Region-based approximation to solve inference in loopy factor graphs : decoding LDPC codes by the Generalized Belief Propagation

Jean-Christophe Sibel





Université de Cergy-Pontoise
École doctorale Sciences & Ingénierie

THÈSE

présentée en vue d'obtenir le grade de

DOCTEUR

spécialité Sciences et Technologies
de l'Information et de la Communication

par

Jean-Christophe Sibel

Region-based approximation to solve inference in loopy factor graphs: decoding LDPC codes by Generalized Belief Propagation

Directeur de thèse	Professeur	David Declercq	ETIS/ENSEA CNRS UMR 8051 Univ. de Cergy-Pontoise
Encadrant de thèse	Docteur	Sylvain Reynal	ETIS/ENSEA CNRS UMR 8051 Univ. de Cergy-Pontoise

Soutenue le 7 juin 2013
devant la Commission d'Examen

Rapporteur	Professeur	Patrick Duvaut	Directeur de recherche, Télécom ParisTech
Rapporteur	Professeur	Charly Poulliat	IRIT, INP-ENSEEIHT Toulouse
Président du jury	Professeur	Brigitte Vallée	Directrice de recherche, Univ. de Caen
Examinateur	Professeur	Bane Vasic	Department of Electrical and Computer Engineering, Univ. d'Arizona, Tucson, USA

Remerciements

Je remercie les personnes qui ont contribué à mon entrée dans le monde de la recherche. Parmi elles, Sylvain Reynal et David Declercq, qui m'ont proposé cette thèse et qui m'ont laissé libre dans mes choix d'axes de recherche, Jean-Pierre Tillich, qui m'a accueilli à l'INRIA ainsi que dans le projet COCQ, et qui a toujours montré de l'intérêt pour mes travaux. Je remercie également les autres membres du laboratoire avec qui j'ai interagi: Frédéric Precioso qui a été très accueillant et de bons conseils, Michel Leclerc et Laurent Protois qui m'ont beaucoup aidé pour que je puisse utiliser au mieux les ressources informatiques, Annick Bertinotti qui malgré les multiples affaires administratives à régler savait prendre du temps pour nous les doctorants. Je remercie Emmanuel Bourdel et Inbar Fijalkow, ainsi que Sylvain Reynal et Michel Leclerc, pour m'avoir donné l'opportunité d'enseigner dans divers domaines, ce que j'ai beaucoup apprécié.

Les doctorants du laboratoire ont également participé, de près ou de loin, à mon travail. Je remercie Antoine Rolland de Rengervé qui savait m'écouter et faire ressortir dans mes problèmes les points importants, Alexis Lechervy qui a passé beaucoup de temps à m'aider à améliorer mes programmes et à structurer mon travail. J'ai une pensée toute particulière pour mes collègues et amis Jean-Emmanuel (au sens de l'humour si personnel mais aigu), Romain Tajan, Gaël Rigaud, et Ludovic Danjean qui ont travaillé avec moi pendant des demi-journées (voire journées) entières sur mes problèmes, qui ont lu, critiqué, corrigé et considérablement enrichi mes articles et mon travail, et qui ont été des oreilles attentives à mes détresses scientifiques. Je remercie aussi mes collègues de bureau Alan Julé pour les déjeuners sympathiques et les gâteaux bretons, Rémi Régnier pour la profonde connaissance et admiration que nous partagions à propos de Kaamelott, ainsi que Liang Zhou, Raouia Masmoudi, Shuji Zhao et Sonia Khatchadourian. Je remercie Erbao Li avec qui je n'ai malheureusement pas eu le temps d'écrire un article, mais qui m'a aidé à développer des idées et concepts, Fouad Sahraoui avec qui j'ai beaucoup parlé de spiritualité mais aussi de la compréhension de l'informatique et ses défis. Je remercie les autres doctorants et docteurs du laboratoire avec qui j'ai passé de bons moments : Laurent Rodriguez, Rodrigue Imad, Babar Aziz, David Gorisse, Thomas Lefebvre, Leïla Meziou, Mathilde Brandon.

En dehors du laboratoire, j'ai été soutenu par mes parents, mes beaux-parents, mes frères et mes belles-soeurs que je remercie tous chaleureusement. Je remercie avec grande joie mes cousins Nathan, Anna, Tom et Maïa avec qui j'ai beaucoup parlé et relativisé les obstacles de la thèse, mes cousines Aurélie et Sarah pour leurs encouragements, et tout spécialement mon cousin Philippe qui, depuis les études supérieures me suit, me conseille et me soutient avec beaucoup d'affection, qui a passé beaucoup de temps avec moi pour discuter des solutions possibles aux impasses que je rencontrais, qui a su lire et critiquer avec précision les écrits que je produisais ainsi que le présent manuscrit. Je remercie mes amis Eric, Julien, Jules, Vanessa, Aurélien(s), Tiphaine, Pierre, Marie-Fanny, Violaine, Benoît, Aurélie, Cédric, Marie-Capucine, Guillaume, Isabelle, Alexandre, Jérémy, Lorine, Brice, Juliette, Lucie, Germain, Maxime, Gabriel et Kevin qui m'ont montré un soutien solide. Je remercie également toutes les personnes qui ont participé à la vidéo des Félicitations (même celles que je ne connais pas) réalisée par Ludovic.

Je termine en remerciant la personne la plus importante : ma femme Julie, qui a fait preuve à la fois de patience et de soutien, qui a accepté mes dérives et mes excès, mais qui a aussi partagé les grandes satisfactions et joies de mon travail, qui a toujours fait montre d'optimisme et d'espoir à mon égard, qui m'a encouragé et félicité. Sans elle, je n'aurais pas réussi, sans elle, je n'aurais même pas essayé...

Short abstract

English version

This thesis addresses the problem of inference in factor graphs, especially LDPC codes, almost solved by message-passing algorithms. The Belief Propagation algorithm (BP) is extensively investigated as a particular message-passing algorithm and it is demonstrated to be equivalent to Bethe approximation in statistical physics of spin glasses. We raise the issue about its subotpimality when applied to factor graphs with loop-like topologies. Afterwards is detailed the Generalized Belief Propagation (GBP), another message-passing algorithm established according to a generalization of the Bethe approximation, specifically the region-based approximation. Based on an non-unique clustering of factor graph, it is experimentally shown that GBP can significantly outperform BP when the clustering deals with harmful topological structures, particularly trapping sets, that prevent BP from rightly decoding. We do not only confront BP and GBP performance using the bit-error rate but also with respect to their dynamical behaviors for non-trivial error events. Using original dynamical quantifiers, it is shown that GBP can overcome BP in terms of accuracy and convergence.

Version française

Dans cette thèse, nous étudions le problème de l'inférence bayésienne dans les graphes factoriels, en particulier les codes LDPC, quasiment résolus par les algorithmes de message-passing. Nous réalisons en particulier une étude approfondie du Belief Propagation (BP), un algorithme de message-passing dont nous démontrons l'équivalence avec l'approximation de Bethe utilisée pour étudier les verres de spins en physique statistique. La question de la sous-optimalité est soulevée dans le cas où le graphe factoriel présente des boucles. Ensuite est détaillé le Generalized Belief Propagation (GBP), un autre algorithme de message-passing élaboré à partir d'une généralisation de l'approximation de Bethe, l'approximation basée régions. Fondé sur un découpage en clusters du graphe factoriel, nous montrons par des expériences que le GBP peut être significativement plus performant que le BP dans les cas où le découpage donne naissance à des clusters très liés au structures topologiques, en particulier les trapping sets, qui empêchent le BP de bien décoder. Nous ne confrontons pas les performances des deux algorithmes uniquement par rapport à leurs taux d'erreur binaire mais aussi par rapport à leurs comportements dynamiques face à des événements d'erreur non triviaux. En utilisant des estimateurs inédits, nous montrons que l'algorithme du GBP peut dominer celui du BP en termes de précision et de convergence.

Résumé

Dans cette thèse, nous nous attaquons aux problèmes liés à l'inférence statistique, en particulier l'approximation des distributions marginales sur les noeuds d'un graphe factoriel à boucles. L'algorithme type message-passing du Belief Propagation (BP) créé par Pearl en 1988 ne parvient pas à approcher ces grandeurs de manière optimale, malgré de très bonnes performances. En effet, dans le cas des codes LDPC, le décodage par le BP présente des oscillations voire des comportements non classiques, comme du chaos. Après avoir présenté le fonctionnement de cet algorithme, nous montrons qu'il est équivalent à l'approximation de Bethe utilisée en physique statistique comme une approche de champ moyen pour estimer la fonction de partition des verres de spins par une minimisation de l'énergie libre variationnelle.

Les problèmes d'oscillations dans les graphes factoriels, dits aussi de frustration pour les verres de spins, peuvent être résolus grâce à l'approximation basée régions qui consiste à créer un réseau bayésien associé, appelé graphe des régions, dont les noeuds, ou régions, sont des regroupements des noeuds qui composent le graphe factoriel. Par ce moyen, il est attendu que la précision de l'approximation soit meilleure que celle donnée par l'approximation de Bethe. Par la même équivalence qui lie cette dernière au BP, nous démontrons que l'approximation basée régions peut être traduite comme un algorithme de message-passing appelé Generalized Belief Propagation (GBP). Les performances de décodage du GBP peuvent s'avérer meilleures que celles du BP pourvu que le découpage en clusters ayant donné naissance au graphe des régions, soit bien réalisé. En effet, aucun critère de choix permettant de justifier de l'utilisation d'un graphe des régions ou d'un autre n'a émergé jusque-là. Après avoir détaillé les équations de mise à jour du GBP, nous détaillons la construction du graphe des régions et nous insistons sur le fait que, conformément à son existence, il doit gérer les structures topologiques du graphe factoriel qui posent le plus de problèmes. Dans le cadre du décodage des codes LDPC, modélisables graphiquement par un graphe dit de Tanner, de telles structures ont été répertoriées dans la littérature. Il s'agit des trapping sets. Responsables de la dégradation du taux d'erreur binaire à très forts rapports signal-à-bruit (RSB), nous utilisons en conséquence ces structures pour réaliser un découpage original et pertinent du graphe de Tanner. Nous précisons que le GBP étant un algorithme naturellement très instable, nous introduisons un amortissement dans toutes nos simulations dont la loi d'évolution force l'algorithme à converger.

Par des résultats expérimentaux sur un code LDPC particulier entièrement couvert par des trapping sets, nous montrons que le GBP muni d'un amortissement a de meilleures performances que le BP pour de très forts RSB.

La comparaison du BP et du GBP pour des RSB moyens en termes de taux d'erreur binaire n'est pas concluante, les deux algorithmes étant équivalents. Nous utilisons donc les outils de la théorie des systèmes dynamiques pour tenter de les départager. Le champ de recherche lié à l'étude de la dynamique des décodeurs itératifs pour les codes LDPC est récente, par conséquent la littérature à ce sujet n'est pas très fournie. Les articles publiés sur la dynamique du BP traitent de l'utilisation des fonctions classiques liées aux systèmes chaotiques: le diagramme de bifurcation pour obtenir les RSB correspondant à de brusques changements de comportement et l'exposant de Lyapunov pour juger de la stabilité, *i.e.* de la sensibilité aux conditions initiales. Le GBP n'a en revanche jamais été traité de ce point de vue, faisant de notre étude un cas nouveau. A l'aide de ces fonctions mais également d'outils nouveaux que nous introduisons, nous réalisons une étude expérimentale afin de connaître les types d'attracteurs auxquels sont soumis le BP et le GBP ainsi que leur relation sur leurs performances de décodage. Il apparaît que les deux décodeurs ont tous deux des comportements typiques similaires: convergence vers un point fixe, attraction vers un cycle limite, attraction vers un attracteur chaotique. Cependant, les propriétés de ces attracteurs nous permettent d'affirmer que le GBP a une tendance plus prononcée que le BP à converger vers un bon décodage.

Résultats principaux et contributions:

- utilisation des trapping sets pour la construction du graphe des régions,
- introduction d'un coefficient d'amortissement,
- mise en place d'une méthode pour l'étude des décodeurs à très forts RSB,
- étude de la dynamique du GBP,
- introduction d'outils originaux pour la description de la dynamique des décodeurs en lien avec leurs performances de décodage.

Cette thèse est également un travail original au sens où elle est à la croisée entre divers champs de recherche: les codes correcteurs d'erreurs, la physique statistique, l'inférence statistique, l'optimisation lagrangienne et la théorie des sytèmes dynamiques. C'est pourquoi la première partie du manuscrit est dédiée à une revue détaillée des méthodes et outils issus de ces champs de recherche, tandis que la seconde partie traite de l'explication détaillée des deux algorithmes BP et GBP ainsi que de leur étude d'un point de vue dynamique.

Mots-clés : graphe factoriel, codes LDPC, énergie libre variationnelle, approximation basée régions, trapping sets, chaos.

Abstract

In this thesis, we tackle statistical inference on loopy factor graphs to approximate marginal probability distributions on nodes. In 1988, Pearl introduced a message-passing algorithm, called the Belief Propagation algorithm (BP), to solve this problem, but in spite of very good performance, BP did not offer optimal approximations. As a matter of fact, when applied on LDPC codes decoding, BP may present oscillations or even chaotic behaviors. Following the presentation of BP equations, we show that it is equivalent to Bethe approximation, used in statistical physics as a mean field approach, to estimate partition function of spin glasses by minimizing the variational free energy.

Oscillations of factor graphs, also called frustrations for spin glasses, may be solved by region-based approximation. It consists in creating an associated Bayesian network, called region-graph, which nodes, or regions, are gathering of factor graph nodes. This way, accuracy of this new approximation is hoped to be better than Bethe approximation's. According to equivalence between Bethe approximation and BP, we demonstrate that region-based approximation may be derived as a message-passing algorithm, called Generalized Belief Propagation (GBP). GBP decoding performance may be better than BP's provided that the clustering used to gather factor graph nodes is well suited. In fact, no choice criterion has been introduced to discriminate between all possible clusterings. After demonstrating GBP update equations, we present in detail the region-graph construction, and we emphasize that, in accordance with its origin, it has to deal with harmful topological structures of factor graph. LDPC codes, represented by particular factor graphs called Tanner graphs, present such structures, that have been published in the literature, called trapping sets. They are responsible for degradation of the Bit-Error Rate (BER) for high values of the Signal-to-Noise Ratio (SNR). We accordingly use these structures to cluster, in an original and relevant way, Tanner graphs. We specify that the GBP algorithm is naturally instable, which makes us introduce damping factors, of various evolutions, that help decoder converge. We show with experimental results on a specific LDPC code, entirely covered with trapping sets, that a dedicated construction coupled with a damped GBP offer better performance than BP for high SNR values.

Comparing BP and GBP for middle SNR values only using BER is not conclusive as both decoders exhibit equivalent results. Tools of dynamical systems theory open opportunities to distinguish them. Research area on dynamics of iterative LDPC decoders has been recently introduced making the literature on this topic not very abundant. Published papers on BP dynamics deal with usual functions that describe chaotic systems: bifurcation diagram offers SNR values where the decoder abruptly changes its behavior, Lyapunov exponent informs about BP stability, *i.e.* sensitivity to initial conditions. GBP has never been studied from this point of view, making this work innovative. By means of usual functions but also of new tools, we carry out an experimental investigation to bring out BP and GBP attractors and their links with decoding performance. Both algorithms similarly behave as they present close attractors for same SNR values: fixed point, limit cycle, chaos. However, attractors properties allow us to affirm that GBP tends to converge to correct decoding better than BP.

Main results and contributions:

- use of trapping sets construct region-graphs,
- introduction of a general damping factor for GBP,
- development of an experimental protocol to study decoders for high SNR values,
- study of GBP dynamics,
- introduction of innovative tools to describe decoders dynamics and to link them to their decoding performance.

The originality of this thesis remains in the fact that is is at the crossroads of several research areas: error-correcting codes theory, statistical physics, statistical inference, Lagrange optimization and dynamical systems theory. Therefore, first part of the manuscript is dedicated to reviews on methods and tools from most of these fields, and second part deals with foundations of the BP and GBP algorithms as well as the study of their dynamics.

Keywords : factor graph, LDPC codes, variational free energy, region-based approximation, trapping sets, chaos.

To spread a part of our ideas and results, we have participated to the following conferences:

13th IEEE International Conference on Communicating Systems (ICCS), November 2012. J.C. Sibel, S. Reynal and D. Declercq: A novel region-graph construction based on trapping-sets for the Generalized Belief Propagation.

1st IEEE International Conference on Control, Automation and Information Sciences (ICCAIS), November 2012. J.C. Sibel, S. Reynal and D. Declercq: On the Generalized Belief Propagation and its Dynamics.

9th European Conference on Complex Systems (ECCS), September 2012. J.C. Sibel, S. Reynal and D. Declercq: On the Belief Propagation and it Dynamics.

5th International Conference on Chaos, Modeling and Simulations (CMSIM), June 2012. J.C. Sibel, S. Reynal and D. Declercq: Evidence of chaos in the Belief Propagation for LDPC codes.

