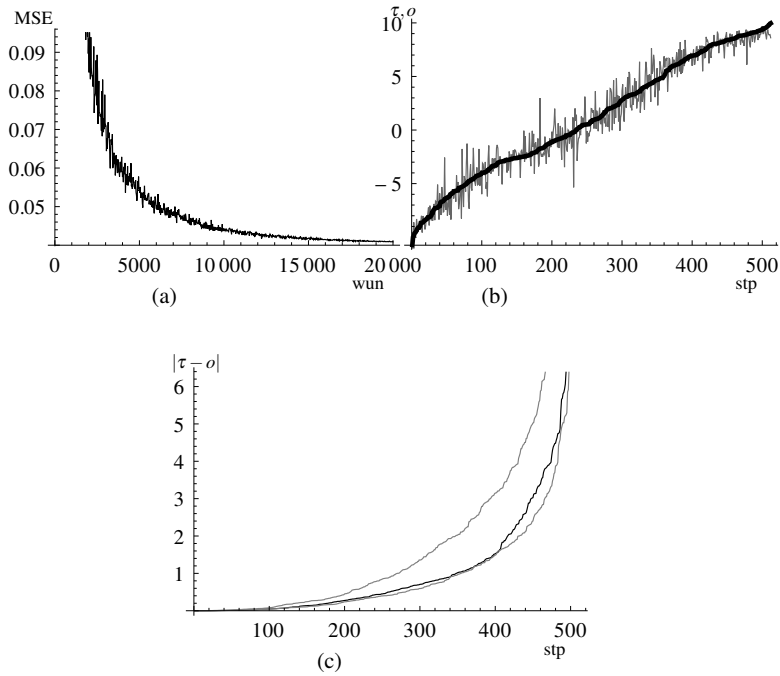


Fig. 4.11 Errors on regressing Pumadyn. Course of: (a) training MSE with weight updates' number , (b) network output with sorted target patterns (stp), and (c) sorted absolute test error of our model (black curve) and competitor mlp-mc-1 and mlp-wd-1 (gray curves) in Table 4.2.

culiar features as well. As mentioned before, we cannot claim to have reached a great classification result in absolute. However we use a relatively small amount of computational resources, and process examples that are roughly compressed just by averaging neighboring pixels. Nevertheless, we are able to classify digits which would prove ambiguous even to a human eye. In addition, we are able to train a network of 5 layers and 442 nodes without falling in a local minimum outputting the average target value, as often happens with these deep networks. Thus we are motivated to investigate whether and how the introduction of the dynamics in the network layers could care these benefits.

A first consideration is that thanks to the large number of the employed neurons and their stable evolution, almost the entire (60,000 long) training set is well digested by the training procedure, even though each example is visited meanly twice according to our batching strategy. We assume it to be due to the cognitive structure the moving neurons realize within the network. As an early analysis, in Fig. 4.15 we capture the typical

dendritic structure of the most correlated and active nodes reacting to the features of a digit. Namely, here we represent only the nodes whose output is significantly far from 0 with a significant frequency during the processing of the test set. Then we establish a link between those neurons, belonging to contiguous layers, which are highly correlated during the processing of a specific digit. In this respect, the pictures in Fig. 4.15 highlight a dendritic structure underlying the recognition of digits '3' and '4'. An analogous analysis on intra-layer neurons highlights cliques of neurons jointly highly correlated in correspondence of the same digit, as shown in Fig. 4.16.

4.6 Further improvements

Fuzzyfication of λs

The influence of the cognition on the dynamics is ruled by the cognitive masses and the elastic constants, as shown before. The reciprocal influence of the dynamics on cognition is determined by the penalty terms λ_{ji}s. We may consider these coefficients in terms

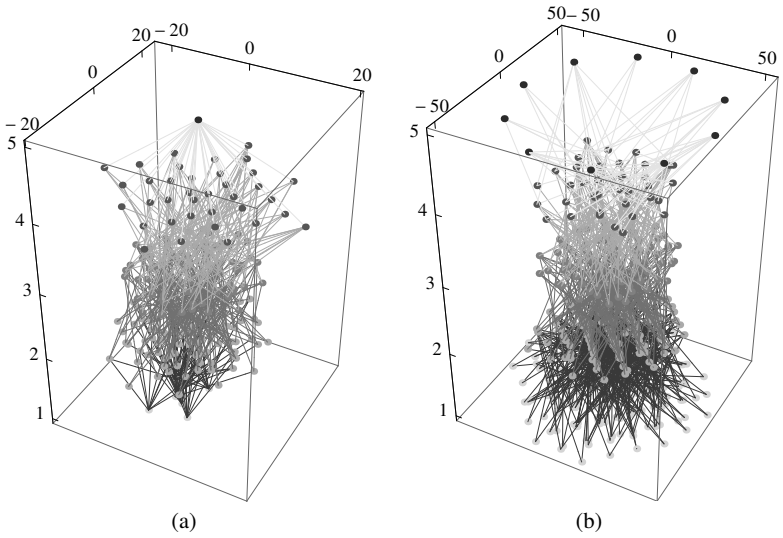


Fig. 4.12 The final layout of the multilayer perceptrons in Fig. 4.1e.

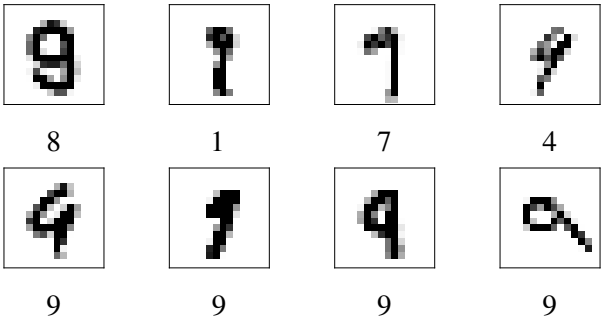


Fig. 4.13 Examples of wrongly and correctly classified '9's.

of membership degrees (see Fig 4.17). Actually the upper layer neurons act as attractors of the lower-layer neurons as for the information piping. As their core business is to learn, the latter must decide to which attractor to pay more attention. In terms of fuzzy sets, this translates in the solution of a fuzzy c-means algorithm (FCM) [42] with given cluster centroids (the above attractors), once a proper metric has been fixed. Using the Euclidean distance of the lower-layer neuron to its upper-layer attractor as a metric, the classical instantiation consists in solving the minimization problem:

$$\min_{\lambda} \sum_{ij} \lambda_{ji}^{\mu} d_{ji}^{2}; \quad \mu > 0; \sum_{j} \lambda_{ji} = 1 \tag{4.21}$$

which find solution:

target \ output	0	1	2	3	4	5	6	7	8	9	% err
0	964	0	2	2	0	2	4	3	2	1	1.63
1	0	1119	2	4	0	2	4	0	4	0	1.41
2	6	0	997	2	2	2	4	11	8	0	3.39
3	0	0	5	982	0	7	0	6	6	4	2.77
4	1	0	2	0	946	2	7	2	1	21	3.67
5	5	1	1	10	0	859	7	1	5	3	3.70
6	5	3	1	0	3	10	933	0	3	0	2.61
7	1	5	14	5	2	1	0	988	3	9	3.89
8	4	1	2	8	3	3	8	4	938	3	3.70
9	5	6	0	11	12	8	1	11	7	948	6.05

Table 4.3 Confusion matrix of the MNIST classifier.

$$\lambda_{ji} = \frac{1}{\sum_k \left(\frac{d_{ji}^2}{d_{ki}^2}\right)^{\frac{1}{\mu-1}}} \tag{4.22}$$

Here μ plays the role of the overposition factor. Like as in quantum mechanics, a neuron may belong to more than one attracting basin (actually, its state) when $\mu \neq 1$. Whereas, the deterministic case, i.e. each neuron of the lower layer attributed to a single neuron of the upper layer, occurres with μ exactly equal to 1. We get a solution more compliant with our expression of the penalty term in (4.1), considering the negative entropy as membership regularization [131] in the FCM solution. In this case the objective function read:

$$\min_{\lambda} \sum_{ij} \lambda_{ji} d_{ji}^2 + \mu \sum_{ij} \lambda_{ji} log_{ji}; \quad \mu > 0; \sum_j \lambda_{ji} = 1 \tag{4.23}$$

with solution:

$$\lambda_{ji} = \frac{\exp(-\mu d_{ji}^2)}{\sum_k \exp(-\mu d_{ki}^2)} \tag{4.24}$$

with an analogous meaning of λ. The λ values computed in this way present some normalization problems, since the constraint $\sum_j \lambda_{ji} = 1$ gives rise to very small values when the number of cluster is high, as usual with our networks. Therefore we restrict the sum to the sole values exceeding a suitable quantile. In Fig. 4.18 is represented the *per* layer λs distribution of these coefficients. Their shapes denote a different role of the nodes in the various layer as for their specialization.