WASET International Conference on Computational Physics (ICCP), May 2012. J.C. Sibel, S. Reynal and D. Declercq: Experimental results about the dynamics of the Generalized Belief Propagation used on LDPC codes.

23rd Groupement de Recherche en Traitement du Signal et des Images (GRETSI), September 2011. J.C. Sibel, S. Reynal and D. Declercq: Etude de la dynamique pour le décodage itératif par Propagation de Croyances Généralisée.

7th Manifestation des Jeunes Chercheurs en Sciences et Technologies de l'Information et de la Communication (MAJECSTIC), November 2009. J.C. Sibel, S. Reynal and D. Declercq: Algorithme de décodage généralisé par propagation de croyances.

Results on dynamics of the Belief Propagation algorithm are published in the following journal:

Chaos, Modelling and Simulations Journal, Vol.1, 2013. J.C. Sibel, S. Reynal and D. Declercq: Evidence of chaos in the Belief Propagation for LDPC codes.

Contents

Introduction **1**

I Overview: methods and tools **7**

1 Context: channel coding **11**
1.1 Introduction . . . 11
1.2 Digital communications model . . . 11
1.3 Naive estimation . . . 13
1.4 Channel coding . . . 14
1.4.1 Encoding . . . 14
1.4.2 Maximum Likelihood Decoding . . . 15
1.5 Practical decoding algorithms . . . 16
1.5.1 Graphical representation . . . 16
1.5.2 Hard decoding . . . 17
1.5.3 Belief Propagation . . . 18
1.5.4 Alternatives to the BP . . . 20
1.6 Optimization . . . 21
1.7 Dynamics of LDPC decoders . . . 24
1.8 Conclusion . . . 25

2 Statistical physics of spin glasses **27**
2.1 Introduction . . . 27
2.2 Problems . . . 27
2.3 Spin glasses . . . 28
2.4 Mean-Field approach . . . 30
2.4.1 Message-Passing . . . 32
2.5 Replica method . . . 32
2.6 Cavity approach . . . 33
2.6.1 Ising chain . . . 34
2.6.2 Case of a factor graph . . . 36
2.6.3 Bipartite messages . . . 37
2.7 Mean-Field generalizations . . . 38
2.8 Conclusion . . . 38

3 Dynamical systems **39**
3.1 Introduction . . . 39
3.2 Chaotic attractors . . . 40
3.2.1 Logistic map . . . 40
3.2.2 Lorenz attractor . . . 42
3.3 Route to chaos . . . 43
3.3.1 Period-doubling . . . 44
3.3.2 Quasi-periodicity . . . 45
3.3.3 Intermittency and crises . . . 48
3.4 Lyapunov exponent . . . 48
3.4.1 Computation method 1 . . . 49
3.4.2 Computation method 2 . . . 49
3.4.3 Assumptions . . . 51
3.4.4 Computation method 3 . . . 52
3.5 State space reconstruction . . . 53
3.6 Conclusion . . . 55

II Graphical models and Belief Propagation algorithms **59**

4 Bethe approximation: an approach to the Belief Propagation **63**
4.1 Introduction . . . 63
4.2 LDPC codes and factor graphs . . . 63
4.2.1 Error correction capability of LDPC codes . . . 63
4.2.2 Tanner graph . . . 64
4.2.3 Factor graph . . . 65
4.3 Pairwise Tanner graph . . . 66
4.3.1 Tree case . . . 67
4.3.2 Loopy case . . . 67
4.4 General Tanner graph . . . 68
4.5 Energy . . . 70
4.6 Variational approach . . . 71
4.7 Optimization . . . 73
4.7.1 Constraints . . . 74
4.7.1.1 Normalization . . . 74
4.7.1.2 Omitted constraints . . . 77
4.7.2 Karush-Kuhn-Tucker conditions . . . 78
4.7.3 Solution . . . 78
4.8 BP assumption . . . 80
4.8.1 Tree-like Tanner graph . . . 81
4.8.2 Non tree-like Tanner graph . . . 81
4.8.3 Loops . . . 82
4.8.4 Physical interpretation . . . 83
4.8.5 Oscillation . . . 83
4.8.6 Trapping-sets . . . 84
4.9 Conclusion . . . 86

5 Region-based approximation: beyond the Belief Propagation 87
5.1 Introduction . . . 87
5.2 The region-based formalism . . . 88
5.2.1 Links with MF approach . . . 90
5.2.2 Counting numbers . . . 90
5.2.2.1 Computation rule . . . 91
5.3 Region-based approximation . . . 91
5.3.1 Example . . . 92
5.3.2 Region graph . . . 92
5.4 Generalized Belief Propagation . . . 94
5.4.1 Variational free energy . . . 94
5.4.2 Optimization . . . 96
5.4.3 Rotation of the Lagrange multipliers . . . 96
5.4.3.1 From λ to μ . . . 97
5.4.3.2 From μ to λ . . . 97
5.4.4 Message-passing . . . 99
5.4.4.1 Example . . . 100
5.4.4.2 Complexity . . . 101
5.4.5 Region-graph reduction . . . 101
5.4.6 Damping factor . . . 103
5.4.6.1 Constant damping factor . . . 104
5.4.6.2 Decreasing damping factor . . . 106
5.4.7 Iteration index . . . 108
5.5 Region-graph construction . . . 110
5.5.1 Systematic construction . . . 110
5.5.2 Square lattice . . . 111
5.5.3 Trapping-sets . . . 112
5.5.3.1 Local optimality . . . 113
5.5.4 Experiments . . . 117
5.5.4.1 BSC . . . 117
5.5.4.2 AWGNC . . . 119
5.6 Conclusion . . . 121

6 Dynamical behavior of iterative decoders 123
6.1 Introduction . . . 123
6.2 Preliminaries . . . 124
6.3 State space . . . 125
6.4 Mean square beliefs . . . 127
6.4.1 Definition . . . 127
6.4.2 Experimental protocol . . . 127
6.4.2.1 Attractor . . . 128
6.4.3 Experiments . . . 128
6.4.3.1 BP . . . 128
6.4.3.2 GBP . . . 129
6.5 Bifurcation diagram . . . 130
6.5.1 Definition . . . 130
6.5.2 Experiments . . . 130
6.5.2.1 BP . . . 130
6.5.2.2 GBP . . . 132
6.6 Reduced trajectory . . . 133

6.6.1 State space reduction 134
6.7 Sensitivity to initial conditions 138
6.7.1 Preliminaries 138
6.7.2 Lyapunov exponent 140
6.8 Hyperspheres and attractors width 142
6.8.1 Static hypersphere 142
6.8.2 Local hypersphere 144
6.9 Lengths ratio to determine the relative position in the state space 146
6.9.1 Complementary length 147
6.9.2 Lengths ratio 148
6.10 Conclusion 151

Conclusion **155**

List of figures **161**

List of tables **165**

Bibliography **167**

Introduction

Introduction

History and motivations

In this thesis we expose the analysis and the developments of methods to construct and solve graphical models. Many engineering and research problems can be formulated as a statistical inference wording, provided that they are modeled as Bayesian networks, Markov random fields or factor graphs. These problems mostly take part in activities about image processing, neural networks, spin glasses, error-correcting codes, etc. All of them are aimed at extracting either the joint probability distribution of their graphical model or the marginal distributions of the variables that build the graph. This way, it can be extracted a large number of properties that make sense for any of the concerned research areas, that helps construct strong connections between them.

The spin glasses are physical networks naturally represented by factor graphs. Statistical physics has the great challenge to obtain thermodynamical functions at equilibrium, as average energy, free energy, entropy, pressure etc. It has been demonstrated that all these functions are analytically dependent on a single quantity that is the partition function Z. The partition function has its origin in the Boltzmann's law that provides the joint distribution of any spin glass. Indeed, Z is a constant that ensures that the Boltzmann's law is normalized, *i.e.* it has to be a true distribution. It has been proved that computing Z consists in spanning of all possible states of the spin glass, that represents a considerable number of computations. Partition function is accordingly intractable, therefore it became a central topic of research in statistical physics. Physicists created original methods to approximate its value in particular cases: Sherrington-Kirkpatric model, Edwards-Anderson model, etc. The main method still studied nowadays is the Mean-Field (MF) approximation. The core of this method is to consider that any spin of a given spin glass is not singularly correlated to each of its neighbors anymore but to a mean field. The energy function, that summarizes all spins couplings, is then modified. Boltzmann's law assigns a probability weight to each state of the spin glass according to its energy, therefore the MF approximation entails a modification of the joint distribution. It turns out to be the product of all marginal distributions of probability. This mean field distribution is the basis to approximate the partition function by minimizing the mean-field variational free energy, whose minimum is the log-partition function.

By means of the statistical physics, the joint distribution of any graphical model could be approximated thanks to the MF approach. In addition, the notions of energy and entropy became familiar in other research areas. For instance, researchers connected parity-check equations of error-correcting codes to the spin glass energy function, and they succeed in relating the information and thermodynamical entropies. Low-Density Parity-Check (LDPC) codes, created by R. Gallager in 1963 and known to approach Shannon's limit, were then given a physical meaning. In the research area on error-correcting codes, LDPC codes are one of the main topics of research. Their links with spin glasses still spark much interest, as shown by the numerous papers published each year on this issue. The graphical model of any LDPC code is a Tanner graph, a specific factor graph. Edges of any factor graph are used as supports to solve inference by the use of message-passing algorithms, *e.g.* the Belief Propagation algorithm (BP) invented by J. Pearl in 1988. This algorithm has proved to be the best compromise between computation time and accuracy. Beyond energy, entropy and other physical quantities that are shared with statistical physics, it turned out that specific methods as BP are also available in both research fields. Indeed, fixed-points of BP are strictly equivalent to stationary points of the Bethe approximation, an extension of the MF approach. This equivalence helped researchers define BP as an iterative procedure to minimize the Bethe variational free energy F_β of any factor graph. In other words, the general inference problem was demonstrated to be identical to an optimization problem solved by the Lagrangian formalism.

At the origin, J. Pearl specified that BP was a perfect inference algorithm for any graphical model whose topology is tree-like. In addition, it was demonstrated in papers of T. Heskes in the 2000s that the optimization of F_β of any factor graph was always possible if it contains at most one loop. Unfortunately, this remarkable work and the related results based on the convexification of F_β could not be extended to any loopy factor graph. At this time, the challenge was to modify the BP algorithm to overcome the undesired loopy effects. Several methods were proposed, *e.g.* the ConCave-Convex Procedure (CCCP) from A. Yuille, but they implied a dramatic increase in the computation time. Once again, connections with spin glasses turned out to be conclusive. As a matter of fact, the problem of loops especially in BP decoding of the LDPC codes was perfectly understandable as the problem of frustration in statistical physics. Actually, not only the quantities and the methods but also the drawbacks are shared with the statistical physics, that helps to solve few issues. Physicists could make use of the cluster variation method or region-based approximation invented by Kikuchi in 1951 to deal with the frustration in spin glasses. This method, also called Kikuchi approximation after its designer, was eventually a MF approach that generalized the Bethe approximation by clustering any spin glass. Rather than factorizing the joint distribution on the single spins marginal distributions, it is approximated by the product of estimates of the pseudo-marginal distributions, namely the beliefs, on subsets of spins, called regions. It resulted in a new variational free energy F_K, which minimization by Lagrangian formalism provided a message-passing algorithm, fairly detailed in 2001 by J. Yedidia *et al*: the Generalized Belief Propagation algorithm (GBP). The support of this algorithm is the region-graph: a Bayesian network constituted by regions.

The relevancy of the region-based approximation relies on the significant degree of freedom to cluster any factor graph. Therefore, it appeared that the GBP algorithm would be an efficient substitute to the BP algorithm to annihilate the frustration, *i.e.* to cancel the harmful loops effect in the decoding of the LDPC codes.

At this point comes the main issue of the region-based approximation. Indeed, no criterion was given by Kikuchi to discriminate the numerous clusterings of any factor graph. Thus, given a factor graph, no one is able to judge the relevancy of an arbitrary construction of the region-graph. J. Yedidia *et al* introduced two principles that any region-graph should fulfill to present challenging results. The first states that the energy function of the region-graph should strictly equal the energy function of the basic factor graph. Accordingly, the number of occurrences of a given spin, or node of the factor graph, has to be identical in both energy functions. The second principle relies on the fact that the Kikuchi approximation evaluates the entropy by a region-based entropy, that is a linear decomposition over the regions entropies of the region-graph. It can be shown that even in the case of a tree-like topology, this decomposition remains an approximation. The best that can be done is to verify that making all the beliefs uniform distributions implies that the region-based entropy is maximum. Despite the power of this principle, it is again intractable to check for any region-graph. Thus the only way to evaluate the advantage of a region-graph is to run the GBP algorithm and to analyze the corresponding results versus BP's. This is the core of the work presented in this thesis.

A booming topic is the study of the dynamical behaviors of the inference algorithms, in particular in the case of the LDPC codes decoding. Few papers emerged in the last decade to deal with the bifurcations and the chaos in the decoding algorithms, especially the BP algorithm. Given that the dynamical system related to the BP algorithm is a multi-dimensional space, the mathematical developments are very hard tasks. Therefore, the study is oriented toward a use of numerous experiments of specific quantifiers to highlight typical behaviors. It appears that the BP algorithm is subject to non-trivial behaviors for particular signal-to-noise ratio range. However, the literature on the dynamics of the LDPC decoding is not substantial, given that this research domain is only in its early age. We demonstrate in this thesis that the study of the dynamics of the decoders helps compare and confront them. We introduce a toolbox to analyze the behaviors of the BP and the GBP. In this way, we emphasize the power of a region-graph construction, that provides a way to judge its relevancy for the region-based approximation.

Organization of the manuscript

The investigation we present in this manuscript lies within the areas of error-correcting code theory, statistical physics and dynamical systems theory. From these three fields are extracted tools and methods to develop and analyze message-passing algorithms with an extensive focus on BP and GBP. Thus we organize the manuscript as follows.

Part I is dedicated to the introducing the context in which the thesis began, digital communications. Brief notions on channel coding are presented the Belief Propagation algorithm and its derivatives, together with the optimization problem it is equivalent to. Then, we show preliminary results on the performance of BP that tend to investigate its dynamical properties. This constitutes Chap.1.

Chap.2 deals with statistical physics. We expose a few methods that are aimed at providing the partition function, as MF approach, replica method and cavity approach. About the latter, we accurately derive the corresponding equations to highlight their similarity with the BP equations.

In Chap.3 are exposed the main notions concerning the dynamical systems. By the use of famous examples, such as the logistic map and the Lorenz system, we present the classical quantifiers utilized to numerically describe the non-trivial behaviors like chaos.

Part II deals with the detailed development of the message-passing algorithms. Chap.4 contains the foundations of the BP algorithm. We introduce the Bayesian inference on factor graphs, then is presented the Bethe approximation and the equivalence between its stationary points and the fixed-points of the BP algorithm. We end this chapter by the open problems linked to the loop-like topology of the Tanner graph.

In Chap.5 is presented the region-based approximation. We demonstrate the equations of the GBP algorithm by the minimization of the region-based free energy. After describing the introduction of a damping factor in the GBP equations to maintain its convergence, we expose the basic construction rules of the region-graph with few examples on small factor graphs. After that is detailed our original constructions based on the specific topology of a selected LDPC code, then we exhibit encouraging experimental results.

In Chap.6 is described a toolbox to investigate the dynamics of the BP and the GBP algorithms for the LDPC codes. Presented as dynamical systems, we define their mathematical environment and the classical quantifiers, the bifurcation diagram and the Lyapunov exponents, among others. We then expose original tools to relate dynamics and decoding performance of the algorithms, allowing us to confront them.

Part I

Overview: methods and tools

Introduction

In this first part, we tackle a large overview of different algorithms that are aimed at analyzing and solving factor graphs. We describe the issue of this work according to several research areas, given that this thesis deals with a complex multifaceted investigation. As a matter of fact, inference problem on graphical models represents a challenge in error-correcting codes theory, statistical physics, Bayesian inference, nonlinear optimization and discrete dynamical systems theory. Each of these domains helps construct an exceptional research field where it is crucial and equally hard to merge the corresponding expertises. The work that we have carried out in this thesis needs an introduction in all these fields to well-understand the in-depth-study we will present. To this end, we describe in a few pages essential notions that helps the reader to get accustomed to these theories and the associated notations. It also allows us to focus on open problems that we will deal with all along the manuscript.

The methods presented in this part are not original, as many articles and theses have already dealt with their foundations and their properties. Thus, we do not exhibit many experimental results but the main papers and their significant contributions to the advances in the corresponding topics. In addition, we make an effort to introduce connections between different chapters in order to make the whole presentation consistent, given that problems and solutions emerging from any research field is not independent from the other ones. Moreover, we attempt to tackle each theory with the same global vision to enlighten noteworthy points, useful in our work. Accordingly, we do not describe all relevant aspects of each theory but only the ones that we will make use of in the manuscript, that still represent a quite substantial work.