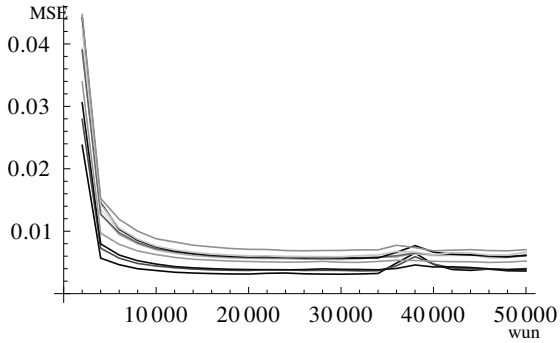


Fig. 4.14 Course of the single digits testing errors with the number of weight updates.

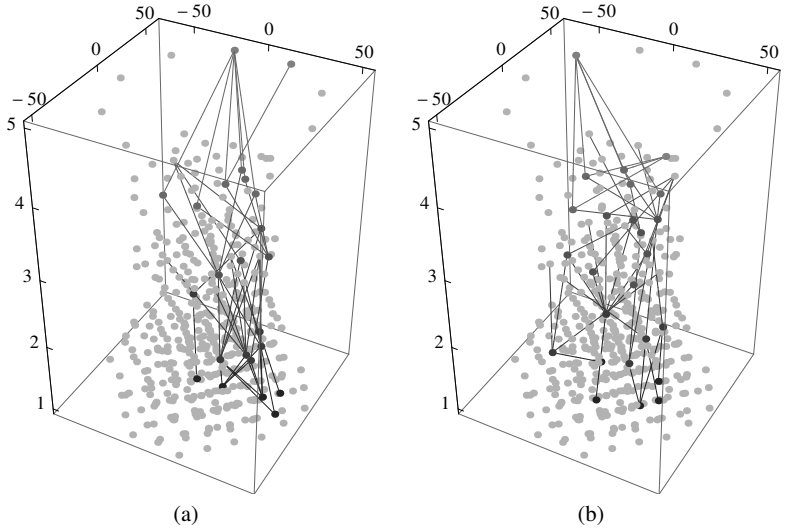


Fig. 4.15 Dendritic structure in the production of digits: (a) '3', and (b) '4'.

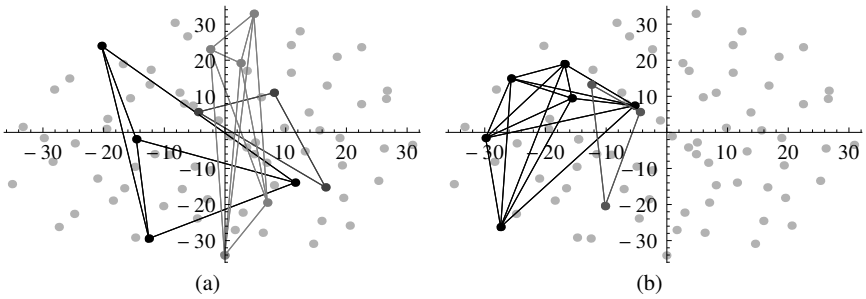


Fig. 4.16 Cliques of highly correlated neurons on the same digits of Fig. 4.15. Bullets: locations of all 2-nd layer neurons.

A unified learning space

Recalling our aim regarding the study about the impact of the intentionality in learning processes we want to strengthen the relationship between the topological space motion of neurons and the cognitive information passing through them. In this way we are

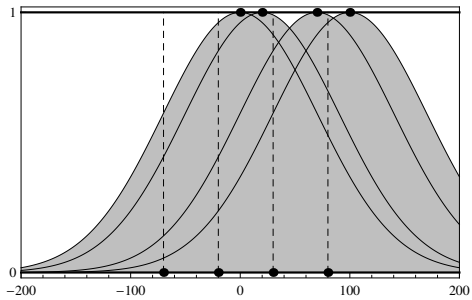


Fig. 4.17 Example of the membership functions determining the influence received by the downward neurons from upward ones.

merging two substantially different spaces - the topological space and the weights space - into a unified space. As a result we stress the membership of a lower-layer neuron to a upper-layer companion by directly enhancing the absolute value of their connection weight when the neurons get closer (i.e. the distance between them diminishes) and *vice versa* when they get farther apart. Symmetrically, we directly push closer those neurons in contiguous levels whose absolute value of the connection weights increases (and *vice versa*). Namely, letting α, δw_{ji}, c be the momentum term, the weight increment term ruled by the back-propagation algorithm, and a suitable regularization term respectively, we have the following update rules:

$$\begin{aligned} w_{ji}(t+1) &= w_{ji}(t) + \Delta w_{ji}(t) \\ \Delta w_{ji}(t) &= ((1-\alpha)\Delta w_{ji}(t-1) + \\ &\quad \alpha\,(\delta w_{ji}(t) - c\ \mathrm{sign}(w_{ji}(t))\,(d_{ji}(t) - d_{ji}(t-1)))) \end{aligned} \tag{4.25}$$

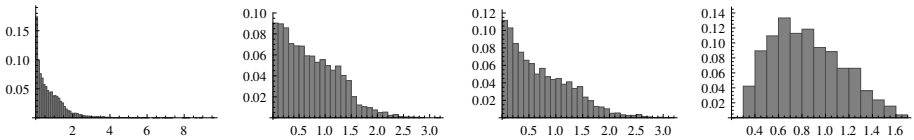


Fig. 4.18 Distribution of λ coefficients on the various layers ℓs at the last training iteration.

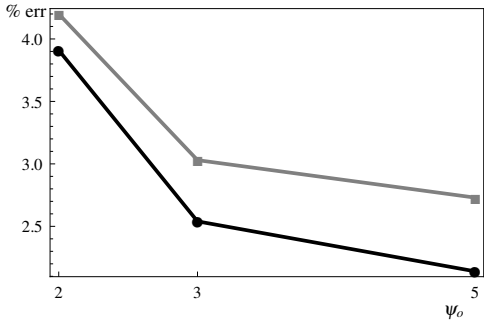


Fig. 4.19 Course of the train (black) and test (gray) misclassification error on MNIST dataset with an increasing number of output neurons ψ_o associated to the single target classes.

$$\begin{aligned} \mathbf{x}_i(t+1) &= \mathbf{x}_i(t) - \Delta\mathbf{x}_i(t) \\ \Delta\mathbf{x}_i(t) &= \sum_j \frac{\mathbf{x}_i(t) - \mathbf{x}_j(t)}{d_{ji}(t)} \operatorname{sign}(w_{ji}(t)) \delta w_{ji}(t) \frac{1}{c} \end{aligned} \tag{4.26}$$

In this way both weight and positions are very tightly correlated.