This part is organized as follows. In first chapter, we introduce the general engineering context in which the thesis began, and the corresponding raised issues. We take the time to present a few decoding algorithms that attempt to solve inference on Tanner graphs. The second chapter is dedicated to an overview of statistical physics, especially methods that help compute the partition function – to some extent the free energy – of any spin glass. It includes the main equations of the mean-field approximation and the cavity method that both enable us to connect with the equations of iterative decoding algorithms. The third chapter deals with a description of the tools and methods that we generally need to describe any dynamical system. We explain the main notions about chaos, stability and bifurcations by use of typical examples, like the Lorenz system and the logistic map. We also exhibit calculations of the Lyapunov exponent that help numerically analyze chaos, which will prove to be very helpful in the second part of the manuscript.

- Chapter 1 -

Context: channel coding

§ 1.1 Introduction

In this chapter we present the engineering context and the motivations that inspired this thesis. To this end, we first provide few basic definitions about channel coding and Low-Density Parity-Check (LDPC) codes. Afterwards, we focus on the main decoding algorithms derived from the Belief Propagation (BP) that have been published in the literature. We exhibit brief results concerning their relative performance. We also relate the iterative decoding with the optimization of the free energy in statistical physics, that will help us focus on drawbacks of BP. We mention the main articles that made the research progress. We end this chapter considering BP as a dynamical system, and we give the main publications that have proposed first investigations on this topic.

§ 1.2 Digital communications model

We consider a couple made with an emitter and a receiver. The emitter sends a signal, mapped to a sequence $\mathbf{u} = [u_1, \ldots, u_\kappa]$ of κ bits, to the receiver. In the case where no disturbance affects the signal the receiver would obtain the very sequence $\mathbf{u}$. However, in real life, the "space" between both users is non-empty in the sense that many other signals are traveling inside it. These signals interfere with the one we are working on. Furthermore, space itself is disruptive due to its non-homogeneous constitution, therefore it is as well a source of noise. The consequence is that the emitted signal is strongly disturbed. The raised issue is then how to faithfully recover it. We call the space between the emitter and the receiver the *channel*. A channel can be modeled as a function $ch(\mathbf{u})$ that maps the input sequence $\mathbf{u}$ to an output sequence $\mathbf{y} = [y_1, \ldots, y_\kappa]$ which values depend on the nature of the channel, see Fig.1.2.

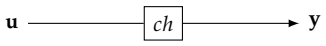


Figure 1.1: *Channel*

The channel is assumed to produce random noise and makes the output a random reading. Therefore we can define a conditional probability distribution $p(\mathbf{y}|\mathbf{u})$ that is called the *likelihood*. We consider a memoryless channel therefore the likelihood is factorisable as:

$$p(\mathbf{y}|\mathbf{u}) = \prod_{i=1}^{\kappa} p(y_i|u_i) \tag{1.1}$$

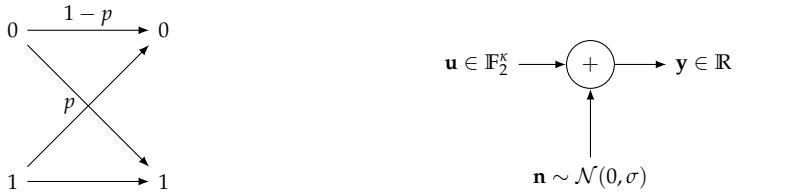
One of the main channels investigated in the literature are:

- the Binary Symmetric Channel (BSC) of crossover probability p. Any input bit u_i is flipped by the channel with a probability p. We represent this channel by the diagram in Fig.1.2(a). The likelihood of the bit u_i is:

$$p(y_i = 0|u_i = 0) = p(y_i = 1|u_i = 1) = 1 - p \tag{1.2}$$
$$p(y_i = 0|u_i = 1) = p(y_i = 1|u_i = 0) = p \tag{1.3}$$

- the Additive White Gaussian Noise Channel (AWGNC) of power σ^2. Any input bit u_i is disturbed by a random element n_i that follows a Gaussian law whose variance is σ^2, *i.e.*, $n_i \sim \mathcal{N}(0, \sigma)$. We represent this channel by the diagram in Fig.1.2(b). The likelihood of any bit u_i is given by

$$p_i(y_i|u_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - u_i)^2}{2\sigma^2}} \tag{1.4}$$



(a) Binary symmetric channel (b) Additive white Gaussian noise channel

Figure 1.2: *Two channel models*

The aim of the receiver is to estimate $\mathbf{u}$ by a sequence $\hat{\mathbf{u}}$ using the likelihoods and metrics like the Hamming distance $d_H(\mathbf{u}, \hat{\mathbf{u}})$. Defined as the number of bits that differ between both sequences, it is aimed at being as low as possible.

§ 1.3 Naive estimation

The input sequence $\mathbf{u}$ is estimated by scanning all possible states of $\mathbf{u}$ and extract the most likely one according to the *a posteriori* probability distribution $p(\mathbf{u}, \mathbf{y})$. This distribution is computed according to the Bayes rule:

$$p(\mathbf{u}, \mathbf{y}) = p(\mathbf{y}|\mathbf{u})p(\mathbf{u}) \tag{1.5}$$

As every word $\mathbf{u} \in \mathbb{F}_2^\kappa$ has the same probability to be sent, then $\hat{\mathbf{u}}$ is:

$$\hat{\mathbf{u}} = \arg\max_{\mathbf{u}} p(\mathbf{y}|\mathbf{u}) \tag{1.6}$$

The memoryless assumption allows us to reduce the search from 2^κ states to a search over two states for each bit, *i.e.* 2κ states. In other words, maximizing the joint probability distribution is equivalent to maximizing each marginal likelihood:

$$\hat{\mathbf{u}} = [\hat{u}_1, \ldots, \hat{u}_\kappa] \tag{1.7}$$

$$\forall i \in \{1, \ldots, \kappa\}, \quad \hat{u}_i = \arg\max_{u_i} p_i(y_i|u_i) \tag{1.8}$$

We display in Fig.1.3 the performance of this method with $\kappa = 200$. The commonly used estimator to judge about the accuracy of any transmission is the Bit-Error Rate (BER), *i.e.*, the ratio between the number of wrong estimated bits and κ.

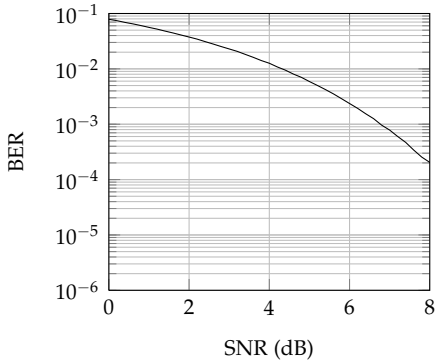


Figure 1.3: *Bit-error rate of an uncoded transmission*

This transmission method is not very efficient given that the BER is too far away from Shannon's limit. Practically we need the information to be more robust against the channel noise to ensure a reliable transmission.

§ 1.4 Channel coding

To improve the performance of a transmission, error-correcting codes (ECC) are used. An ECC makes the input signal more robust against the channel noise basically by encoding it. The *encoding* consists in adding redundancy bits that the receiver uses to decode. The *decoding* process is aimed at correcting errors caused by the channel. The model is presented in Fig.1.4.

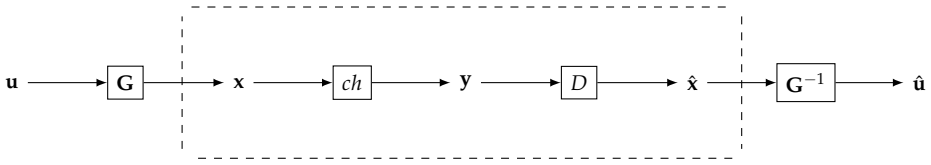


Figure 1.4: *Model with an error-correcting code*

Blocks **G** and D are respectively the encoding and the decoding procedures. The dashed box frames the channel coding part, *i.e.*, the part that we will tackle in this manuscript, particularly the box D. We do not focus on the design and the inversion of **G** that are part of a related but different research area in coding theory.

1.4.1 Encoding

In this work, we investigate the linear ECC, whose encoding blocks are binary matrices, denoted by **G**, that map $\mathbf{u} \in \mathbb{F}_2^\kappa$ to $\mathbf{x} \in \mathbb{F}_2^N$ with $N > \kappa$. Each coordinate x_i is a linear combination of few elements of **u**. We define the ECC, called simply the code, as the set $\mathcal{C} = Im(\mathbf{G})$, each vector of this set is called a *codeword*. For example the Hamming encoding matrix [R.W50] is:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

whose associated code is a set of $2^4 = 16$ codewords $\mathcal{C} = \{\mathbf{x} = [x_1 \dots x_7]\}$ such that for any of these codewords:

$$\begin{aligned} x_1 &= u_1 \\ x_2 &= u_2 \\ x_3 &= u_3 \\ x_4 &= u_4 \\ x_5 &= u_1 \oplus u_2 \oplus u_3 \\ x_6 &= u_1 \oplus u_2 \oplus u_4 \\ x_7 &= u_1 \oplus u_3 \oplus u_4 \end{aligned}$$

with $\oplus$ the binary addition. We define the parity-check matrix $\mathbf{H}$ of a code as $Im(\mathbf{G}) = Ker(\mathbf{H})$. The advantage of this matrix is that it maps the bits $\mathbf{x} = [x_1, \ldots, x_N]$ that are the elements we work on in the channel coding. It is easier to check if a vector $\mathbf{x}$ is a codeword by comparing $\mathbf{H}.\mathbf{x}$ with $\mathbf{0}$ rather than searching for a vector $\mathbf{u}$ such that $\mathbf{x} = \mathbf{G}.\mathbf{u}$. For example the Hamming parity-check matrix is:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

thus any codeword $\mathbf{x} \in \mathcal{C}$ verify the following equalities:

$$\begin{aligned} x_1 \oplus x_2 \oplus x_3 \oplus x_5 &= 0 \\ x_1 \oplus x_2 \oplus x_4 \oplus x_6 &= 0 \\ x_1 \oplus x_3 \oplus x_4 \oplus x_7 &= 0 \end{aligned}$$

The most famous codes are the Turbo-codes [C. 93] and the Low-Density Parity-Check (LDPC) codes [R.G63]. The work we will present here is oriented to the latter ones. Any row H_i of $\mathbf{H}$ applied on a vector $\mathbf{x}$ is a binary sum of few elements of $\mathbf{x}$:

$$H_i.\mathbf{x} = x_j \oplus x_k \oplus x_l \oplus \ldots \tag{1.9}$$

with $j, k, l, \cdots \in \{1, \ldots, N\}$. This sum equals zero if $\mathbf{x}$ is a codeword, and one otherwise. That is why the M rows of $\mathbf{H}$ are called parity-check equations. In the example of the Hamming code, M is smaller than N which is actually a common property of ECC. Indeed, using an ECC makes the number N of bits to transmit $\mathbf{x} = [x_1, \ldots, x_N]$ larger than the number κ of original bits $\mathbf{u} = [u_1, \ldots, u_\kappa]$. Adding parity-check equations provides a strong robustness against the channel noise, but it also dramatically reduces the information flow. Using more parity-check equations than the number of transmission bits is not reasonable, therefore we always consider that $M < N$, and most times, $\kappa = N - M$.

1.4.2 Maximum Likelihood Decoding

Using the ECC that encoded the information, the receiver is now able to recover the word $\mathbf{x}$ as faithfully as possible, which allows it to recover the information word $\mathbf{u}$. To this end it uses a decoding algorithm. A decoding algorithm or decoder consists in finding the codeword $\hat{\mathbf{x}} \in \mathcal{C}$ that is the closest one to the vector $\mathbf{x}$, according to the Hamming distance, given the knowledge of $\mathbf{H}$ and $\mathbf{y}$. The Maximum Likelihood Decoder (MLD) is said to be optimal in the sense that it offers the lowest Hamming distance. This algorithm consists in spanning through the whole code to extract the codeword $\hat{\mathbf{x}}$ that maximizes the likelihood:

$$\forall \mathbf{x} \in \mathcal{C} \backslash \hat{\mathbf{x}}, \quad p(\mathbf{y}|\mathbf{x}) \leq p(\mathbf{y}|\hat{\mathbf{x}}) \tag{1.10}$$

In Fig.1.5 is presented the performance of the MLD applied on the Hamming code. Even though this algorithm is optimal, it involves a really large computation time given that the code contains 2^κ codewords.

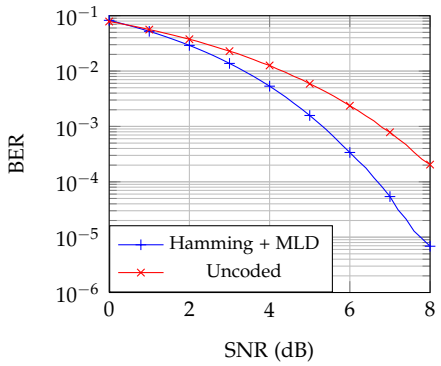


Figure 1.5: *Maximum likelihood decoding on the Hamming code*

Furthermore, in practice the code $\mathcal{C}$ is not systematically available, *i.e.* the receiver only has $\mathbf{H}$ at its disposal. In this case, it is unavoidable to scan the 2^N sequences of $\mathbb{F}_2^N$ to find the optimal $\hat{\mathbf{x}}$, given that N can reach very large values. Finally this optimal decoding is not well-suited because of its computation time.

§ 1.5 Practical decoding algorithms

Any error-correcting code has to be associated with a decoding algorithm, also called decoder, to retrieve the information bits that were corrupted by the channel noise. A few codes are associated with specific decoders, *i.e.* algorithms that are either built especially according to their own topology, *e.g.* the polar codes [E. 09], or built with dedicated algebraic rules *e.g.* the Reed-Solomon codes [I.S60] and the non-binary BCH codes [A. 59],[R.C60]. As mentioned in the previous paragraph, the optimal decoder for any LDPC code is the MLD but it has no practical interest therefore it is replaced by iterative decoders of dramatic lower complexity.

1.5.1 Graphical representation

Any ECC that is represented by its parity-check matrix has a graphical representation, called the Tanner graph [F.R01]. It is a bipartite graph whose vertices are:

- variable nodes $\{X_i\}_i$ $\circ$ (one per bit),
- check nodes $\{c_a\}_a$ $\square$ (one per parity-check equation).

To lighten notations, we denote by X_i (resp. c_a) the corresponding bit and the variable node (resp. the corresponding parity-check equation and the check node). The value x_i is said to be the state of X_i. We draw an edge e_{ia} between nodes X_i and c_a if and only if X_i is an argument of the parity-check equations c_a.

And we say that c_a (resp. X_i) is a neighbor of X_i (resp. c_a), *i.e.*, $c_a \in \mathcal{N}_i$ (resp. $X_i \in \mathbf{X}_a$). Usually, we represent the Tanner graph in a 2-level mode, *e.g.* the Tanner graph of the Hamming code represented in Fig.1.6.

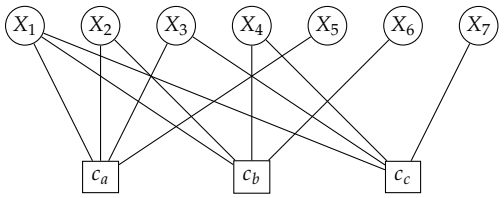
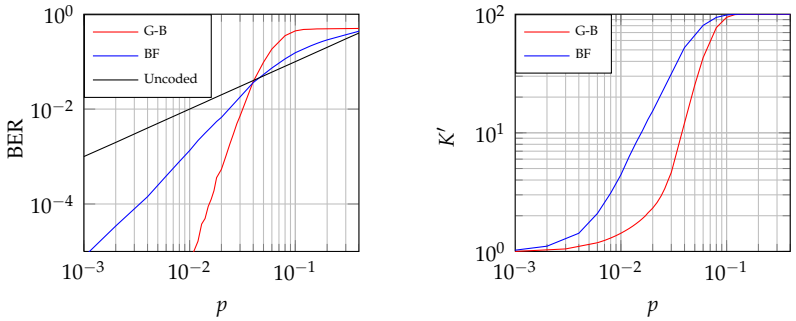


Figure 1.6: *2-level Tanner graph of the Hamming code*

1.5.2 Hard decoding

The decoders that were first introduced were the hard decoders, *e.g.* the Bit-Flipping (BF) decoder [D. 11] and the Gallager-B (G-B) [R.G63],[S. 06].



(a) Bit-error rate

(b) Number of iterations

Figure 1.7: *Tanner code ($N = 155$) hard decoding on a BSC of crossover probability p*

The G-B is a message-passing algorithm where each edge of the Tanner graph carries two directed messages. A message from a variable node to a check node is the state that this variable node should take according to all the check nodes it is related to, except the destination. A message from a check node to a variable node is the XOR between all the incoming messages except the one from the destination. Actually, any message is valued in $\{0, 1\}$. The BF appears to be the most natural algorithm, after the MLD: knowing that the aim of an LDPC code is to obtain all its parity-check equations verified, it consists in flipping the bits whose neighboring check nodes are mainly unsatisfied until all the check nodes are verified.