Replicated outputs

We further drawn on the biological original of our artificial neural network by replicating the goals of the output nodes. This induces a replicated representation of external objects and, possibly, a better abstraction capability of our machinery. Think about the classification task presented before with the MNIST dataset: we have 10 classes referring to handwritten digits from 0 to 9 and it is plausible to think that even in each single class of digits (e.g. 0) we can find different samples that could be clustered in respect to secondary features of the handwritten characters. As a counterpart, these features, if not properly managed, may induce misclassifications. To fill this gap our approach is to increase the number of output neurons per single class and train the network without any further modification. At a first glance the results are promising, in Fig. 4.19 we can notice a decreasing trend in the error percentage that goes with the number of output neurons per single target class. In Table 4.5 is shown the final confusion matrix w.r.t. the test set, that, if compared with Table 4.3 we can notice a significant, though not yet really competitive (see Table 4.6), improvement in the misclassification error percentage reaching a final 2.76%.

The new setup of the last layer introduce the necessity of putting into the game a proper approach to deal with replicated outputs. We investigated different ways to achieve this: the first and until now more profitable approach consists in the plain multiplication of the number of output neurons per target class, followed by the usual back-propagation of the corresponding δ-errors. We tried also to induce a different behavior in term of the quantity of δ-error back-propagated in order to promote a better differentiation between the intra-class output but without getting better results than with the plain approach. However this is still an open research line. In Fig. 4.20 we report the fifth layer layout of an experiment where we split ten times the output. Hence 10 neurons are deputed to output 0, then 1 and so on. The location of the neurons on the layer prove to be essential for the success of the results.

target \ output	0	1	2	3	4	5	6	7	8	9	% err
0	5842	2	11	4	4	9	23	1	18	9	1.37
1	1	6626	35	15	8	0	3	17	30	7	1.72
2	7	8	5850	15	10	8	5	28	17	10	1.81
3	4	1	50	5910	3	55	2	44	37	25	3.60
4	5	13	9	0	5722	5	18	7	7	56	2.05
5	7	4	3	30	6	5303	35	1	15	17	2.18
6	13	6	1	2	8	26	5851	0	11	0	1.13
7	3	9	33	6	24	1	0	6124	11	54	2.25
8	10	22	18	16	11	25	14	3	5712	20	2.38
9	11	7	4	18	39	19	2	40	31	5778	2.87

Table 4.4 Confusion matrix of the MNIST classifier on $60,000$ train samples.

target \ output	0	1	2	3	4	5	6	7	8	9	% err
0	967	0	1	0	0	2	4	1	5	0	1.33
1	0	1121	4	2	0	1	2	0	5	0	1.23
2	7	1	1005	4	3	0	0	6	6	0	2.62
3	0	0	9	980	0	7	1	7	4	2	2.97
4	1	1	2	0	954	1	4	1	3	15	2.85
5	3	0	0	7	1	863	7	2	5	4	3.25
6	2	2	0	0	6	11	935	0	2	0	2.40
7	1	3	13	1	1	1	0	987	6	15	3.99
8	4	0	2	7	2	2	5	3	948	1	2.67
9	4	3	0	7	11	7	0	5	5	967	4.16

Table 4.5 Confusion matrix of the MNIST classifier on $10,000$ test samples in the *replicated output experiment*

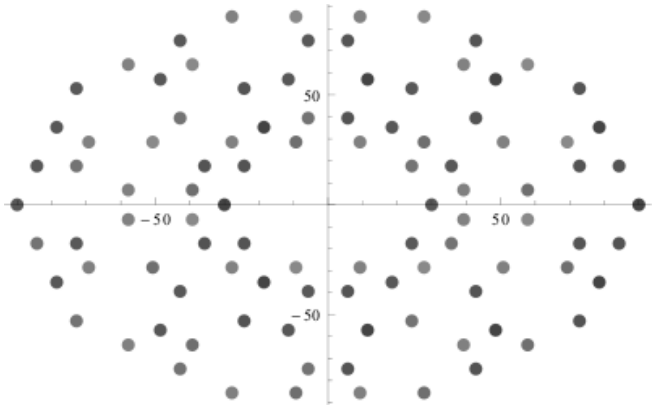


Fig. 4.20 Example of last last layer layout in the replicated output configuration.

4.7 Concluding remarks

At the end of this chapter we cannot conclude with breaking news. Our algorithms work well, but not optimally. The final layouts of the considered neural networks suggest some intriguing features, but do not show anything clear. The last contrivances we adopt to improve the training procedure are promising, but need further refinements. Thus we may sell at this moment only the philosophy, which we synthesize, in a very abstract way, in two points:

- we manage motion as a physical counterpart of agent intentionality in a unified space, where both cognitive forces and physical forces merge in a homogeneous milieu. This appear a natural consequence of a mechanistic perspective where thoughts are the output of neural circuits and the latter obey to the physical laws of Nature. Of course, we are not claiming our Newtonian force fields to be the ones actually ruling the intentionality. We simply assume them to may be a very simplistic modeling of the physical-cognitive interaction.
- we revive the original Artificial Neural Network project, where a single, possibly complex, neural device is able to face any kind of learning task, like our brain does. Also in this case we carry out an extreme simplification of the matter. However, we did so to solve two far different problems, as for goals, features and dimensionality, with a same architecture and same training algorithm. The unique differences concern the input and output layer layouts – which are determined by the training set features – and a few tuning parameters, whose values need a relatively limited set of numerical experiments for their assessment.

As a matter of fact, the contrivance we raised up is definitely complex, and we expect a long work is still necessary to get ride of its behavior. We just hope having thrown a stone into the water and wait for a ripple reaching other researchers in the field.

CLASSIFIER	PREPROC.	MISS%
linear classifier (1-layer NN)	none	12
boosted stumps	none	7.7
K-nearest-neighbors, Euclidean (L2)	none	5
2-layer NN, 300 hidden units, mean square error	none	4.7
2-layer NN, 1000 hidden units	none	4.5
2-layer NN, 1000 HU, [distortions]	none	3.8
1000 RBF + linear classifier	none	3.6
2-layer NN, 300 HU, MSE, [distortions]	none	3.6
40 PCA + quadratic classifier	none	3.3
K-nearest-neighbors, Euclidean (L2)	none	3.09
3-layer NN, 300+100 hidden units	none	3.05
3-layer NN, 500+150 hidden units	none	2.95
K-nearest-neighbors, L3	none	2.83
Mobneu, 120+80+36 hidden units	**none**	**2.76**
3-layer NN, 300+100 HU [distortions]	none	2.5
3-layer NN, 500+150 HU [distortions]	none	2.45
2-layer NN, 800 HU, Cross-Entropy Loss	none	1.6
3-layer NN, 500+300 HU, softmax, cross entropy, weight decay	none	1.53
boosted trees (17 leaves)	none	1.53
SVM, Gaussian Kernel	none	1.4
products of boosted stumps (3 terms)	none	1.26
2-layer NN, 800 HU, cross-entropy [affine distortions]	none	1.1
Convolutional net LeNet-4	none	1.1
Convolutional net LeNet-4 with K-NN instead of last layer	none	1.1
Convolutional net LeNet-4 with local learning instead of last layer	none	1.1
NN, 784-500-500-2000-30 + nearest neighbor, RBM + NCA training [no distortions]	none	1
Convolutional net LeNet-5, [no distortions]	none	0.95
2-layer NN, 800 HU, MSE [elastic distortions]	none	0.9
large conv. net, random features [no distortions]	none	0.89
Convolutional net LeNet-5, [huge distortions]	none	0.85
Trainable feature extractor + SVMs [no distortions]	none	0.83
Virtual SVM deg-9 poly [distortions]	none	0.8
Convolutional net LeNet-5, [distortions]	none	0.8
2-layer NN, 800 HU, cross-entropy [elastic distortions]	none	0.7
Convolutional net Boosted LeNet-4, [distortions]	none	0.7
Virtual SVM, deg-9 poly, 1-pixel jittered	none	0.68
large conv. net, unsup features [no distortions]	none	0.62
Convolutional net, cross-entropy [affine distortions]	none	0.6
large conv. net, unsup pretraining [no distortions]	none	0.6
unsupervised sparse features + SVM, [no distortions]	none	0.59
Trainable feature extractor + SVMs [elastic distortions]	none	0.56
Trainable feature extractor + SVMs [affine distortions]	none	0.54
large conv. net, unsup pretraining [no distortions]	none	0.53
Convolutional net, cross-entropy [elastic distortions]	none	0.4
large conv. net, unsup pretraining [elastic distortions]	none	0.39
6-layer NN 784-2500-2000-1500-1000-500-10 (on GPU) [elastic distortions]	none	0.35

Table 4.6 Comparison between several methodologies applied on MNIST dataset [77]. All the techniques do not use preprocessed data.