Such an algorithm is not strictly speaking a message-passing algorithm, but thinking that the flip of a variable node X_i results from the information collected on its neighborhood, it could be regarded as messages transmissions. In the case of a BSC, the BF and the G-B have the clear advantage to be very light according to the hardware requirements, and of quite low complexity

given that the update rules are very simple. We display in Fig.1.7(a) the necessary number of iterations K' to obtain a valid answer from both decoders, *i.e.* when the estimate of any of them is either verified by all the check nodes or identical to the estimate of the previous iteration. We observe that the G-B turns out to be significantly faster than the BF. In Fig.1.7(a) we display the BER of the two algorithms. We clearly conclude that the G-B is also more accurate than the BF. As an example, for $p \approx 10^{-2}$, G-B yields a hundred times lower error rate than BF with only four times less computing iterations.

1.5.3 Belief Propagation

Diversity of transmission channels made research being more active about decoders given that some channels result in soft output values. Hard decoders are not able to deal with them, *e.g.* AWGNC, Rayleigh channel etc. In addition, even for hard valued channels, see results on Fig.1.7, generalizations of the G-B worth studying in order to improve it. Extending the messages alphabet should provides significant improvements.

Considering no hardware implementation, we can make use of an infinite alphabet by assuming that any message from a node a to a node b in the Tanner graph is the probability distribution of b conditioned by a. This assumption makes the decoder being not a hard decoder anymore but a soft decoder. Such a model was proposed in [Pea88] as a solution to the inference problem on Bayesian trees and polytrees. It provides marginal probability distributions, or marginals, of all random variables that build any tree. Extracting arguments of the maximum of all marginal distributions then results in the most likely state. This model is the Belief Propagation (BP) that constitutes the basis of many other iterative algorithms as it was pointed out in [F.R98],[D. 95],[R.J98].

A decoder that is largely used in practical cases and that stems from BP is the Viterbi algorithm [A. 67]. It was created for decoding convolutional codes and Turbo-codes. It does not extract approximates of marginals but it provides the most likely sequence of bits according to a graphical representation of the code, the trellis, that is not a Bayesian network but a hidden Markov chain. The Viterbi algorithm is also known as the Max-Product version of BP closely related to dynamics programming [D.P01]. Also used to decode Turbo-codes and convolutional codes, the BCJR decoder [F.R01], often called Forward-Backward, focuses on the minimization of the bit-error rate whereas the Viterbi is aimed at minimizing the word-error rate.

The Sum-Product Algorithm (SPA) [F.R01] is the name of BP written especially for factor graphs *e.g.* Tanner graphs of LDPC codes. Originally built for pairwise Bayesian networks that are not bipartite graphs, we only find one update rule. The message from the variable node X_i to the variable node X_j, displayed on Fig.1.8, is defined as:

$$\boxed{m_{ij}(x_j) = \sum_{x_i} \psi_{ij}(x_i, x_j)\phi_i(x_i, y_i) \prod_{X_k \in \mathcal{N}_i \setminus X_j} m_{ki}(x_i)} \tag{1.11}$$

where:

- $\psi_{ij}(x_i, x_j)$, the *compatibility function*, indicates if the values x_i and x_j of the variable nodes X_i and X_j are valid. In the context of LDPC codes, $\psi_{ij}(x_i, x_j) = 1 \oplus x_i \oplus x_j$ is the parity-check equations that links X_i with X_j,
- $\phi_i(x_i, y_i)$, the *evidence*, provides the observation we obtain from the node X_i. It is the likelihood $p_i(y_i|x_i)$.

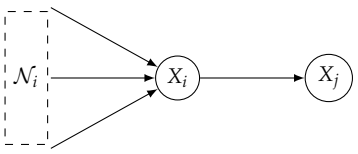


Figure 1.8: *Message m_{ij} in a Bayesian network*

This update equation is a sum-product rule, given that it contains both basic operations. Qualitatively, sum operand allows or not to propagate information coming to X_i (except information from X_j), and product operand provides resulting information. This update rule has been extended [F.R01],[J.S05] to graphs that are neither pairwise nor Bayesian networks but multi-body interaction undirected graphs as Markov Random Fields (MRF) or factor graphs $\mathcal{G} = (\mathbf{V}, E)$. These models are bipartite, *i.e.* the set of nodes $\mathbf{V}$ is a union of two sets of nodes of different kinds $\mathbf{X} \cup \mathbf{C}$. Multi-body interactions between nodes of $\mathbf{X}$ are not pairwise then they cannot be implicitly represented by edges, they need their own nodes $\mathbf{C}$. An edge $e \in E$ represents a contribution of a given variable of $\mathbf{X}$ to a given multi-body interaction in $\mathbf{C}$. Two update rules, that gave birth to the SPA, were defined according to Fig.1.9:

from variable nodes to check nodes: $\forall e_{ia} \in E, \quad \forall x_i \in \{0,1\}$,

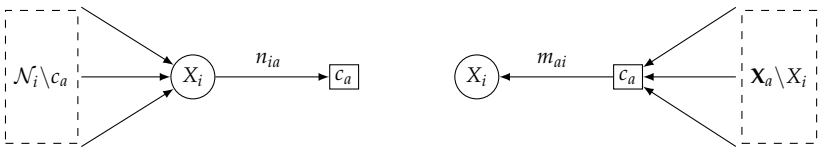
$$m_{ia}^{(k)}(x_i, \mathbf{y}) = \frac{l_i(x_i, \mathbf{y})}{Z_{ia}} \prod_{c_b \in \mathcal{N}_i \setminus c_a} n_{bi}^{(k-1)}(x_i, \mathbf{y}) \qquad (1.12)$$

from check nodes to variable nodes: $\forall e_{ia} \in E, \quad \forall x_i \in \{0,1\}$,

$$n_{ai}^{(k)}(x_i, \mathbf{y}) = \sum_{\mathbf{x}_a \cup x_i} f_a(\mathbf{x}_a) \prod_{X_j \in \mathbf{X}_a \setminus X_i} m_{ja}^{(k)}(x_j, \mathbf{y}) \qquad (1.13)$$

where:

- k is the iteration index,
- $l_i(x_i, \mathbf{y}) = p(y_i|x_i)$ is the likelihood of the variable X_i in the state x_i,
- Z_{ia} ensures that $m_{ia}^{(k)}(0, \mathbf{y}) + m_{ia}^{(k)}(1, \mathbf{y}) = 1$,
- $\mathbf{x}_a = \bigcup_{X_i \in \mathbf{X}_a} x_i$,
- $f_a(\mathbf{x}_a) = 1 \oplus \bigoplus_{X_i \in \mathbf{X}_a} x_i$.



(a) Message n_{ia} from the variable node X_i to the check node c_a with $\mathcal{N}_i = \{c_b \in \mathbf{C} \mid (c_b, X_i) \in E\}$

(b) Message m_{ai} from the check node c_a to the variable node X_i with $\mathbf{X}_a = \{X_j \in \mathbf{X} \mid (X_j, c_a) \in E\}$

Figure 1.9: *Messages in a factor graph*

At the first iteration $k = 0$, messages $\{m_{ia}^{(0)}(x_i, \mathbf{y})\}_{i,a,x_i}$ are initialized either to uniform random values or to the likelihoods $m_{ia}^{(0)}(x_i, \mathbf{y}) = l_i(x_i, \mathbf{y})$. Statistically, the initialization choice does not wield much influence on the outcome of the algorithm, therefore it is up to the experimenter. BP provides the probability distribution $\{b_i(x_i, \mathbf{y})\}_{x_i,\mathbf{y}}$ of each variable node X_i, called the belief, according to the following equation:

$$\forall X_i \in \mathbf{X}, \quad \forall x_i \in \{0,1\}, \quad b_i^{(k)}(x_i, \mathbf{y}) = \frac{l_i(x_i, \mathbf{y})}{Z_i} \prod_{c_a \in \mathcal{N}_i} n_{ai}^{(k)}(x_i, \mathbf{y}) \tag{1.14}$$

where $Z_i = b_i^{(k)}(0, \mathbf{y}) + b_i^{(k)}(1, \mathbf{y})$ ensures that the belief builds a true distribution. Finally a codeword estimate is computed as the concatenation of the most likely states of these distributions:

$$\hat{\mathbf{x}}^{(k)} = \bigcup_{X_i} \arg\max_{x_i} b_i^{(k)}(x_i, \mathbf{y}) \tag{1.15}$$

SPA is stopped whether $\mathbf{H}.\hat{\mathbf{x}} = \mathbf{0}$, *i.e.* it results in a codeword, or for any edge e_{ia}, the message $m_{ia}^{(k)}(x_i, \mathbf{y})$ (or $n_{ai}^{(k)}(x_i, \mathbf{y})$) does not change from an iteration to the next one, *i.e.* SPA has converged.

1.5.4 Alternatives to the BP

Even though the complexity of the SPA was very low compared with the MLD one, it turned out necessary to simplify the update equations for specific decoding applications. One of the solutions consists in working in the log-domain. It leads us to the logarithm version of the SPA, partly introduced in [R.G63], where a vector message $[m_{ai}(0)\ m_{ai}(1)]^T$ becomes a scalar message $M_{ai} = \log \frac{m_{ai}(0)}{m_{ai}(1)}$. The update rule is then:

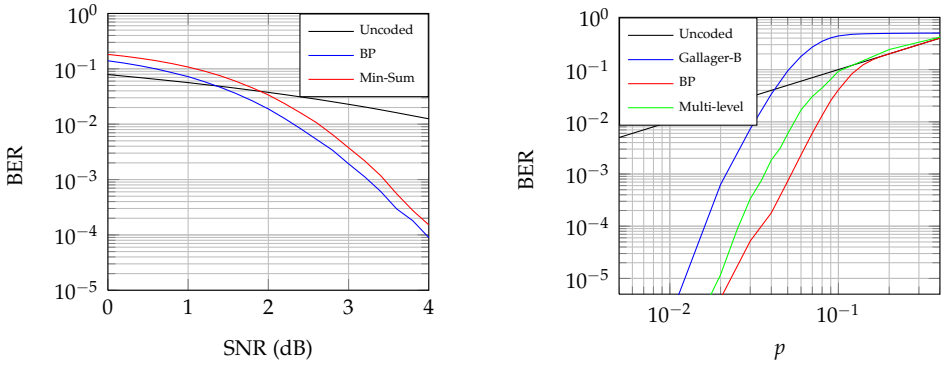
$$M_{ai} = 2 \tanh^{-1} \left(\prod_{X_j \in \mathbf{X}_a \backslash X_i} \tanh \frac{N_{ja}}{2} \right) \tag{1.16}$$

where $N_{ja} = \log \frac{n_{ja}(0)}{n_{ja}(1)}$. Computation complexity is dramatically decreased, and computation memory is twice less given that only one quantity is used for any message and for any belief. However the applications tanh and $\tanh^{-1}$ are not easily handled. To circumvent this undesired effect of the log-domain we could store an accurate tab of these functions but it would need a non-negligible memory. The Min-Sum (MS) algorithm, an approximation to the log-SPA [J. 96], was defined such that a message from check node c_a to variable node X_i contains only basic operations:

$$M_{ai} = \left(\prod_{X_j \in \mathbf{X}_a \setminus X_i} \text{sign}(N_{ja}) \right) \min_{X_j \in \mathbf{X}_a \setminus X_i} \{|N_{ja}|\} \tag{1.17}$$

that substantially reduced computation time, unfortunately at the expense of the performance.

Last generic decoders we have to mention are the Finite Alphabet Iterative Decoders (FAID) also called multilevel decoders presented in [S.K10],[S.K11]. They consist in modifying the MS algorithm by changing the update rule of the messages from the variable nodes to the check nodes $\{n_{ia}\}_{i,a}$, each of these messages being computed according to nonlinear equations. Studies tend to prove that these decoders provide lower BER than the SPA on the BSC for very low values of the crossover probability of the channel, that is a non-negligible progress [D. 12]. Unfortunately, FAID are somehow hard decoders that are not easy to use on AWGNC, furthermore they cannot be used on any LDPC code. We display on Fig.1.10 performance of hard and soft decoders on BSC and AWGNC.



(a) Gaussian channel (b) BSC

Figure 1.10: *Tanner code decoding ($N = 155$)*

§ 1.6 Optimization

BP sparked a lot of interest among different research areas: Bayesian inference, channel coding and also nonlinear programming. In [J.S03a],[J.S02] is described a physical aspect of the decoding algorithms for spin glasses [Nis01]

and more generally for factor graphs, *e.g.* the Tanner graphs. It turns out that the inference problem is equivalent to the minimization of the free energy F of the graph. The free energy is computed from the Boltzmann's joint distribution $p(\mathbf{x})$:

$$p(\mathbf{x}) = \frac{e^{-E(\mathbf{x})}}{Z} \tag{1.18}$$

$$F = -\log Z \tag{1.19}$$

where Z is the *partition function* that ensures the normalization of the distribution, $Z = \sum_{\mathbf{x}} e^{-E(\mathbf{x})}$. Given that computing Z consists in spanning through the 2^N possible states of the graph, we conclude that it is not tractable. Instead of the free energy, variational free energies [H.A35],[R. 51] computed by tractable factorized joint distributions prove to be relevant approximations. Therefore the main issue is to find the factorized distribution that minimizes the variational free energy.

This problem can be solved by BP. Indeed, it was shown in [J.S05],[P. 03] that fixed points of BP are equivalent to stationary points of the variational Bethe free energy F_β. However, the update rules of BP are not exactly established according to the constrained Lagrangian optimization of F_β, the equivalence is only true for steady states, that are not always encountered. In addition the loop-like topology of most factor graphs prevents the minimization from being optimal, *i.e.* either the result is only a local minimum, or the minimization completely fails in the sense that it oscillates between two or more states or even it falls into chaotic behaviors. Actually, the barrier is the fact that F_β is very rarely a convex functions of the beliefs, making its minimization a very hard task *i.e.* lack of convergence of BP is an open issue.

The optimization of F_β is dealt in [T. 03a],[T. 04],[T. 06a] by extracting conditions on the graph for the existence and the uniqueness of the stationary points, *e.g.* the BP yields exact minimum of F_β for graphs that contain only one single loop. In spite of this encouraging property, this result makes the investigation on the BP more complex because many graphs contains much more than one single loop. An alternative version of the BP is proposed in [A.L03] to ensure the convergence. This ConCave-Convex Procedure (CCCP) consists in splitting F_β into a convex function F_{vex} and a concave function F_{cav} in order to iteratively find, as in Newton's method, the point that cancels the gradient of the variational free energy:

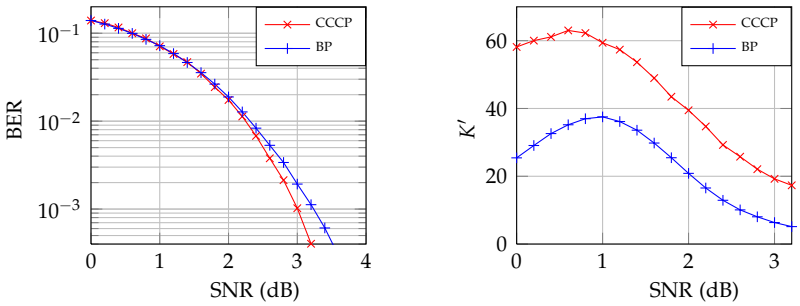
$$\nabla F_{vex}(\{b_i^{(k+1)}\}_i) = -\nabla F_{cav}(\{b_i^{(k)}\}_i) \Longrightarrow \nabla F_\beta(\{b_i^{(k\to K)}\}_i) = \mathbf{0} \tag{1.20}$$

with K the large number of iterations. The algorithm is made of two nested loops:

- the inner loop on the Lagrange multipliers ensures that the linear constraints are verified (iteration index: k),
- the outer loop provides the values of the beliefs (iteration index: τ).

On one hand, update equations of the multipliers involve the beliefs, on the other hand, the update equations of the beliefs involve the multipliers. Then depending on the initialization of the algorithm, computation time can be

very large compared to the original BP one. This property often makes this algorithm unpractical. However, messages update rules are strictly consistent with the Bethe free energy minimization, it even appears clear that the messages are none other than the Lagrange multipliers themselves. Thus the CCCP is of very high interest to understand a few subtleties of BP. We display on Fig.1.11(a) the BER of the CCCP and the BP algorithms on the Tanner code.



(a) Bit-error rate (b) Computation time

Figure 1.11: *CCCP decoding of the Tanner code*

We observe that the accuracy of the CCCP is slightly improved compared with the BP, as the SNR is increased. The drawback of this method is depicted in Fig.1.11(b) where it appears that the number K' of loops (outer loops for the CCCP) is almost doubled by the use of the CCCP compared with the BP. Furthermore, according to Fig.1.12, for low SNR values, the number of inner loops τ' is non-negligible. For instance, for an SNR $= 3.2$ dB, we obtain $\tau' = 77$ that is very high. Even though this value decreases as k increases, computation time is highly lengthened by the CCCP method.