Chapter 5
Conclusions

In this thesis work we state a bridge between two kind of forces ruling the motion of natural agents: cognitive forces expressed by the intentionality of the agents and physical forces entailed by their masses. This bridge is at the basis of a mobility model that we may recognize in Nature in humans, but also in animals and in other scenarios like in the morphogenesis of the brain cortex.

Abandoning the symmetry of the Brownian motion – the template of non intelligent agent motion – during this thesis work we have been drawn to understand the general mechanism behind this duality between physics and intentionality.

First of all, we take under observation the social communities in terms of mobility models. These models represented a challenge both for their exploitation in many social contexts and for the sophistication of the theoretical tools required by their analysis and identification. To this aim, we introduce an atomic wait and chase motion of the single agents, essentially following a heuristic approach. However, we had in mind and tried to relate methods and results to two very sophisticated theoretical paradigms represented by the Lévy flights and the Palm Calculus. Thus we dealt with an ergodic process that we followed along the single trajectories drawn by the dodgem cars in the case study, but also along real trajectories emerging from operational scenarios such as those of opportunistic networks, GPS tracking, and so on. To describe these real data we had to abandon the canonical statistical framework in favor of a more ductile methodology coming from Algorithmic Inference. In this way we were able to quickly infer the parameters of the mobility model as a necessary step to validate its suitability. Actually we were successful even in the case of observations that are not mutually independent, as we may expect from the behavior of humans rather than mechanical particles.

Then we moved our mobility model, and related intentional processes, form the macro scale scenarios to the micro scale ones. The subject changes but the overall behavior does not: biological neurons that moves separately from one place to another in

order to constitute the brain. Inspired by this behavior we tried to exploit it to design a machinery that could benefit by this kind of intentionality, so that we build an artificial neural network where each neuron has additional degrees of freedom deriving on a new element: its mobility. These entails a new kind of connectionist paradigm whose exploration is at a very preliminary stage. We may observe that the multilayer networks we handle do not suffer from the typical deep architecture network drawback of getting stuck in local minima around the average output during the training. Tossing these networks on canonical benchmarks we cannot match the top results shown in the literature. However, we work with a robust connectionist tool that requires no exceedingly sophisticated tuning in correspondence of the specific computational tasks and, in any case, no extra expedients generally devised by competitor methods. The neuron motion has a *social value*: each neuron try to find its best reciprocal position w.r.t. the others in terms of information transmission and production efficiency. This emerge a layout adaptivity which seems to revive the initial connectionist project of one architecture for a vast variety of learning tasks. A changing layout is more difficult to mould than a static one. Thus, at this early stage of the research, the question whether the additional degrees of freedom represent a benefit or a drawback. Preliminary experiments lean towards the first option because of the emerging self-organizing attitude of the network. The current advantage is that we get rid of complex networks, as for number of both layers and neurons, with a suitable behavior and acceptable results.

To give a bit pretentious conclusion to this thesis work, and delineate a future research line as well, we may observe the following. A milestone connection between cognitive and physical aspects of thinking agents life has been stated in the past century by the Boltzmann constant numerically connecting thermodynamic entropy to Shannon entropy. Our idea is that this *materialization* of thought could nowadays continue with an analogous numerical connection between the various kind of forces, from physical to cognitive, ruling the motion of the thinking agents. We also remark that in this thesis work we started with a model of the physical motion of the agents, and come to an overall model where the distinction between motion and learning is only a matter of different coordinate spaces where the agent moves.

Appendix A
Derivation rules for the back-propagation algorithm

A.1 Potential derivatives

Let us determine the sensibility of the potential $\mathscr{P}$ w.r.t. the synaptic weights

$$\frac{\partial \mathscr{P}}{\partial w_{ji}} = \sum_{k=1}^{3} \xi_k \frac{\partial P_k}{\partial w_{ji}} \tag{A.1}$$

A.1.1 Gravitational attraction P_1

Let be the following derivation chain rule:

$$\frac{\partial P_1}{\partial w_{ji}} = \frac{\partial P_1}{\partial m_i}\frac{\partial m_i}{\partial \delta_i}\frac{\partial \delta_i}{\partial w_{ji}} + \frac{\partial P_1}{\partial m_j}\frac{\partial m_j}{\partial \delta_j}\frac{\partial \delta_j}{\partial w_{ji}} \tag{A.2}$$

- The first addend of the sum is the product of m_j times $\text{sign}(\delta_i)\frac{\|\delta\|_1-\delta_i}{\|\delta\|_1^2}$ that we can write as $\frac{\text{sign}(\delta_1)}{\|\delta\|_1}(1-m_i)$. Recalling the definition of δ_i in (4.5) we say that even not expressively stated we will not consider the contributions of the neurons of the output layer.
- The second addend does not contribute to the global variations due to $\frac{\partial \delta_j}{\partial w_{ji}} = 0$.

A.1.2 Elastik repulsion P_2

Starting with the following derivation chain rule

$$\frac{\partial P_2}{\partial w_{ji}} = \sum_{i'} \frac{\partial P_2}{\partial k_{ii'}} \frac{\partial k_{ii'}}{\partial w_{ji}} + \sum \frac{\partial P_2}{\partial d_{ii'}} \frac{\partial d_{ii'}}{\partial w_{ji}} \tag{A.3}$$

- We have

$$\frac{\partial P_2}{\partial k_{ii'}} = \frac{1}{2} \max\{0, l - d_{ii'}\}^2 \tag{A.4}$$

whereas for the second factor $\frac{\partial k_{ii'}}{\partial w_{ji}}$:

$$\begin{aligned} \frac{\partial k_{ii'}}{\partial w_{ji}} &= \operatorname{sign}\left(\frac{\langle \mathbf{w}_i, \mathbf{w}_{i'} \rangle}{\|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|} \right) \frac{\frac{\partial \langle \mathbf{w}_i, \mathbf{w}_{i'} \rangle}{\partial w_{ji}} \|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\| - \langle \mathbf{w}_i, \mathbf{w}_{i'} \rangle \frac{\partial \|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|}{\partial w_{ji}}}{(\|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|)^2} \\ &= \operatorname{sign}\left(\frac{\langle \mathbf{w}_i, \mathbf{w}_{i'} \rangle}{\|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|} \right) \frac{w_{ji'} \|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\| - \langle \mathbf{w}_i, \mathbf{w}_{i'} \rangle w_{ji} \frac{\|\mathbf{w}_{i'}\|}{\mathbf{w}_i}}{(\|\mathbf{w}_i\| \cdot \|\mathbf{w}_{i'}\|)^2} \end{aligned} \tag{A.5}$$

- For the second addend we have

$$\frac{\partial P_2}{\partial d_{ii'}} = -k_{ii'} \max\{0, l - d_{ii'}\} \tag{A.6}$$