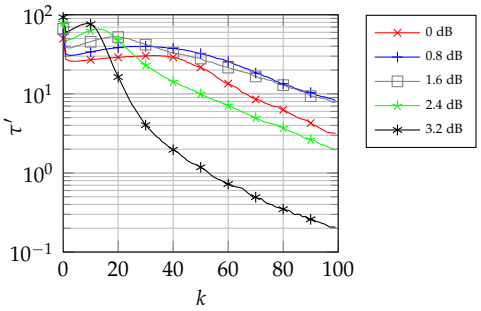
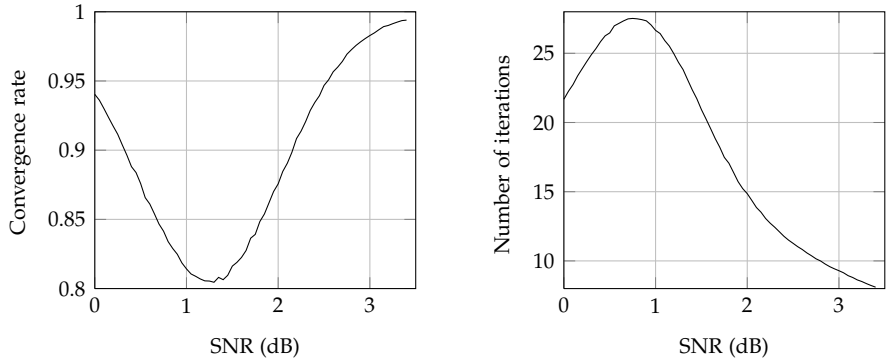


Figure 1.12: *Inner loop of the CCCP decoding of the Tanner code*

§ 1.7 Dynamics of LDPC decoders

Knowledge of the average BER is not sufficient to understand the whole BP algorithm, *e.g.* properties about stability, convergence, fluctuations around the BER, etc. For instance, we represent on Fig.1.13 the convergence rate and the required number of iterations to converge, both averaged over ten thousand simulations.



(a) Convergence rate

(b) Number of iterations before convergence

Figure 1.13: *Tanner code ($N = 155$) decoding by BP on AWGNC*

One would *a priori* think that, as the BER, these quantifiers should decrease according to the SNR. However, it appears an unexpected behavior of BP around SNR $= 1.0$dB: the convergence rate significantly decreases and the number of iterations required to have the BP converge considerably increases. Between 2.0dB and 3.0dB both quantities recover their expected behavior relatively to their initial values at 0.0dB. As mentioned in many papers, the loop-like topology of most codes could be responsible for this undesired phenomenon.

Another factor is the nonlinearity of the update rule that enables us to compute the messages from the check nodes to the variable nodes (1.13), (1.16), (1.17). According to research results on dynamical systems, nonlinearities are responsible for non-trivial or even unexpected phenomena [Hiloo] *e.g.* oscillations, aperiodicity or even chaos.

Studies of the nonlinearities of the BP has been well introduced for Turbocodes [D. 00],[L. 02], with few improvements that help the decoding to be more accurate. It was introduced measurements to quantify nonlinear effects. The bifurcation diagram revealed that SNR values could be shared into intervals according to the BP behavior, and the largest Lyapunov exponent helped confirm the chaotic behavior of the decoder for particular SNR values. Concerning LDPC codes, a less extensive work was presented in the sequel [L. 06]: only pseudo-trajectories without any bifurcation diagram or Lyapunov spectrum that could exhibit BP chaotic behaviors. In [X. 05] is presented a dense work on BP decoding using the *mean square beliefs*:

$$E(k) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(b_i^{(k)}(x_i)\right)^2} \tag{1.21}$$

that models quite faithfully the BP behaviors. The depicted bifurcation diagram reveals clear SNR intervals, as established for Turbo-codes, and few figures that represent the eigenvalues evolutions of the Jacobian matrix according to the SNR for fixed-points help extract the bifurcations natures. However, chaos did not appear very easily, the largest Lyapunov exponent does not accurately reflect the BP behaviors. Another work on the nonlinear dynamics of the BP was introduced in [B.S10a],[B.S10b] but the goal was to study the dynamics understood as the bounds of the log-*a posteriori*-ratio. It proved that the BP does not generate values that increase these data *ad infinitum*. Finally the whole behavior of the BP for the decoding of LDPC codes is not well understood despite very good works that provide interesting clues.

§ 1.8 Conclusion

Research on decoding algorithms is confronted to the problem of suboptimality of BP-based iterative decoders. Optimizing the variational free energy, taken as the cost function of any message-passing algorithm, is a very hard task given that this function is almost never convex. The CCCP appears a good candidate to overcome this problem, but we saw that its computation time makes it unrealistic. Thus, investigating on the free energy is necessary to understand its importance in the decoding algorithms. In this purpose, we describe, in the following of the manuscript, basic notions of statistical physics of spin glasses.

Furthermore, preliminary results that we exhibited about BP convergence allows us to deeply investigate this decoder with the tools provided by the dynamical systems theory. Therefore we need to introduce basic notions and tools of this theory, in order to use them to highlight the dynamical properties of the decoder.

- Chapter 2 -

Statistical physics of spin glasses

§ 2.1 Introduction

In this chapter we describe physics behind the work presented in this thesis. Decoding algorithms have deep relations with methods of statistical physics of spin glasses. In order to keep a reasonable manuscript size, we will not go deeply into the following theories. The purpose is to give an overview of the tools used in our research.

In section 2.2 are exposed the main principles of statistical physics that are necessary to develop the decoding algorithms for error-correcting codes. To this end, we explain in section 2.3 the notion of spin glasses and the associated free energy. In sections 2.4, 2.5, 2.6 and 2.7 we describe methods used to compute or estimate the free energy.

§ 2.2 Problems

In statistical physics, one of the main goals is to estimate thermodynamical functions of any system at equilibrium. Among them, the average energy U and the entropy S help us compute the Helmholtz free energy F:

$$F = U - TS \tag{2.1}$$

where T is the temperature set arbitrarily to 1 in the whole study as it simply sets a scale for the units in which one measures energy. This quantity is crucial because it provides knowledge of many other state functions and it is notably related to the partition function Z as:

$$F = -\ln Z \tag{2.2}$$

where Z provides the exact probability distribution of a physical network $\mathbf{S} = \{S_i\}_i$, *e.g.* spin glasses, given by the Boltzmann's law:

$$p(\mathbf{S} = \mathbf{s}) = \frac{e^{-E(\mathbf{s})}}{Z} \tag{2.3}$$

where $E(\mathbf{s})$ is the energy function. However, computing F is a very hard task as physical systems are often very large and interactions between elements of the system are very complex. More precisely there is no global solution but using equation (2.1), *i.e* researchers have no choice but to study all systems one by one.

Fortunately, physical systems are divided up into classes, as Sherrington-Kirkpatrick model, Edwards-Anderson model, etc. Each class has been studied to exhibit its own solving method. Among them we focus on the spin glasses because they are in a one-to-one match with graphs used in error-correcting coding theory.

§ 2.3 Spin glasses

From [Steo4],[M. 87],[B. oo], spin glasses are defined as systems of atoms with localized magnetic moments, namely *spins*, whose interactions are characterized by quenched randomness: a given pair of spins have a roughly equal *a priori* probability of having a ferromagnetic or an anti-ferromagnetic interaction, see Fig.2.1.

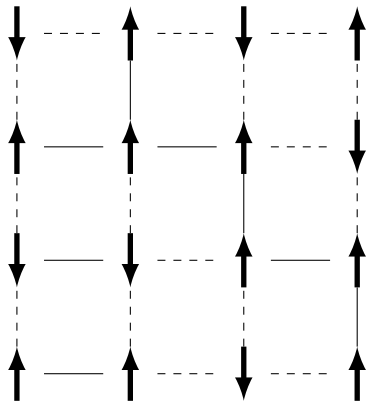


Figure 2.1: *Spin glass: the dashed lines are the anti-ferromagnetic interactions and the solid lines are the ferromagnetic interactions, the arrows are the spins (up or down)*

The prototype material is a dilute magnetic alloy, with a small amount of magnetic impurity randomly substituted into the lattice of a nonmagnetic metallic host. The signature feature of spin glass behavior is the phase transition, *i.e.* a dramatical change in the physical magnitudes of the system. The main interesting property of this kind of network, for our work, is that spin glasses suffer from *frustration*: no spin configuration can simultaneously satisfy all couplings. Such a phenomenon is observed also in channel coding when not

all parity-check equations of an error-correcting code are satisfied because the channel has changed the values of the bits.

According to [F. 12], the coupling J_{ij} between two spins s_i, s_j around a magnetic impurity has random sign and intensity because the distance between two spins is a random variable. The simplest idealization of the interaction between two spins s_i and s_j is a coupling term $-J_{ij}s_is_j$ in the energy function $E(\mathbf{s})$ with J_{ij} a random variable. The *disorder* of the material comes from the random values of the couplings $\{J_{ij}\}_{<i,j>}$. Due to the time independent valuation of these terms, the disorder is said *quenched*. The spin glass energy function without any external magnetic field is:

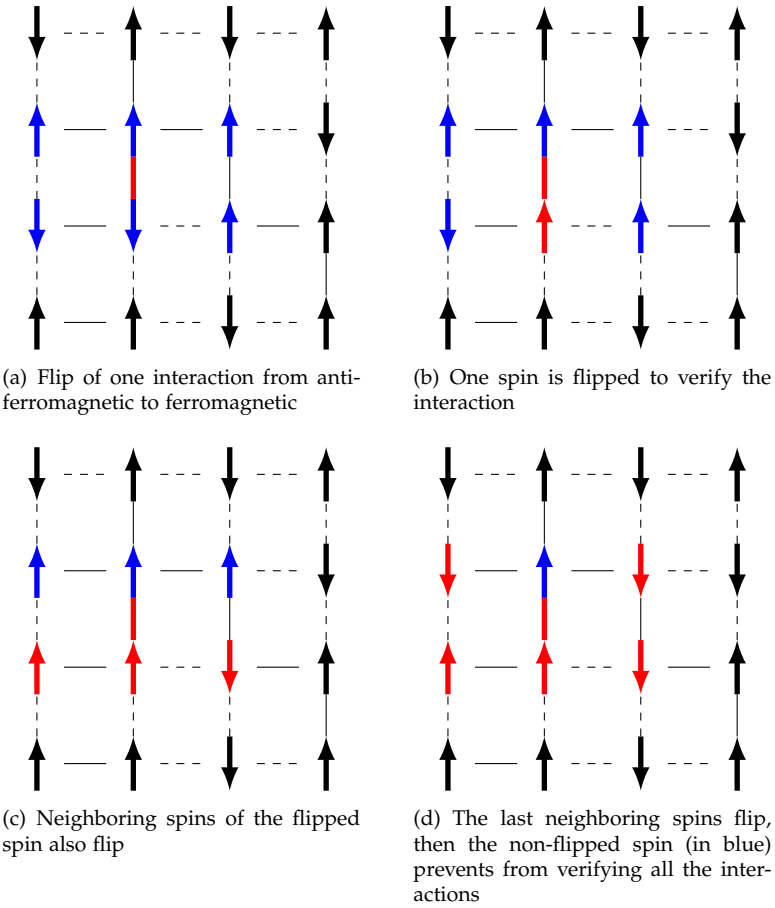
$$E(\mathbf{s}) = -\sum_{<i,j>} J_{ij}s_is_j \tag{2.4}$$

where the notation $<.,.>$ means neighboring sites. The spins variables $\{s_i\}_i$ are Ising variables *i.e.* they equal ± 1 (only pairwise interactions are considered but we will see in our research that are often needed high order interactions). If the terms $\{J_{ij}\}_{ij}$ are random variables, usually Gaussian variables, then the set of spins represent an *Edwards-Anderson model* [M. 06]. The infinite range model is called the *Sherrington-Kirkpatrick model* [D. 75].

As explained previously, *frustration* is a phenomenon that emerges because of the randomness of coupling constants: any spin is subjected to fields due to its neighbors that have different signs. Some want it to point up and others to point down then it cannot be set to a unique state. We represent this event in Fig.2.2.

The first figure shows a spin glass where one interaction has been modified such that it goes from anti-ferromagnetic to magnetic. Then on the other figures are depicted the route to the frustration: step by step, the spins flip (in red) to make the interactions, either ferromagnetic or anti-ferromagnetic, true. We then observe that the left blue spin should flip to be consistent with its left and right neighbors but in this case it prevents the interaction with its bottom neighbor from being true. Practically, if the spin flips then we again observe a step by step route to frustration that leads to the same situation. This is very close to the problem of loop-like topology in the case of the LDPC codes.

Frustration makes the search for the ground state not trivial, many degenerate ground states, or metastable states, emerge. Actually, a challenge of the physics is to find the state $\hat{\mathbf{s}}$ that provides the lowest value of the energy function $E(\mathbf{s})$. If all the J_{ij}'s were of same sign then the solution would appear very easily. However, in most cases, systems are *paramagnetic* the coupling constants have different signs. The naive though optimal method would consist in computing the energy function for all spins configurations, then we extract the lowest energy state. Unfortunately, if it could be done for low dimensional systems, it is completely intractable for high dimensional systems, given that 2^N states have to be computed (N being the number of spins). One would notice that this is actually closely related to the MLD that is also intractable for LDPC codes for the same reason. We here after present an alternative solution based on the Mean-Field (MF) approximation.



(a) Flip of one interaction from anti-ferromagnetic to ferromagnetic

(b) One spin is flipped to verify the interaction

(c) Neighboring spins of the flipped spin also flip

(d) The last neighboring spins flip, then the non-flipped spin (in blue) prevents from verifying all the interactions

Figure 2.2: *Spin glass: dashed lines are anti-ferromagnetic interactions and solid lines are ferromagnetic interactions; arrows are spins (up or down). The red line is an interaction modified by an external noise and red spins are spins modified by interactions.*

§ 2.4 Mean-Field approach

More than thirty five years ago, Sherrington and Kirkpatrick introduced this method to solve the spin glasses [D. 75],[M. 87]. Its relevance surely comes from the fact that it successfully attempts to represent some important features of the physical spin glass systems of great interest for their peculiar properties, at least at the level of the mean field approximation. A relevant summary of the MF with theory and results is exposed in [F. 06].

From [Kop10],[Nis01], the idea consists in developing the term $J\sum_{<i,j>} s_i s_j$ in the expression of the energy function, where J is the coupling assumed to be invariant on the couples of spins. We consequently use the mean spin value:

$$m_i =< s_i > . \tag{2.5}$$

Its value can be derived as:

$$\begin{aligned} m_i &= p_i(S_i = +1) \times (+1) + p_i(S_i = -1) \times (-1) && (2.6) \\ &= 2p_i(S_i = +1) - 1 && (2.7) \end{aligned}$$

Given that for each spin S_i the state is $s_i = m_i + \delta_i$, where δ_i is the fluctuation of the spin around its mean value, the cross product between two neighboring spins is:

$$s_i s_j = m_i m_j + m_i \delta_j + m_j \delta_i + \delta_i \delta_j \tag{2.8}$$

The MF approximation is to neglect the quadratic fluctuations $\delta_i \delta_j$, considering small enough in particular for high dimensional systems, *i.e.*:

$$s_i s_j \approx m_i m_j + m_i \delta_j + m_j \delta_i \tag{2.9}$$

Furthermore the mean value m_i of a spin s_i is expected to be site-independent, *i.e.* for any spin s_i, $< s_i >= m$. This yields:

$$s_i s_j \approx m^2 + m(\delta_j + \delta_i) \tag{2.10}$$

Using the definition of the fluctuation $\delta_i = s_i - m$, it appears that:

$$s_i s_j \approx -m^2 + m(s_i + s_j) \tag{2.11}$$

As the spins are valued in $\{-1, +1\}$ their sum is either 0 or $2s_i$. Thus the remaining term is:

$$s_i s_j \approx -m^2 + 2ms_i \tag{2.12}$$

The energy function is written such that all the couplings are considered equivalent:

$$E(\mathbf{s}) = -J \sum_{<i,j>} s_i s_j \tag{2.13}$$

Under the MF approximation it becomes:

$$E(\mathbf{s}) = JN_c m^2 - Jzm \sum_i s_i \tag{2.14}$$

where N_c is the number of interacting couples and z is said to be the *coordination number*. Considering an external magnetic field interacting with the spins through the energy $E_{ext} = -h \sum_i s_i$ we obtain the total MF approximation:

$$\boxed{E(\mathbf{s}) = JN_c m^2 - (h + Jzm) \sum_i s_i} \tag{2.15}$$

The whole system has then been decoupled into a sum of one-body system with an effective mean-field $h_{eff} = h + Jzm$, which is the sum of the external field h and of the mean-field induced by the neighboring spins. Using the Boltzmann's law it appears that the MF joint distribution is:

$$p_{MF}(\mathbf{x}) = \frac{1}{Z} e^{-JN_c m^2} \prod_{i=1}^{N} e^{h_{eff} s_i} \tag{2.16}$$

Finally we can write that the distribution is a factorization along the spins:

$$\boxed{p_{MF}(\mathbf{x}) = \frac{1}{Z_{MF}} \prod_{i=1}^{N} p_{MF,i}(x_i)} \tag{2.17}$$

The MF approximation provides analytic expressions of physical quantities as the partition function and the free energy, *i.e.* it solves the spin glasses.