- and

$$\frac{\partial d_{ii'}}{\partial w_{ji}} = \frac{\partial d_{ii'}}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial w_{ji}} + \frac{\partial d_{ii'}}{\partial \mathbf{x}_{i'}} \frac{\partial \mathbf{x}_{i'}}{\partial w_{ji}} = -\frac{\mathbf{x}_{i'} - \mathbf{x}_i}{d_{ii'}} \frac{\partial \mathbf{x}_i}{\partial w_{ji}} \tag{A.7}$$

Going deeper inside the last equation, we need to consider the contributions from all the components of the vector $\mathbf{x}_i = \{x_{i1}, \ldots, x_{id}\}$, with $d = 2$:

$$\frac{\partial d_{ii'}}{\partial w_{ji}} = \sum_{p=1}^{d} \frac{\partial d_{ii'}}{\partial x_{ip}} \frac{\partial x_{ip}}{\partial w_{ji}} + \frac{\partial d_{ii'}}{\partial x_{i'p}} \frac{\partial x_{i'p}}{\partial w_{ji}} \tag{A.8}$$

In order to lighten the notation we rewrite the latter as

$$\frac{\partial d_{ii'}}{\partial w_{ji}} = \frac{\partial d_{ii'}}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial w_{ji}} + \frac{\partial d_{ii'}}{\partial \mathbf{x}_{i'}} \frac{\partial \mathbf{x}_{i'}}{\partial w_{ji}} \tag{A.9}$$

It is worth noting that the motion inside the layer ℓ and indexed with i remove the sensibility of $x_{i'}$ with reference to w_{ji}. Instead, for the first addend we have

$$\frac{\partial d_{ii'}}{\partial x_i} = -\frac{x_{i'} - x_i}{\|x_{i'} - x_i\|} = -\frac{x_{i'} - x_i}{d_{ii'}}$$

on the one side; on the other side we can exploit (4.11-4.13) as follows:

$$\frac{\partial \mathbf{x}_i^n}{\partial w_{ji}} = \frac{\partial \mathbf{x}_i^{n-1}}{\partial w_{ji}} + \frac{\partial \mathbf{v}_i^n t_n}{\partial w_{ji}} = \frac{\partial \mathbf{x}_i^{n-1}}{\partial w_{ji}} + t_n \frac{\partial \mathbf{v}_i^n}{\partial w_{ji}} + \mathbf{v}^n \frac{\partial t_n}{\partial w_{ji}} = \frac{\partial x_i^{n-1}}{\partial w_{ji}} + t_n \frac{\partial a_i^n}{\partial w_{ji}} \tag{A.10}$$

$$\frac{\partial \mathbf{v}_i^n}{\partial w_{ji}} = \frac{\partial \mathbf{v}_i^{n-1}}{\partial w_{ji}} \frac{\partial \mathbf{a}_i^n t_n}{\partial w_{ji}} = t_n \frac{\partial \mathbf{a}_i^n}{\partial w_{ji}} + \mathbf{a}_i^n \frac{\partial t_n}{\partial w_{ji}} = \frac{\partial \mathbf{v}_i^{n-1}}{\partial w_{ji}} + t_n \frac{\partial \mathbf{a}_i^n}{\partial w_{ji}} \tag{A.11}$$

Summarizing

$$\frac{\partial \mathbf{x}_i^n}{\partial w_{ji}} = \frac{\partial \mathbf{x}_i^{n-1}}{\partial w_{ji}} + t_n \left(\frac{\partial \mathbf{v}_i^{n-1}}{\partial w_{ji}} + t_n \frac{\partial \mathbf{a}_i^n}{\partial w_{ji}} \right) \tag{A.12}$$

It remains to calculate the term $\frac{\partial a_i^n}{\partial w_{ji}}$ that we will explicit in Section A.2. Finally we can interpret

$$\frac{\partial d_{ii'}}{\partial w_{ji}} = -\frac{\mathbf{x}_{i'} - \mathbf{x}_i}{d_{ii'}} \frac{\partial \mathbf{x}_i}{\partial w_{ji}} \tag{A.13}$$

as

$$\frac{\partial d_{ii'}}{\partial w_{ji}} = -\frac{1}{d_{ii'}} \sum_{p=1}^{d} (x_{i'p} - x_{ip}) \frac{\partial \mathbf{x}_i}{\partial w_{ji}} \tag{A.14}$$

Kinetic energy P_3

From the usual derivation chain rule we have

$$\frac{\partial P_3}{\partial w_{ji}} = \frac{\partial P_3}{\partial m_i} \frac{\partial m_i}{\partial w_{ji}} + \frac{\partial P_3}{\partial \mathbf{v}_i} \frac{\partial \mathbf{v}_i}{\partial w_{ji}} \tag{A.15}$$

- For the first added we have $\frac{1}{2}\|\mathbf{v}_i\|^2$ multiplied by $\frac{\partial m_i}{\partial w_{ji}}$ as in Subsection A.1.1.
- For the second one we have $m_i \mathbf{v}_i$ multiplied by $\frac{\partial \mathbf{v}_i}{\partial w_{ji}}$. The computation of $\frac{\partial P_3}{\partial w_{ji}}$ is done component-wise in the vector $\mathbf{v}_i$.

A.2 Computation of the acceleration derivative

From (4.15) we decompose the acceleration in its three components:

$$\mathbf{a}_i = \sum_{k=1}^{3} \xi_k \mathbf{a}_i^k \tag{A.16}$$

and we derive the following derivation chain rules:

$$\frac{\partial \mathbf{a}_i^1}{\partial w_{ji}} = \frac{\partial \mathbf{a}_i^1}{\partial m_j}\frac{\partial m_j}{\partial w_{ji}} + \frac{\partial \mathbf{a}_i^1}{\partial \mathbf{x}_i}\frac{\partial \mathbf{x}_i}{\partial w_{ji}} + \frac{\partial \mathbf{a}_i^1}{\partial \mathbf{x}_j}\frac{\partial \mathbf{x}_j}{\partial w_{ji}} \tag{A.17}$$

$$\frac{\partial \mathbf{a}_i^2}{\partial w_{ji}} = \sum_{i'} \frac{\partial \mathbf{a}_i^2}{\partial k_{ii'}}\frac{\partial k_{ii'}}{\partial w_{ji}} + \frac{\partial \mathbf{a}_i^2}{\partial d_{ii'}}\frac{\partial d_{ii'}}{\partial w_{ji}} + \frac{\partial \mathbf{a}_i^2}{\partial \mathbf{x}_{i'}}\frac{\partial x_{i'}}{\partial w_{ji}} \tag{A.18}$$

$$\frac{\partial \mathbf{a}_i^3}{\partial w_{ji}} \quad \text{momentum term w.r.t. the time } t_{n-1} \tag{A.19}$$

In particular we have that:

$\mathbf{a}_i^1$ No addends here are dependent from the synaptic weights because of:

- m_j does not depend on w_{ji}
- the derivative of the *s*ign function is null
- $\mathbf{a}_i$s are not dependent with respect to the positions $\mathbf{x}_j$ of the upper-layer neurons.

$\mathbf{a}_i^2$ For $d_{ii'} \geq l$ the first addend is null, the third and the fourth are always null thus we obtain:

$$\frac{\partial \mathbf{a}_i^2}{\partial w_{ji}} = \sum_{i'} (\max\{0, l - d_{ii'}\}) \operatorname{sign}(\mathbf{x}_{i'} - \mathbf{x}_i) \frac{\partial k_{ii'}}{\partial w_{ji}} + \\ - \sum_{i'} k_{ii'} \operatorname{sign}(\mathbf{x}_{i'} - \mathbf{x}_i) \frac{\partial d_{ii'}}{\partial w_{ji}} \tag{A.20}$$

where the two derivatives are computed according to (A.5) and (A.7).