2.4.1 Message-Passing

The MF approximation provides an algorithm to obtain the $\{m_i\}_i$. Given (2.16) and (2.17) we can write the MF marginal on the spin S_i as:

$$p_i(s_i) = \frac{1}{Z_i} e^{h + J \sum_{S_j \in \mathcal{N}_i} s_j} \tag{2.18}$$

Then the magnetization of the spin S_i is according to (2.6):

$$m_i = \tanh(h + \sum_{S_j \in \mathcal{N}_i} m_j) \tag{2.19}$$

A simple approach to obtain $\{m_i\}_i$ is iteratively updating them according to the following equation:

$$\boxed{m_i^{(k+1)} = \tanh(h + \sum_{S_j \in \mathcal{N}_i} m_j^{(k)})} \tag{2.20}$$

This message-passing results after K iteration in the computation of the marginals given that:

$$p_i(s_i = +1) = \frac{1 + m_i^{(K)}}{2}, \quad p_i(s_i = -1) = \frac{1 - m_i^{(K)}}{2} \tag{2.21}$$

Actually the MF expression of m_i is not as accurate as we would expect. Furthermore when averaged over all the configurations of the spin glass, $< m_i >$ is not easily handled because of the nonlinearity of the tanh function

$$< \tanh(h + \sum_{S_j \in \mathcal{N}_i} m_j) > \neq \tanh(h + \sum_{S_j \in \mathcal{N}_i} < m_j >)$$

that has been investigated, particularly in the TAP method proposed by Thouless, Anderson and Palmer [Mea01].

§ 2.5 Replica method

As mentioned in the introduction of the chapter, statistical physics is aimed at computing the free energy $F = -\ln Z$. It turns out that computing integer moments of the partition function Z^n with $n \in \mathbb{N}$ is much easier than computing the log-partition function. Fortunately, mathematics offer a way to use

this alternative by the following identity:

$$\ln Z = \lim_{n \to 0} \frac{Z^n - 1}{n} \tag{2.22}$$

Now we DESCRIBE, according to [M. 06],[Mea01], the basics of the replica method based on this identity. Z^n is a product of n identical partition functions. We consider a system $\mathcal{S}$ of N spins, represented by an energy function $E(\mathbf{s})$, that is independently duplicated into *n replicas*. The overall distribution of these n systems is:

$$p(\mathcal{S}_1 = \mathbf{s}_1, \ldots, \mathcal{S}_n = \mathbf{s}_n) = p_1(\mathcal{S}_1 = \mathbf{s}_1) \ldots p_n(\mathcal{S}_n = \mathbf{s}_n) \tag{2.23}$$

where the distribution of a system $\mathcal{S}_i$ verifies the Boltzmann's law, *i.e.*:

$$\begin{aligned} p(\mathcal{S}_1 = \mathbf{s}_1, \ldots, \mathcal{S}_n = \mathbf{s}_n) &= \frac{1}{Z_1} e^{-E(\mathbf{s}_1)} \ldots \frac{1}{Z_n} e^{-E(\mathbf{s}_n)} && (2.24) \\ &= \frac{e^{-(E(\mathbf{s}_1) + \cdots + E(\mathbf{s}_n))}}{Z_1 \ldots Z_n} && (2.25) \end{aligned}$$

We obtain a Boltzmann distribution with an energy function linearly decomposed over the different replicas of the system:

$$\tilde{E}(\mathbf{s}_1, \ldots, \mathbf{s}_n) = E(\mathbf{s}_1) + \cdots + E(\mathbf{s}_n) \tag{2.26}$$

and the partition function is:

$$\tilde{Z} = \sum_{\mathbf{s}_1} \cdots \sum_{\mathbf{s}_n} e^{-(E(\mathbf{s}_1) + \cdots + E(\mathbf{s}_n))} \tag{2.27}$$

For more details on this method that sparked off much interest, we refer the reader to the excellent reviews in [Nis01] and [M. 06].

§ 2.6 Cavity approach

The cavity method [F. 11] relies on the computation of the partition function Z by an iterative process. We will see that such an algorithm is completely equivalent to the BP as it propagates messages between nodes of the graph. That is why we will describe it in a more detailed way than the other methods.

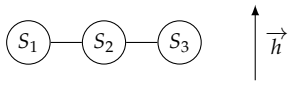


Figure 2.3: *Ising chain with $N = 3$ spins embedded in an external magnetic field*

The idea is to compute the partition function recursively, by considering a dynamical graph: at each step is added a variable node to the graph, the graph being completely recovered at the last step. Equivalently one may think of taking one variable node out of the system and computing the change in the

partition function. The name of the method comes exactly from this image: one digs a cavity in the system.

2.6.1 Ising chain

As for BP, the original algorithm is defined on trees. Let's consider an Ising chain, *i.e.* a pairwise tree whose spins are binary variables and of degree ≤ 2, with a small number of spins to develop the equations, see Fig.2.6. We now derive the equations step by step given that two spins S_i, S_j interact via the interaction $-JS_iS_j$ and that a spin S_i interacts with the magnetic field via the energy $-h_iS_i$.

Step 1: spin S_1

The tree is made with spin S_1 only and field $\overrightarrow{h}$.

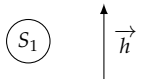


Figure 2.4: $N = 1$

Then the partition function is:

$$Z^{(1)} = \sum_{s_1} Z_1(s_1) \tag{2.28}$$

with $Z_1(s_1)$ the partial partition function with the spin S_1 fixed to the value s_1:

$$Z_1(s_1) = e^{h_1 s_1} \tag{2.29}$$

Step 2: spins S_1 and S_2

The tree is made with the spin S_1 and the spin S_2 and the field $\overrightarrow{h}$.

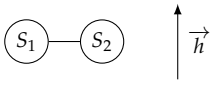


Figure 2.5: $N = 2$

Then the partition function is:

$$Z^{(2)} = \sum_{s_2} Z_2(s_2) \tag{2.30}$$

with $Z_2(s_2)$ the partial partition function with the spin S_2 fixed to the value s_2:

$$Z_2(s_2) = e^{h_2 s_2} \sum_{s_1} e^{J s_1 s_2} e^{h_1 s_1} \tag{2.31}$$

In this equation we recognize $Z_1(s_1)$ defined earlier. However the context is different given that we added a variable to the system. Therefore $Z_1(s_1)$ is not

anymore the partial partition function with S_1 fixed to the value s_1. In order to make it clear we denote the former $Z_1(s_1)$ by $Z_{1\to 2}(s_1)$. Then:

$$Z_2(s_2) = e^{h_2 s_2} \sum_{s_1} e^{J s_1 s_2} Z_{1\to 2}(s_1) \tag{2.32}$$

Step 3: spins S_1, S_2 and S_3

The tree is made with the spin S_1 and the spin S_2 and the field $\overrightarrow{h}$.

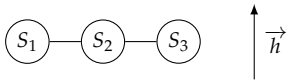


Figure 2.6: $N = 3$

The partition function is:

$$Z^{(3)} = \sum_{s_3} Z_3(s_3) \tag{2.33}$$

with $Z_3(s_3)$ the partial partition function with S_3 fixed to the value s_3:

$$Z_3(s_3) = e^{h_3 s_3} \sum_{s_2} e^{J s_2 s_3} e^{h_2 s_2} \sum_{s_1} e^{J s_1 s_2} e^{h_1 s_1} \tag{2.34}$$

Extending the same previous argument, we now denote the former $Z_2(s_2)$ by $Z_{2\to 3}(s_2)$, that provides:

$$Z_3(s_3) = e^{h_3 s_3} \sum_{s_2} e^{J s_2 s_3} Z_{2\to 3}(s_2) \tag{2.35}$$

Interpretation

We observe for each step i, the partial partition function $Z_i(s_i)$ is a product of information coming from the the magnetic field

$$e^{h_i s_i}$$

with the information coming from its neighbor S_j revealed at the previous step

$$\sum_{s_j} e^{J s_i s_j} Z_{j\to i}(s_j)$$

Generally for any tree model, not necessarily a chain, the partial partition function of any spin S_i fixed to the value s_i is computed as:

$$Z_i(s_i) = e^{h_i s_i} \prod_{S_j \in \mathcal{N}_i} \sum_{s_j} e^{J s_i s_j} Z_{j\to i}(s_j) \tag{2.36}$$

where $\mathcal{N}_i$ is the set of spins connected to S_i. The quantity $Z_{j\to i}(s_j)$ is the partial partition function of the system without the spin S_i with S_j fixed to s_j. One could verify this way that we obtain that $\sum_{s_i} Z_i(s_i)$ is strictly identical to the total partition function Z.

2.6.2 Case of a factor graph

The previous claim seems to only address pairwise trees, but it also applies to any model such that the energy function is of the following form:

$$E(\mathbf{s}) = -\sum_i h_i s_i - \sum_a J \prod_{S_i \in \mathcal{N}_a} s_i \tag{2.37}$$

where $\sum_a$ is the sum over all the spins interactions, and $\mathcal{N}_a$ is the set of spins that interact through a common interaction denoted by a. Let's consider the simple example on Fig.2.7 to develop the cavity method.

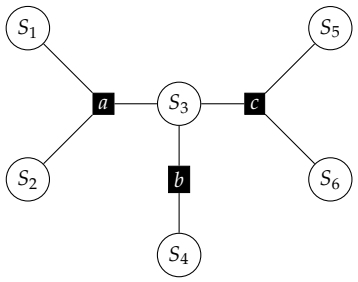


Figure 2.7: *p-range model*

The partial partition function for the spin S_1 in the state s_1 is:

$$Z_1(s_1) = e^{h_1 s_1} \sum_{s_2} \sum_{s_3} e^{J s_1 s_2 s_3} Z_{2\to_a 1}(s_2) Z_{3\to_a 1}(s_3) \tag{2.38}$$

where:

$$\begin{aligned} Z_{2\to_a 1}(s_2) &= e^{h_2 s_2} && (2.39)\\ Z_{3\to_a 1}(s_3) &= e^{h_3 s_3} && (2.40)\\ &\times \sum_{s_4} e^{J s_3 s_4} Z_{4\to_b 3}(s_4) && (2.41)\\ &\times \sum_{s_5} \sum_{s_6} e^{J s_3 s_5 s_6} Z_{5\to_c 3}(s_5) Z_{6\to_c 3}(s_6) && (2.42) \end{aligned}$$

and:

$$\begin{aligned} Z_{5\to_c 3}(s_5) &= e^{h_5 s_5} && (2.43)\\ Z_{6\to_c 3}(s_6) &= e^{h_6 s_6} && (2.44) \end{aligned}$$

The subscript a in $Z_{2\to_a 1}$ helps understand "from S_2 to S_1 through a". One could easily verify that we get the exact total partition function by computing:

$$\sum_{s_1} Z_1(s_1)$$

2.6.3 Bipartite messages

In order to simplify the implementation, above equations could be sorted into two kinds. To understand how, one should have a look to the equations of $Z_{i\to\alpha j}(s_i)$. Since such a quantity is a partial partition function for the system without spin S_j, it should not depend on S_j. In the previous example, one would check that:

$$Z_{3\to a1}(s_3) = Z_{3\to a2}(s_3) \tag{2.45}$$

In this case, the spins S_1 and S_2 do not play any role in the computation, therefore both quantities should be computed as a single *message* from the spin S_3 to the interaction, called *factor*, a. Generally we have for two spins S_i, S_j interacting through the factor a:

$$\begin{aligned} Y_{i\to a}(s_i) &\triangleq Z_{i\to aj}(s_i) && (2.46)\\ &= e^{h_i s_i} \prod_{\alpha\in\mathcal{N}_i\setminus a} \sum_{s_{\alpha_1}}\sum_{s_{\alpha_2}}\cdots\sum_{s_{\alpha_m}} e^{J s_{\alpha_1} s_{\alpha_1}\ldots s_{\alpha_m}} \prod_{S_j\in\mathcal{N}_\alpha} Y_{j\to\alpha}(s_j) && (2.47)\end{aligned}$$

where $\{S_{\alpha_1},\ldots,S_{\alpha_m}\}$ is the set of spins interacting via factor α. The second kind of message that we define is the reverse of $Y_{i\to a}(s_i)$, *i.e.* the message that comes from a to S_i. The equations of the previous case provide that:

$$\boxed{W_{a\to i}(s_i) \triangleq \sum_{s_{a_1}}\sum_{s_{a_2}}\cdots\sum_{s_{a_m}} e^{J s_{a_1} s_{a_2}\ldots s_{a_m}} \prod_{S_j\in\mathcal{N}_a} Y_{j\to a}(s_j)} \tag{2.48}$$

Then we can rewrite $Y_{i\to a}(s_i)$ as:

$$\boxed{Y_{i\to a}(s_i) = e^{h_i s_i} \prod_{\alpha\in\mathcal{N}_i\setminus a} W_{\alpha\to i}(s_i)} \tag{2.49}$$

In this way it is possible to give the partial partition function for a spin S_i in the state s_i as:

$$\boxed{Z_i(s_i) = e^{h_i s_i} \prod_{a\in\mathcal{N}_i} W_{a\to i}(s_i)} \tag{2.50}$$

Equations (2.48), (2.49) and (2.50) are message-passing equations, analogous to those BP equations we mentioned in the previous chapter such that:

- $W_{a\to i}$ is analogous to the message $m_{a\to i}$ from the check node c_a to the variable node X_i,
- $Y_{i\to a}$ is analogous to the message $n_{i\to a}$ from the variable node X_i to the check node c_a,
- Z_i is analogous to the belief b_i on the variable node X_i.

It helps define the equivalence between the cavity method and BP. As a notice, the cavity method can also be expressed as a message-passing either between the magnetizations of the spins or the probability distributions of the spins, see [F. 11] for details about this.

§ 2.7 Mean-Field generalizations

In chapters 4 and 5 we will present improvements of the MF approach, namely the Bethe approximations and the region-based approximations [J.S02],[J.S05]. The common principle of both methods is to generalize the equation (2.17) to the following:

$$p(\mathbf{x}) = \frac{1}{Z_{\mathcal{R}}} \prod_{r \in \mathcal{R}} p_r^{c_r}(\mathbf{x}_r) \tag{2.51}$$

Instead of factorizing the joint distribution on single variable nodes, we factorize on subsets or *regions* of variable nodes. This way, we keep, to some extent, the couplings between the variables mentioned in the energy function. These methods are actually part of the global *region-based free energy approximation* detailed in the chapter 5.

§ 2.8 Conclusion

The MF approximation is a relevant way to provide tractable computation of the thermodynamical functions. We saw that it can also be expressed as a message-passing algorithm between the spins of any spin glass. Similarly, the cavity approach has been showed to present the same equations as the BP's, highlighting a strong connection between statistical physics and decoding algorithms. As such, this method is also subject to the loop-like harmful effects of the spin glasses, or any other physical factor graphs. To our knowledge, the replica method has not been extended to decoding algorithms, partly due to its quite high complex theoretical content. Other methods that we did not describe here are also investigated in this respect, *e.g.* the renormalization group [G. 10],[J. 03b].

We will see in the next part that these methods can surely be improved by considering less restrictive decoupling between spins. That is the rationale behind MF generalizations, also known as Bethe and region-based approximations.

- Chapter 3 -

Dynamical systems

§ 3.1 Introduction

In this chapter, we present a few tools that help study dynamical systems. We cannot describe all the extent of the theory, therefore we will focus on the ones that are to be used on the decoding algorithms that are nonlinear systems due to the nonlinearities involved by the update rules. Linear systems are not surprising as they always give linear response, *i.e.* their effects are proportional to the causes. On the contrary, evolutions of nonlinear systems turn out to be often unexpected, though their update equations are deterministic. Nonlinearities can make saturate a system or even make it diverge.

Study of so called *dynamical systems* results from the fact that most systems evolve in time and that they need to be reliably anticipated. The property of *stability* remains central in the study of any nonlinear system. It is qualitatively for a system to keep a regular and controlled evolution when something is slightly changed in its initial conditions.

Dynamical systems are defined by *state variables* which time evolutions, or state equations, provide the description of the systems. These evolutions are logically defined by differential equations when variables are continuous or iterative equations when variables are discrete in time. Complex and often unexpected behaviors of nonlinear systems do not appear obvious even though state equations are completely deterministic. As a matter of fact, one could think that the lack of control would come out if the state equations followed some probabilistic laws, because it somehow represents a lack of knowledge. Actually, we will see that strange behaviors do not need such a property but only a series of causes and effects submitted to particular initializations and parameters values. This is partly due to the absence of analytic solution to the state equations but only numerical ones. Consequently, a slight difference in the numerical values of few state variables or parameters can lead to massive effects in the evolutions of the system, such as *chaos*. In such a case there is no predictability and no control on the system, the evolution seems to be erratic and random. Such a sensitivity is inherent to the system itself. Although the purpose of the investigation is to predict the states of any nonlinear system at

any time, chaos prevents us from completely mastering it. Therefore, a crucial study is to bring out the main properties of chaos, particularly the minimal conditions under which it occurs.