A.3 Error derivatives

For the sake of simplicity we will use the subscript o to indicate those neurons belonging to the output layer. Let us recall the error formulation:

$$E = \frac{1}{2}\sum_o (\tau_o - z_o)^2 \tag{A.21}$$

where z_o is the o-th target w.r.t. the o-th neuron.
Here we are interested in $\frac{\partial E}{\partial w_{ji}}$ that we calculate through the usual back-propagation rule on the δs defined in (4.5). However, due to the introduction of an additional term λ inside the *net* formula, we have to recompute the derivatives as follows:

- on the output neurons:

$$\begin{aligned}\frac{\partial E}{\partial w_{oi}} &= (\tau_o - z_o) f'(\text{net}_o)\frac{\partial \text{net}_o}{\partial w_{oi}} = \delta_o\left(\frac{\partial \text{net}_o}{\partial w_{oi}} + \frac{\partial \text{net}_o}{\partial \lambda_{oi}}\frac{\partial \lambda_{oi}}{\partial w_{oi}}\right)\\ &= \delta_o\left(\lambda_{oil}\tau_o - \tau_o w_{oi}\frac{\partial d_{oi}}{\partial w_{oi}}\right) = \left(1 - w_{oi}\frac{\partial d_{oi}}{\partial w_{oi}}\right)\lambda_{oi}\tau_o\delta_o\end{aligned} \tag{A.22}$$

- on the hidden layer neurons the derivatives are:

$$\begin{aligned}\frac{\partial E}{\partial w_{ji}} &= \sum_o \delta_o \frac{\partial net_o}{\partial w_{ji}} = \sum_o \delta_o \frac{\partial \text{net}_o}{\partial \tau_j}\frac{\partial \tau_j}{\partial \text{net}_j}\left(\frac{\partial \text{net}_j}{\partial w_{ji}} + \frac{\partial \text{net}_j}{\partial \lambda_{ji}}\frac{\partial \lambda_{ji}}{\partial w_{ji}}\right)\\ &= f'(\text{net}_j)\sum_o w_{oj}\delta_o\left(\lambda_{ji}\tau_i - w_{ji}\tau_i\frac{\partial d_{ji}}{\partial w_{ji}}\right) = \left(1 - w_{ji}\frac{\partial d_{ji}}{\partial w_{ji}}\right)\lambda_{ji}\tau_i\delta_j\end{aligned} \tag{A.23}$$

A.4 Thresholds

The introduction of the thresholds θ_j into the network state function

$$\text{net}_j = \sum_i w_{ji}\lambda_{ji}\tau_i + \theta_j \tag{A.24}$$

can be interpreted as an additional fictitious neuron whose output is constantly equal to 1 and whose connections toward the $\ell + 1$ layer are defined by the θ_j parameter. However, this neuron has no specific position in the plane; actually we may imagine it located at infinity so as to have cosntant λ and no derivative in the *Hamiltonian* apart the error term *error term*:

$$\frac{\partial E}{\partial \theta_j} = \delta_j \tag{A.25}$$

References

1. J. Abello, A. L. Buchsbaum, and J. Westbrook. A functional approach to external graph algorithms. In *Proceedings of the 6th European Symposium on Algorithms*, pages 332–343, Berlin, 1998. Springer.
2. W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 171–180, New York, 2000. Association of Computing Machinery.
3. H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.
4. R. Albert and A. L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.
5. B. Apolloni, G. Apolloni, S. Bassis, G. L. Galliani, and G. P. Rossi. Collaboration at the basis of sharing focused information: The opportunistic networks. In *Computational Intelligence: Collaboration, Fusion and Emergence*, Intelligent Systems Reference Library Part VI, pages 501–524. Springer, Berlin, Heidelberg, 2009.
6. B. Apolloni and S. Bassis. Algorithmic inference of two-parameter gamma distribution. *Communications in Statistics - Simulation and Computation*, 38:1–19, 2009.
7. B. Apolloni, S. Bassis, S. Gaito, and D. Malchiodi. Appreciation of medical treatments by learning underlying functions with good confidence. *Current Pharmaceutical Design*, 13(15):1545–1570, 2007.
8. B. Apolloni, S. Bassis, S. Gaito, and D. Malchiodi. Bootstrapping complex functions. *Nonlinear Analysis: Hybvrid Systems*, 2(2):665–683, 2008.
9. B. Apolloni, S. Bassis, D. Malchiodi, and P. Witold. *The Puzzle of Granular Computing*, volume 138 of *Studies in Computational Intelligence*. Springer Verlag, 2008.
10. B. Apolloni, S. Bassis, and A. Zippo. Processing of information microgranules within an individuals' society. *Human-centric Information Processing Through Granular Modelling*, 182:233–264, 2009.
11. B. Apolloni, G. L. Galliani, F. Giudici, S. Irti, and G. P. Rossi. CARTOON: Context aware routing over opportunistic networks data set (v. 2008-10-31). Downloaded from http://nptlab.dico.unimi.it/index.php/cartoon.html, oct 2008.
12. B. Apolloni, D. Malchiodi, and S. Gaito. *Algorithmic Inference in Machine Learning*. Advanced Knowledge International, Magill, Adelaide, 2nd edition, 2006.
13. Bruno Apolloni, Simone Bassis, Elena Pagani, Gian Paolo Rossi, and Lorenzo Valerio. A mobility timing for agent communities, a cue for advanced connectionist systems. *IEEE Trans. on Neural Networks*, In press.
14. F. Baccelli and P. Bremaud. *Elements of queueing theory, Palm Martingale Calculus and Stochastic Recurrences*, volume 26 of *Stochastic Modelling and Applied Probability*. Springer, 2nd edition, 2003.
15. Mark Bear, Barry Connors, Michael Paradiso, Mark F. Bear, Barry W. Connors, and Michael A. Paradiso. *Neuroscience: Exploring the Brain (Book with CD-ROM)*. Lippincott Williams & Wilkins, second edition, March 2002.
16. Anthony J. Bell and Terrence J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.
17. Y. Bengio and X. Glorot. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of AISTATS 2010*, pages pp. 249–256, 2010,.
18. Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, Université De Montréal, and Montréal Québec. Greedy layer-wise training of deep networks. In *In NIPS*. MIT Press, 2007.

19. S. Borgatti and Everett. M. Ecological and perfect colorings. *Social Networks*, 16:43–55, 1994.
20. M. Bortman and M. Aladjem. A growing and pruning method for radial basis function networks. *IEEE Trans. on Neural Networks*, 20(6):1039–1045, 2009.
21. J. P. Bouchaud. More Lévy distributions in physics. In & U. Frisch M. F. Shlesinger, G. M. Zaslavsky, editor, *Lévy Flights and Related Topics in Physics*, volume 450 of *Lecture Notes in Physics, Berlin Springer Verlag*, pages 237–250, 1995.
22. F. Brauer and C. Castillo-Chavez. *Mathematical Models in Population Biology and Epidemiology*, volume 40 of *Texts in Applied Mathematics*. Springer, 2001.
23. R. Breiger, K. Carley, and P. Pattison, editors. *Dynamic Social Network Modelling and Analysis*, Washington, 2003. The National Academies Press.
24. D. D. Brockmann, L. Hufnagel, and T. Geisel. The scaling laws of human travel. *Nature*, 439:462–465, 2006.
25. A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer Networks*, 33:309–320, 2000.
26. H. Cai and D. Y. Eun. Crossing over the bounded domain: From exponential to power-law intermeeting time in MANET. In *ACM MobiCom 2007*, pages 1578–1591, Montreal, Canada, 2007.
27. K. M. Carley, J. Diesner, J. Reminga, and M. Tsvetovat. Toward an interoperable dynamic network analysis toolkit. *Decision Support Systems*, 43(4):1324–1347, 2007.
28. G. A. Carpenter and S. Grossberg. Adaptive resonance theory. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 87–90. MIT Press, Cambridge, MA, 2nd edition, 2003.
29. A. Chaintreau, P. Hui, J. Crowcroft, C Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.
30. A. S. Chaves. A fractional diffusion equation to describe Lévy flights. *Physics Letters A*, 239(1-2):13–16, 1998.
31. D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep big simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, 2010.
32. V. Colizza, A. Barrat, M. Barthelemy, A.-J. Valleron, and A. Vespignani. Modeling the worldwide spread of pandemic influenza: baseline case and containment interventions. *PLoS Medicine*, 4:95–110, 2007.
33. V. Conan, J. Leguay, and T. Friedman. Characterizing pairwise inter-contact patterns in delay tolerant networks. In *Autonomics '07: Proceedings of the 1st international conference on Autonomic computing and communication systems*, pages 1–9, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
34. P. I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, 1996.
35. S. Csorgo and J. J. Faraway. The exact and asymptotic distributions of Cramer-von Mises statistics. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):221–234, 1996.
36. Yann Le Cun. A theoretical framework for back-propagation, 1988.
37. Patrick F. D. *Measurement and Data Analysis for Engineering and Science*. McGraw–Hill, New York, 2005.
38. Somayeh Danafar, Arthur Gretton, and Jürgen Schmidhuber. Characteristic kernels on structured domains excel in robotics and human action recognition. In José L. Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *ECML/PKDD (1)*, volume 6321 of *Lecture Notes in Computer Science*, pages 264–279. Springer, 2010.
39. D. J. de Solla Price. Networks of scientific papers. *Science*, 149(3683):510–515, 1965.

40. B. Drossel and F. Schwabl. Self-organized critical forest-fire model. *Physcal Review Letter*, 69(11):1629–1632, 1992.
41. R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2001.
42. J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.
43. D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, Cambridge, MA, USA, 2010.
44. A. M. Edwards et al. Revisiting Lévy flight search patterns of wandering albatrosses, bumblebees and deer. *Nature*, 449:1044–1049, 2007.
45. B. Efron and R. Tibshirani. *An introduction to the Boostrap*. Chapman and Hall, Freeman, New York, 1993.
46. A. Einstein. On the motion, required by the molecular-kinetic theory of heat, of particles suspended in a fluid at rest. *Annals of Physics*, 17:549–560, 1905.
47. R. Elitzur and W Anthony. Game theory as a tool for understanding information services outsourcing. *Journal of Information Technology*, 12(1):45–60, 1997.
48. D. Erhan, P. A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *AISTATS 2009*, pages 153–160, Clearwater (Florida), USA, 2009.
49. Facebook. http://www.facebook.com/.
50. Enrico Fermi. *Thermodynamics*. Dover, 1956.
51. A. Fischer and C. Igel. Empirical analysis of the divergence of Gibbs sampling based learning algorithms for restricted Boltzmann machines. In K. Diamantaras, W. Duch, and L. S. Iliadis, editors, *International Conference on Artificial Neural Networks (ICANN 2010)*, volume 6354 of *LNCS*, pages 208–217. Springer-Verlag, 2010.
52. M. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Trans. of the Royal Society of London Ser. A*, 222:309–368, 1925.
53. S. Gaito, E. Pagani, and G. P. Rossi. Fine-grained tracking of human mobility in dense scenarios. In *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON) - Poster Session, June 22-26 2009*, pages 40–42, 2009.
54. S. Geirhofer, L. Tong, and B. Sadler. A measurement-based model for dynamic spectrum access in WLAN channels. *MILCOM*, 0:1–7, 2006.
55. B. V. Gnedenko and A. N. Kolmogorov. *Limit distributions for sums of independent random variables*. Translated from the Russian, annotated, and revised by K. L. Chung. With appendices by J. L. Doob and P. L. Hsu. Revised edition. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills., Ont., 1968.
56. M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453:779–782, 2008.
57. Sanjay Gupta and R. K. P. Zia. Quantum neural networks. *J. Comput. Syst. Sci.*, 63(3):355–383, 2001.
58. P. Hall and A. H. Welsh. Limit theorems for the median deviation. *Annals of the Institute of Statistical Mathematics*, 37(1):27–36, 1985.
59. U. Hansmann. *Pervasive Computing: The Mobile World*. Springer, 2003.
60. Carl M. Harris. The pareto distribution as a queue service discipline. *Operations Research*, 16(2):307–313, 1968.
61. J. A. Hartigan. *Clustering Algorithms*. Wiley, 1975.
62. Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2011/11/01 2006.

63. S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: The difficulty of learning long-term dependencies. In S. C. Kremer and J. F. Kolen, editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.
64. P. Holme, J. Karlin, and S. Forrest. Radial structure of the internet. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463:1231–1246, 2007.
65. S. Hong, I. Rhee, S. J. Kim, K. Lee, and S. Chong. Routing performance analysis of human-driven delay tolerant networks using the truncated Lévy walk model. In *ACM SIGMOBILE International Workshop on Mobility Models for Networking Research (Colocated with MobiHoc 08)*, 2008.
66. P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and the consequence of human mobility in conference environments. In *ACM WDTM*, pages 244–251, Philadelphia, 2005.
67. D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, volume 353, pages 153–181. Kluwer Academic Pub., 1996.
68. M. Kac. Random walk and the theory of Brownian motion. *The American Mathematical Monthly*, 54(7):369–391, 1947.
69. Y. Y. Kagan and F. Schoenberg. Estimation of the upper cutoff parameter for the tapered Pareto distribution. *Journal of Applied Probability*, 38:158–175, 2001.
70. T. Karagiannis, J.Y. Le Boudec, and M. Vojnovic. Power law and exponential decay of inter contact times between mobile devices. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 183–194, 2007.
71. J. Klafter, M. F. Shlesinger, and G. Zumofen. Beyond Brownian motion. *Physics Today*, 49:33–39, 1996.
72. K. Kreutz-Delgado and B. D. Rao. Application of concave/Schur-concave functions to the learning of overcomplete dictionaries and sparse representations. *Signals, Systems & Computers, 1998. Thirty-Second Asilomar Conference*, 1:546–550, 1998.
73. J. F. Kurose and K. W. Ross. *Computer Networking – A Top-Down Approach*. Pearson Addison Wesley, 4th edition, 2008.
74. J. Lamperti. *Stochastic processes: a survey of the mathematical theory*, volume 23 of *Applied mathematical sciences*. Springer-Verlag, New York, 1977.
75. H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, 10:1–40, 2009.
76. J.-Y. Le Boudec and M. Vojnovic. Perfect simulation and stationarity of a class of mobility models. In *INFOCOM 2005. Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2743–2754, 2005.
77. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86(11), pages 2278–2324, 1998.
78. Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *ICML*, volume 382 of *ACM International Conference Proceeding Series*, page 77. ACM, 2009.
79. J. Lerner. Role assignments. In *Network Analysis, Methodological Foundations*, volume 3418/2005 of *Lecture Notes in Computer Science*, chapter 9, pages 216–252. Springer, Berlin / Heidelberg, 2005.
80. I. Levner. *Data Driven Object Segmentation*. PhD thesis, Department of Computer Science, University of Alberta, 2008.
81. LinkedIn. http://www.linkedin.com/.
82. R. Y. Liu, J. M. Parelius, and K. Singh. Multivariate analysis by data depth: Descriptive statistics, graphics and inference. *The Annals of Statistics*, 27:783–858, 1999.