In section 3.1, we describe the notion of chaos with two well-known examples, the logistic map and the Lorenz system. Section 3.2 is dedicated to the different scenarios that lead any nonlinear system to a chaotic behavior. In section 3.3, we detail the computation of the Lyapunov exponent that numerically estimates whether any system is chaotic or not. Last section 3.4 deals with the situations where a dynamical system is incomplete, *i.e.* a few state variables are not available. We briefly present a method that re-constructs the state space.

§ 3.2 Chaotic attractors

It is crucial to understand that *chaos* is neither random nor nonsense. More accurately, chaos is basically opposed to randomness as a chaotic system is a deterministic system whose time evolution is perfectly described by the state variables and the associate update rule(s). Practical systems are subject to chaos because of our incapacity to analytically solve them and to compute state variables with an infinite precision. More precisely, an infinite precision on data would allow us to forecast any state of a system given its initial state whereas a finite precision implies an uncertainty, or computation error, after some computation time.

From [K.T96], chaos is a specific behavior defined as follows: when the system evolution could be done through numerous motions that are initially quite close in the state space, such that after some time, any two of them seem to have nothing in common except they are enclosed in the same region of the given state space (they are dissipative). The separation between them turns out to be often exponential in time. This phenomenon is an intrinsic property of the *chaotic attractors*, also called *strange attractors*. To illustrate this peculiar behavior, we will use two well-known examples: the logistic map that models population growth and Lorenz attractor that models – to some extent – the atmospheric convection.

3.2.1 Logistic map

A common model that represents the evolution $\{x_k\}_k$ of a resource-limited population is given by:

$$x_{k+1} = ax_k(1 - x_k) \tag{3.1}$$

with a a positive real number and $x_0 \in]0, 1[$. It appears that a very small value of x_k entails a quite linear growing because $1 - x_k \approx 1$ whereas a large value of x_k prevents from a proportional growing. In other words, the maps is nonlinear for non small values of x_k. Such a nonlinearity is the source of non trivial behaviors presented hereafter. We restrict the study to the case $a = 4$ but anyone who is interested in extracting the whole information about this system should investigate on all values of a. The first step of the study is to visualize the sequence $\{x_k\}_k$. A common method is displayed in Fig.3.1.

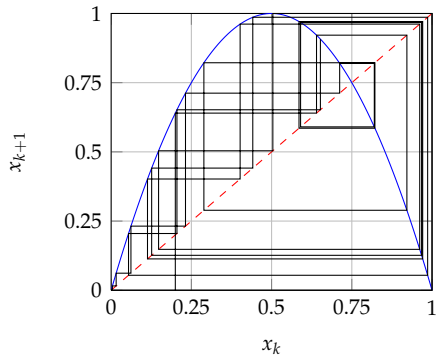


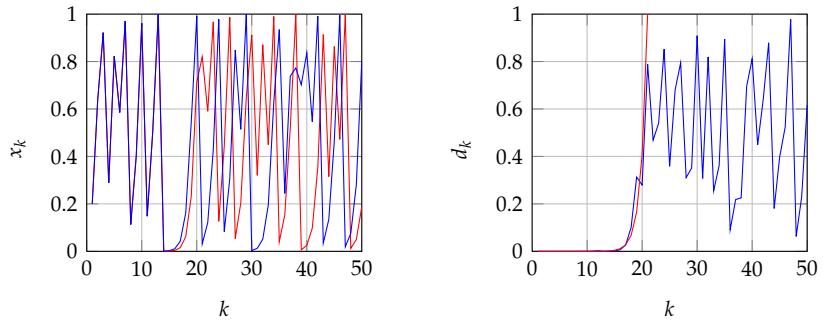
Figure 3.1: *Logistic map* $x_{k+1} = 4x_k(1 - x_k)$

We observe that no particular pattern emerges, we cannot distinguish any fixed-point or repeated motion. This is also exhibited in Fig.3.2(a) where a trajectory of the map is represented, with the time evolution for a slightly different initial condition $x_0' = x_0 + 10^{-6}$. At $k = 14$ appears the beginning of the divergence between both trajectories and as the time increases they evolve completely differently. In Fig.3.2(b) is represented the time evolution of the separation between these trajectories. For $k \leq 21$ it is possible to model it by the function:

$$\hat{d}(k) = e^{\lambda(k-\alpha)}$$

with:

$$\begin{aligned} \lambda &\approx 0.9, \\ \alpha &\approx 21. \end{aligned}$$



(a) Trajectories (b) Separation

Figure 3.2: *Logistic map*

Coefficient α is a delay and λ is a crucial quantity that provides the intensity of the divergence between trajectories. Given that the exponential function takes precedence over any polynomial function, an exponential evolution of the separation is the most divergent motion that can be encountered.

Such a property is an evidence of chaos: two initially close trajectories eventually follow very non similar evolutions. Quantity λ is commonly called the *largest Lyapunov exponent*, or simply the Lyapunov exponent.

In Fig.3.2(b) it appears that $\hat{d}(k)$ is not exponential for $k > 21$. First of all, it is impossible for the separation to acquire values that would be larger than the state space size itself, therefore $\hat{d}(k) \leq 1$ for all k. After that, the trajectories of any dissipative systems live in a restricted part of the state space, named the attractor. This property leads us to evaluate the attractor size by measuring the maximum of $\hat{d}(k)$. We will study this more deeply in the last chapter of the manuscript.

3.2.2 Lorenz attractor

In climate science, the Lorenz attractor helps predict long term atmospheric aspects. Once more, slight difference in numerical initial conditions leads, under particular assignments of external parameters, to very different behaviors. The Ordinary Differential Equations (ODE) that model the Lorenz system are:

$$\frac{dx(t)}{dt} = \sigma(y(t) - x(t)), \tag{3.2}$$

$$\frac{dy(t)}{dt} = rx(t) - y(t) - x(t)z(t), \tag{3.3}$$

$$\frac{dz(t)}{dt} = x(t)y(t) - bz(t), \tag{3.4}$$

where $x(t), y(t), z(t)$ are the state variables.

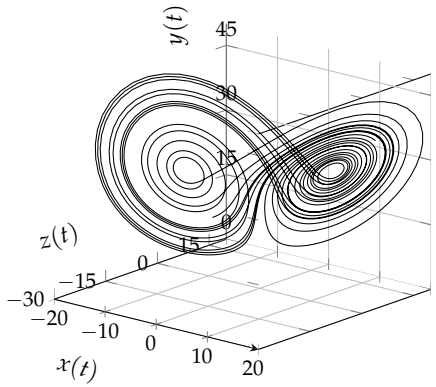


Figure 3.3: *Lorenz attractor*

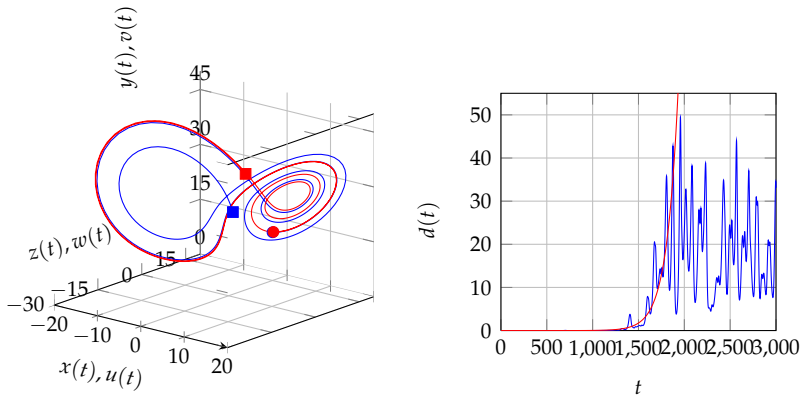
As mentioned previously, a detailed study would require many values of the parameters σ, r, b. In this context, we chose values that provide non trivial phenomena:

$$\sigma = 10, \tag{3.5}$$

$$b = \frac{8}{3}, \tag{3.6}$$

$$r = 28. \tag{3.7}$$

This set of values traps the system into a non trivial attractor as depicted in Fig.3.3. This attractor is the chaotic Lorenz attractor. The sensitivity to initial conditions is very high, see Fig.3.4(a). The second trajectory is initially 10^{-6} away from the first one at the Euclidean sense.



(a) Trajectories (b) Separation

Figure 3.4: *Lorenz map*

The full circles are the initial points and the full squares are the final points. Even though circles are very close, squares are quite far away one from the other, divergence appears substantial. This assumption is confirmed by Fig.3.4(b) where the Euclidean distance $d(t)$ along the time $t \leq 2000$ can be modeled by an exponential function:

$$\hat{d}(t) = e^{\lambda(t-\alpha)}$$

with:

$$\begin{aligned} \lambda &\approx 0.008 \\ \alpha &\approx 1430 \end{aligned}$$

From the brief study on the logistic map and the current one on the Lorenz system, the common point is that the chaos appears as soon as the quantity λ is positive. In other words, the sign of the Lyapunov exponent is the signature of chaos.

§ 3.3 Route to chaos

An aspect that needs investigation is the scenario that is responsible for chaos, in order to find out the situations where it can appear. Actually the chaos theory lists several *routes to chaos* that are sequences of *bifurcations*, a bifurcation being any sudden change in the behavior of a system as some parameter is varied [Hiloo].

3.3.1 Period-doubling

For a given value of an external parameter β, the system originally oscillates with a period T. Increasing β makes the oscillation period doubled *i.e.* $T \leftarrow 2T$. It is actually due to the insertion of a new pattern inside the former periodic one. A larger value of β again doubles the period $T \leftarrow 2T$ etc. After a certain time we observe that the system is trapped into a completely non periodic motion. As an illustration is displayed the evolution of the period T of the logistic map according to the parameter a in Fig.3.5.

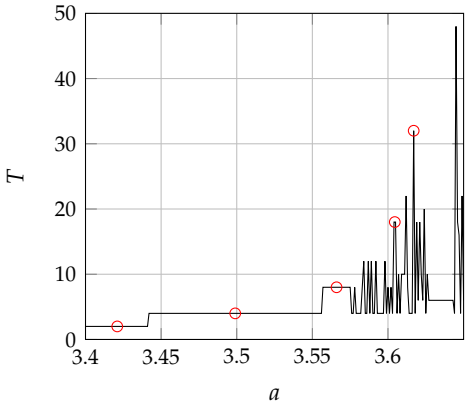
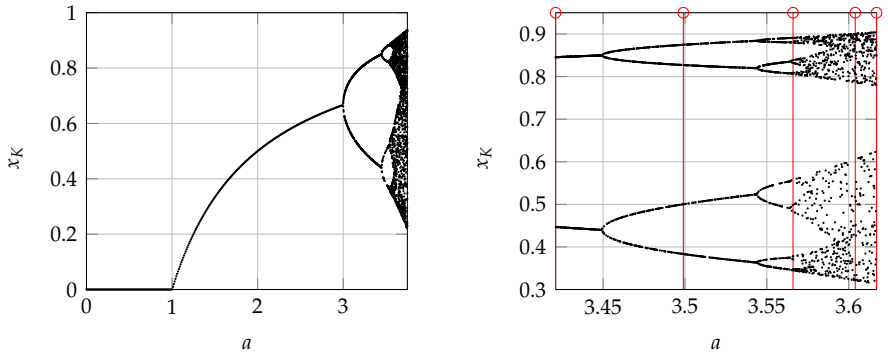


Figure 3.5: *Logistic map - Period doubling*

Even though we can pick a few inter values of the period, the main behavior is such that the period is doubling as a increases. The circles are typical values, also represented by stems in Fig.3.6(b), such that:

$$\begin{aligned} a = 3.421 &\implies T = 2 \\ a = 3.499 &\implies T = 4 \\ a = 3.566 &\implies T = 8 \\ a = 3.604 &\implies T = 16 \\ a = 3.617 &\implies T = 32 \end{aligned}$$

The period-doubling process may continue until the period becomes infinite, that is the trajectory never repeats itself, it is then chaotic. We display in Fig.3.6(a) the bifurcation diagram, *i.e.* the final states x_K ($K = 100$), of the logistic map along the values of a. We observe the first bifurcation that results in oscillations for $a = 3$ and the next bifurcations that double the period in Fig.3.6(b).



(a) Whole range of a (b) Period-doubling

Figure 3.6: *Bifurcation diagram of the logistic map*

3.3.2 Quasi-periodicity

We use the Tinkerbell map to illustrate the quasi-periodicity and the associated route to chaos. Such a map is defined by the following state equations:

$$x_{k+1} = x_k^2 - y_k^2 + ax_k + by_k \tag{3.8}$$
$$y_{k+1} = 2x_k y_k + cx_k + dy_k \tag{3.9}$$

and we choose parameters to be:

$$\begin{aligned} b &= -0.6013, \\ c &= 2, \\ d &= 0.5. \end{aligned}$$

The value of the parameter a will play the role of a potentiometer that makes quasi-periodicity appear, among others.

Poincaré section Actually this two-dimensional discrete map can be thought of as a reduced three-dimensional continuous map. This assumption comes from the *Poincaré section* method. The Poincaré section is somehow a stroboscopic portrait of the continuous trajectory, as displayed in Fig.3.7.

The Poincaré section provides a discrete representation $P_1, P_2, \ldots, P_K$ of the continuous system whose state space dimension is smaller than the original one. Formally, a system like:

$$\begin{aligned} \dot{X} &= f(X, Y, Z) \\ \dot{Y} &= g(X, Y, Z) \\ \dot{Z} &= h(X, Y, Z) \end{aligned}$$

is mapped to an iterative reduced system:

$$P_{k+1} = F(P_k)$$

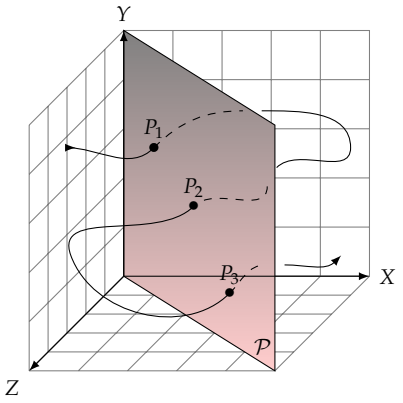


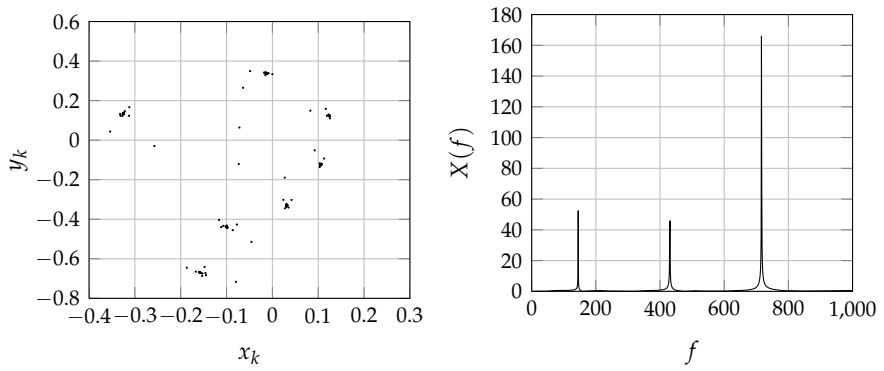
Figure 3.7: *Poincaré section $\mathcal{P}$*

where $P_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix}$ such that:

$$\begin{aligned} x_{k+1} &= F_x(x_k, y_k), \\ y_{k+1} &= F_y(x_k, y_k). \end{aligned}$$

The iterative functions F_x, F_y are defined by the position of the Poincaré section in the three-dimensional state space.

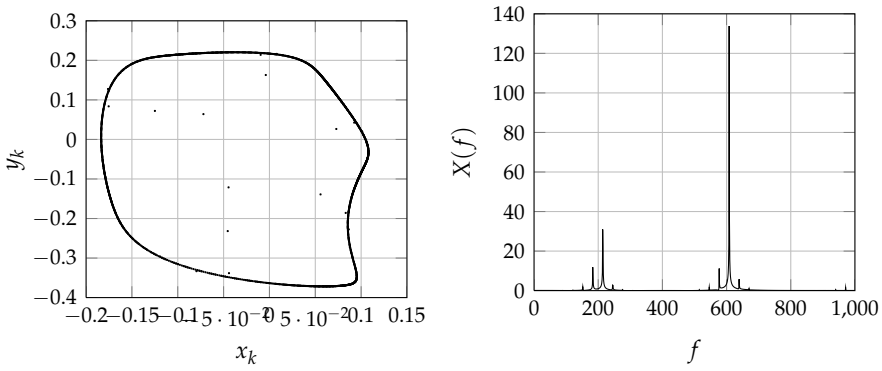
A periodic three-dimensional continuous system looks like a continuous line in its state space, therefore the projection on the Poincaré section is a finite number of points. For $a = 0.5$, the Tinkerbell map follows this rule, see Fig.3.8(a). The power spectrum Fig.3.8(b) of the sequence $\{x_k\}_k$ reveals a single frequency f_0 and its harmonics $2f_0, 3f_0$.