83. M. W. Macy, Kitts J. A., and A. Flache. *Polarization in Dynamic Networks: A Hopfield Model of Emergent Structure*. National Academies, Washington, 2003.
84. H. Mahmound, R. Smythe, and J. Szymansky. On the structure of plane-oriented recursive trees and their branches. *Random structures and Algorithms*, 3:255–266, 1993.
85. Y. Malevergne, V. Pisarenko, and D. Sornette. Empirical distributions of stock returns: Exponential or power-like? *Quantitative Finance*, 5:379–401, 2005.
86. B. B. Mandelbrot. An information theory of the statistical structure of language. *Communication Theory*, pages 503–512, 1953.
87. B. B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman, 1982.
88. O. Marín and G. Lopez-Bendito. Neuronal migration. In J. H. Kaas, editor, *Evolution of Nervous Systems, Four-Volume Set: A Comprehensive Reference*, chapter 1.1. Academic Press, 1st edition, 2007.
89. A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on enron and academic email source. *Journal of Artificial Intelligence Research archive*, 30(1):249–272, 2007.
90. R. Metzler. The random walk's guide to anomalous diffusion: a fractional dynamics approach. *Physics Reports*, 339(1):1–77, 2000.
91. M. Mitzenmacher. Dynamic models for file sizes and double Pareto distributions. *Internet Mathematics*, 1(3):305–333, 2004.
92. Andriy Mnih and Geoffrey E. Hinton. A scalable hierarchical distributed language model. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS*, pages 1081–1088. MIT Press, 2008.
93. H. Mori. Transport, collective motion, and Brownian motion. *Progress of Theoretical Physics*, 33(3):423–455, 1965.
94. M. Musolesi and C. Mascolo. A community based mobility model for ad hoc network research. In *REALMAN '06: Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 31–38, New York, NY, USA, 2006. ACM.
95. S. F. Nadel. *The Theory of Social Structure*. Cohen and West, London, 1957.
96. M. E. J. Newman. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46:323–351, 2005.
97. T. D. Newton and E. P. Wigner. Localized states for elementary systems. *Reviews of Modern Physics*, 21(3):400, 1949.
98. Nokia. Nokia sports tracker, 2009. `http://sportstracker.nokia.com/`.
99. S. Nolfi and D. Parisi. *Handbook of brain theory and neural networks, Second Edition*, chapter Evolution of artificial neural networks, pages 418–421. Cambridge, MA: MIT Press, 2002.
100. Simon Osindero and Geoffrey E. Hinton. Modeling image patches with a directed hierarchy of markov random fields. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *NIPS*. MIT Press, 2007.
101. K. T. Ozan and G. Ferrari. *Ad Hoc Wireless Networks: A Communication-Theoretic Perspective*. John Wiley & Sons, 2006.
102. V. Pareto. *Course d'Economie Politique*. Rouge and Cie, Lausanne and Paris, 1897.
103. M. Piórkowski. Sampling urban mobility through on-line repositories of GPS tracks. In *HotPlanet '09: Proceedings of the 1st ACM International Workshop on Hot Topics of Planet-Scale Mobility Measurements*, pages 1–6, New York, NY, USA, 2009. ACM.
104. V. F. Pisarenko and D. Sornette. Characterization of the frequency of extreme earthquake events by the generalized Pareto distribution. *Pure and Applied Geophysics*, 160(12):2343–2364, 2003.
105. A. R. Radcliffe-Brown. On social structure. *Journal of the Royal Anthropological Institute*, 70:1–12, 1940.

106. G. Ramos-Fernandez and et al. Lévy walk patterns in the foraging movements of spider monkeys. *Behav. Ecol. Sociobiol.*, 273:1743–1750, 2004.
107. Marc'Aurelio Ranzato, Y-Lan Boureau, Sumit Chopra, and Yann LeCun. A unified energy-based framework for unsupervised learning. *Journal of Machine Learning Research - Proceedings Track*, 2:371–379, 2007.
108. C. E. Rasmussen, R. M. Neal, G. E. Hinton, D. van Camp, M. Revow, Z. Ghaharamani, R. Kustra, and R. Tibshirani. The delve manual. Technical report, Department of Computer Science, University of Toronto, Canada, December 1996. Version 1.1.
109. W. J. Reed. The Pareto, Zipf and other power laws. *Economics Letters*, 74(1):15–19, 2001.
110. W. J. Reed. The pareto law of incomes–an explanation and an extension. *Physica A: Statistical Mechanics and its Applications*, 319:469–486, 2003.
111. I Rhee and S. Chong. Human Mobility Model and DTN Group, 2010.
112. I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong. On the Lévy-walk nature of human mobility. In *Proceedings of the 27th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2008.
113. R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princetion, 1970.
114. V. K. Rohatgi. *An Introduction to Probablity Theory and Mathematical Statistics*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York, 1976.
115. Murray Rosenblatt. Remarks on a multivariate transformation. *Annals of Mathematical Statistics*, 23(3), 1952.
116. D. E. Rumelhart, J. L. McClelland, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1: Foundations. MIT Press, Cambridge, MA, 1986.
117. Ruslan Salakhutdinov and Geoffrey E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. *Journal of Machine Learning Research - Proceedings Track*, 2:412–419, 2007.
118. Ruslan Salakhutdinov and Geoffrey E. Hinton. Using deep belief nets to learn covariance kernels for gaussian processes. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *NIPS*. MIT Press, 2007.
119. Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24-th International Conference on Machine Learning*, 2007.
120. E. Schrodinger. *What is Life?* Macmillan, 1945.
121. J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set Cambridge/Haggle (v. 2006-09-15). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, sep 2006.
122. Masashi Sekino and Katsumi Nitta. Automatic model selection via corrected error backpropagation. In Mario Köppen, Nikola K. Kasabov, and George G. Coghill, editors, *ICONIP (2)*, volume 5507 of *Lecture Notes in Computer Science*, pages 220–227. Springer, 2008.
123. M. F. Shlesinger, J. Klafter, and G. Zumofen. Above, below and beyond Brownian motion. *American Journal of Physics*, 67:1253–1259, 1999.
124. V. P. Singh and H. Guo. Parameter estimation for 3-parameter generalized Pareto distribution by the principle of maximum entropy (POME). *Hydrological Sciences*, 40(2), 1995.
125. C. Song, Z. Qu, N. Blumm, and A. L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.
126. S. Stigler. Studies in the history of probability and statistics. XXXII: Laplace, Fisher and the discovery of the concept of sufficiency. *Biometrika*, 60(3):439–445, 1973.
127. M. Surnam and D. Wershler-Henry. *Common Space: Beyond Virtual Community*. Financial Times.com, Canada, 2001.

128. Graham W. Taylor and Geoffrey E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *ICML*, volume 382 of *ACM International Conference Proceeding Series*, page 129. ACM, 2009.
129. C. K. Toh. *Ad Hoc Mobile Wireless Networks*. Prentice Hall Publishers, 2002.
130. Antonio Torralba, Robert Fergus, and Yair Weiss. Small codes and large image databases for recognition. In *CVPR*. IEEE Computer Society, 2008.
131. Kiatichai Treerattanapitak and Chuleerat Jaruskulchai. Membership enhancement with exponential fuzzy clustering for collaborative filtering. In Kok Wai Wong, B. Sumudu U. Mendis, and Abdesselam Bouzerdoum, editors, *ICONIP (1)*, volume 6443 of *Lecture Notes in Computer Science*, pages 559–566. Springer, 2010.
132. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1096–1103. ACM, 2008.
133. M. Webber. Order in diversity: Community without propinquity. In J. Lowdon Wingo, editor, *Cities and Space: The Future Use of Urban Land*, pages 23–54, Baltimore, 1963. Johns Hopkins Press.
134. B. Wellman. The community question: The intimate networks of East Yorkers. *American Journal of Sociology*, 84:1201–1231, 1979.
135. W. Willinger and V. Paxson. Where mathematics meets the internet. *Notices of the American Mathematical Society*, 45:961–970, 1998.
136. L. Xiaochen, W. Mao, D. Zeng, and F. Y. Wange. Agent-based social simulation and modeling in social computing. In *Intelligence and Security Informatics*, volume 5075/2008 of *Lecture Notes in Computer Science*, pages 401,412. Springer, 2008.
137. G. U. Yule. A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis, F.R.S. *Philosophical Trans. of the Royal Society of London, Series B*, 213:21–87, 1925.
138. Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Communications Surveys & Tutorials*, 8(1):24–37, 2006.