(a) Poincaré section (b) Power spectrum

Figure 3.8: *Tinkerbell attractor with $a = 0.5$*

Quasi-periodic attractor A quasi-periodic motion presents two or more different frequencies. These frequencies are such that the three-dimensional continuous system is a torus and the Poincaré section is generated by using a plane that cuts through the torus. If the ratio of any two frequencies can be expressed as a rational fraction then the Poincaré section will consist of a finite number of points. In other words, this is a periodic motion. However, if only one ratio of two frequencies cannot be expressed as a ratio of integers, then the ratio is irrational. For this case, the Poincaré map points will eventually fill in a continuous curve, the motion is said to be quasi-periodic because the motion never exactly repeats itself. The points of the Poincaré section drift around the curve forming what is called a drift ring. As an example, we consider the Tinkerbell map for $a = 0.1$, see Fig.3.10(a). We observe two leading spectral lines on the spectrum Fig.3.10(b) whose associated frequencies f_1, f_2 are such that the ratio is not "strictly" rational. Actually, given that any computer has a finite precision, it is always possible to extract a rational ratio between two arbitrary values, that makes the quasi-periodicity quite hard to detect. For instance, the current values of the two frequencies can be linked such that $f_2 \approx \frac{20}{7} f_1$.



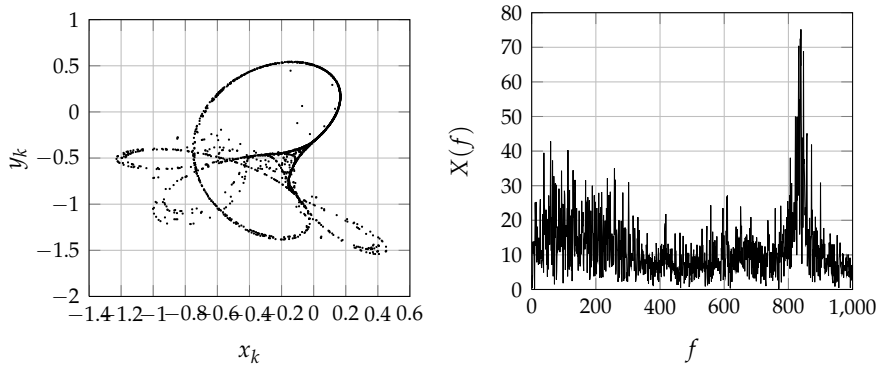
(a) Poincaré section

(b) Power spectrum

Figure 3.9: *Tinkerbell attractor with* $a = 0.1$

A crucial property of the quasi-periodic case is that it is not chaotic. The confusion could be made considering that the chaotic motion also provides a quite continuous curve on the Poincaré section, as we can see with the Tinkerbell map with $a = 0.9$ in Fig.3.10(a). The difference comes from the fact that a quasi-periodic behavior is composed of several periodic components then the spectrum contains different spectral lines whereas in a chaotic motion, the spectrum involves a continuum of frequencies, see Fig.3.10(b).

Ruelle-Takens scenario or quasi-periodicity route to chaos The system begins with a limit cycle. As a control parameter is changed, a second periodicity appears in the behavior of the system. This bifurcation event is a Hopf bifurcation. If the ratio of the period of the second type of motion to the period of the first is not a rational ratio, then the motion becomes quasi-periodic.



(a) Poincaré section (b) Power spectrum

Figure 3.10: *Tinkerbell attractor with* $a = 0.9$

Under some circumstances, if the control parameter is change further the motion becomes chaotic. One would expect to see a long sequence of different frequencies come in as the control parameter is changed, much like the infinite sequence of period-doubling. However at least in some cases, the system becomes chaotic instead of introducing a third frequency for its motion. In [S.E78] the authors proved that if the trajectory of a system is confined to a three-dimensional torus, *i.e.* it is made of three frequencies with irrational ratios, then even a small perturbation of the motion will damage the motion and lead to chaos.

3.3.3 Intermittency and crises

The intermittency route to chaos is characterized by dynamics with irregular occurring bursts of chaotic behavior interspersed with intervals of apparently periodic behavior. As a control parameter is changed, the chaotic bursts become longer and occur more frequently until, eventually the entire time record is chaotic.

A crisis is a bifurcation event in which a chaotic attractor and its basin of attraction suddenly disappear or suddenly change in size as some control parameter is changed. Alternatively is the parameter is changed in the opposite direction the chaotic attractor can suddenly appear or the size of the attractor can suddenly be reduced. A crisis event involves the interaction between a chaotic attractor and an unstable fixed point or an unstable limit cycle.

§ 3.4 Lyapunov exponent

If the system is behaving chaotically, the divergence of nearby trajectories will manifest itself in the following way: if we select some value x_i from the sequence $\{x_k\}_k$ and then search the sequence for another value x_j that is close

to x_i then the sequence of differences:

$$\begin{aligned} d_0 &= |x_i - x_j| \\ d_1 &= |x_{i+1} - x_{j+1}| \\ &\vdots \\ d_k &= |x_{i+k} - x_{j+k}| \end{aligned}$$

is assumed to increase exponentially from one step k to the next one $k+1$, *i.e.* $d_{k+1} = d_k e^{\lambda}$ that could be also written:

$$d_k = d_0 e^{\lambda k} \tag{3.10}$$

where λ is the Lyapunov exponent. According to this assumption we can extract two methods to compute λ.

3.4.1 Computation method 1

The equation (3.10) stipulates that the evolution of $\log d_k$ follows an affine law:

$$\log d_k = \lambda k + \log d_0 \tag{3.11}$$

where λ is the slope of the line. A well suited tool to estimate this slope is the method of least squares. According to the theory we can easily show that the estimate of the slope is:

$$\boxed{\hat{\lambda} = \frac{\sum_{k=1}^{K} k \log d_k - \frac{1}{K}\left(\sum_{k=1}^{K} k\right)\left(\sum_{k=1}^{K} \log d_k\right)}{\sum_{k=1}^{K} k^2 - \frac{1}{K}\left(\sum_{k=1}^{K} k\right)^2}} \tag{3.12}$$

where K is the index of the last sample. Obviously, it is crucial to average this value over a large set of initial conditions in order to obtain a quite faithful value that depends much more on the parameters of the system than on its initial states.

3.4.2 Computation method 2

The value d_k is the separation between the k-th iterate of the map f from x_i and the k-th iterate of f from x_j. Given that x_i and x_j are very close, it is more convenient to consider $x_i \triangleq x_0$ and $x_j \triangleq x_0 + \epsilon$ where $\epsilon \ll 1$. Thus:

$$d_k = |f^k(x_0 + \epsilon) - f^k(x_0)| \tag{3.13}$$

Therefore:

$$\frac{d_k}{\epsilon} = \frac{|f^k(x_0 + \epsilon) - f^k(x_0)|}{\epsilon} \tag{3.14}$$

In this way, it appears that:

$$\lim_{\epsilon \to 0} \frac{d_k}{\epsilon} = (f^k)'(x_0) \tag{3.15}$$

First of all, the derivatives of any function f^k can be decomposed as:

$$(f^k)'(x_0) = f'(x_k)f'(x_{k-1}) \dots f'(x_0)$$

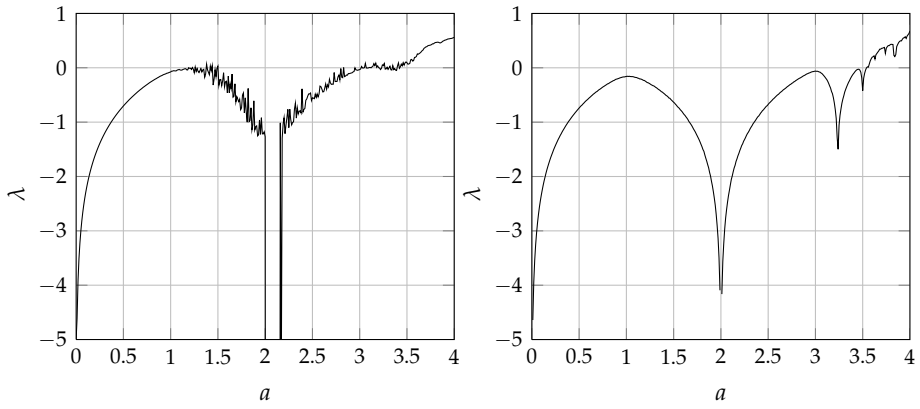
After that, the assumption that the separation evolves exponentially implies that:

$$(f^k)'(x_0) = e^{\lambda k}$$

i.e.

$$\boxed{\lambda = \frac{1}{K} \sum_{k=0}^{K} \log |f'(x_k)|} \tag{3.16}$$

In other words, the Lyapunov exponent λ is the log-average of the slopes over all the samples $x_0, \dots, x_K$. On Fig.3.11 are displayed the estimate of λ from both methods.



(a) Least squares method (b) Differential method

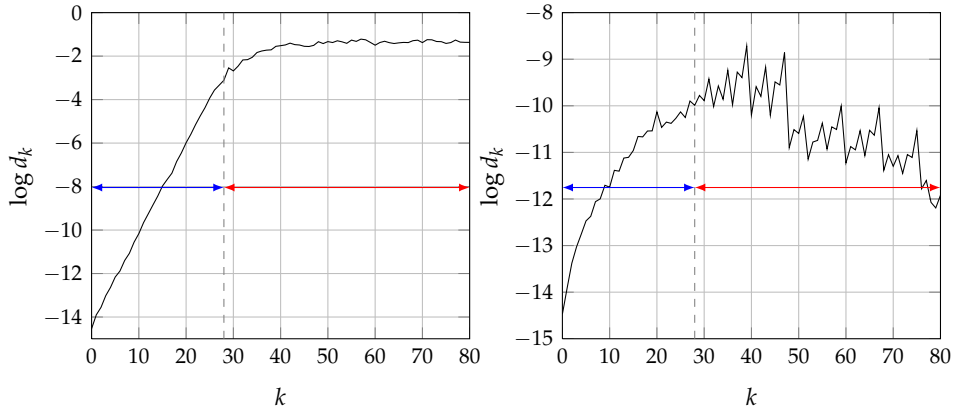
Figure 3.11: *Lyapunov exponent estimates*

Even though we can observe few differences, *e.g.* at $a = 3.24$ the least squares method does not provide a λ as low as the differential method does, both methods offer estimates of a global same shape. On one hand, the least squares method presents variance for $a \in [1.24, 3.48]$ but it is possible to compute λ for any systems, whatever the complexity of its state equations. On the other hand, the differential method is smooth for the whole a interval however it is not always convenient given that the derivatives of the map, or even of the set of maps in the multidimensional case, is not an easy computation task. As an example, it is not well suited for the iterative decoding algorithms as their update rules are sometimes very hard to handle, see chapter 5.

3.4.3 Assumptions

Both computation methods assume two main hypothesis:

- the number of samples K should be large enough to obtain accurate averages,
- the number of samples K should be small enough such that d_K can still be approximated by $d_0 e^{\lambda K}$. For any dissipative system, any attractor whether chaotic or periodic (or quasi-periodic), has a finite size inside the state space. The maximum value of the separation is the attractor size itself. Waiting for too large K iterations would lead to a wrong estimation given that the exponential law of d_K is not valid anymore.



(a) Exponential evolution of the separation

(b) Fake chaos, $K = 80$

Figure 3.12: *Time evolution of the separation*

On Fig.3.12(a) is displayed the evolution of $\log d_k$ for the logistic map with $a = 3.75$. The k values are shared into two sets: for the interval $[0, 28]$ indexed with the blue arrow $\log d_k$ is linear then the approximation is right, for the interval $[29, 80]$ indexed with the red arrow $\log d_k$ is constant to the size of the chaotic attractor then the approximation does not make sense anymore. The maximum iteration to use is then $K = 28$. However, the ideal value K is not always obvious, especially for multidimensional systems. The initial conditions coupled with external parameters largely wield influence on the behavior of the system, *e.g.* the size of the attractor or even its nature (fixed point, periodic, quasi-periodic or chaotic). Therefore finding the value K turns out to be a very hard task.

Furthermore, the system can mimic a chaotic behavior for few iterations and then it collapses to a very small sized attractor or even a fixed point. We display such a phenomenon, the *transient chaos* [Hiloo], in Fig.3.12(b) for the logistic map with $a = 3.56$. In such a case, if we still consider that $K = 28$ then we conclude that the logistic map is chaotic whereas it is absolutely not. Waiting for larger K would lead to the better conclusion that the logistic map has suffered from a temporary almost chaotic behavior. The previous

observation about the versatility of K and the current one about the temporary chaos lead us to consider that neither computation method is perfect.

3.4.4 Computation method 3

The issue is to model by a single value the whole behavior of the system along the iterations for any parameter value. The most convenient solution that can be computed for any dynamical system is to use the least squares method for a quite large K. The main consequence is that the output λ will not be the exact Lyapunov exponent because it cannot describe the exponential expansion. Nevertheless, this λ can describe the whole movement of the system. If the system encounters a chaotic attractor, λ is positive even though it is lower than the true Lyapunov exponent. If the behavior is temporary chaotic then the Lyapunov exponent would exhibit a largely positive value whereas the current one would be considerably smaller or even negative, which is very close to the reality. As an example, the computation of Lyapunov exponent using both methods on the logistic map with $a = 3.56$ in Fig.3.13 results in:

- Lyapunov exponent: $\lambda = 0.0022$,
- alternative Lyapunov exponent: $\lambda = -0.0694$

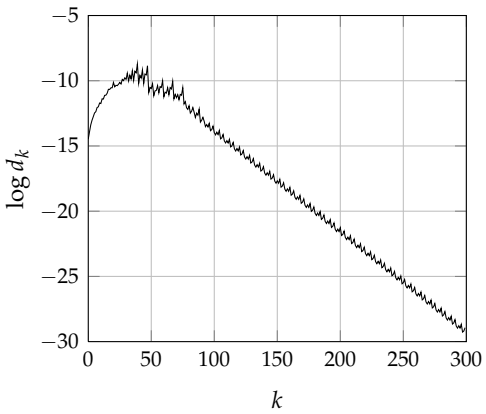


Figure 3.13: *Fake chaos,* $K = 300$

On Fig.3.14 is displayed the alternative exponent along the values of the parameter a in comparison with the Lyapunov exponent given by (3.16). Given that the whole evolution of d_k is used, not only the transient, therefore the least squares approximation results in least values. That is why around $a = 2$ the exponent exhibits less negative values than the Lyapunov exponent ones. Furthermore the fact it does not deal with the transient prevents it from being deceiving as we explained previously. Even though its shape is quite similar to the Lyapunov exponent one, it provides a more faithful explanation of the behavior of the logistic map. We do not display any curves about other systems as the Lorenz one. Nevertheless extensive experiments confirmed that the use of our alternative exponent to describe not only the transient but the whole evolution is relevant.

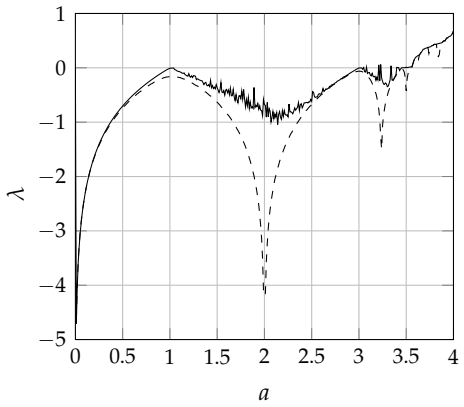


Figure 3.14: *Solid line is the alternative Lyapunov exponent, dashed line is the Lyapunov exponent from the differential method*

§ 3.5 State space reconstruction

The tools of the chaos theory that we have just presented provide information about the behavior of the studied system. The visualization of the associated trajectory in its own state space allows either to confirm the information or to improve it, *e.g.*, the famous Hénon map and the Lorenz map represented in their state space in Fig.3.16(a) and Fig.3.15(a).

However, multidimensional systems cannot be so easily represented, given that our human skills prevent us from observing more than three dimensions (or four if we consider time as a dimension). Furthermore, multidimensional dynamical systems are described by so many variables that it appears often very hard to acquire them all. Even in reasonable state space, as the Hénon map one, a few state variables are unavailable. To keep a reliable representation of a system, we need an alternative state space built by the State Space Reconstruction (SSR) [Kug96]. The SSR aims to put up an approximate state space by the only use of a subset of the state variables. Most time, the evolution of a single variable is enough to recover a faithful alternative trajectory, or more generally a one dimensional map of the state variables. Practically, we get back a time series $\{E(1), \ldots, E(K)\}$ of K points that we will map by the SSR to a matrix such that each column is a state variable of the alternative state space:

$$T = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \\ \vdots \\ \mathbf{T}_L \end{bmatrix} = \begin{bmatrix} E(1) & E(1+\tau) & \ldots & E(1+(m-1)\tau) \\ E(2) & E(2+\tau) & \ldots & E(2+(m-1)\tau) \\ \vdots & \vdots & & \vdots \\ E(n) & E(n+\tau) & \ldots & E(n+(m-1)\tau) \\ \vdots & \vdots & & \vdots \\ E(L) & E(L+\tau) & \ldots & E(L+(m-1)\tau) \end{bmatrix}$$

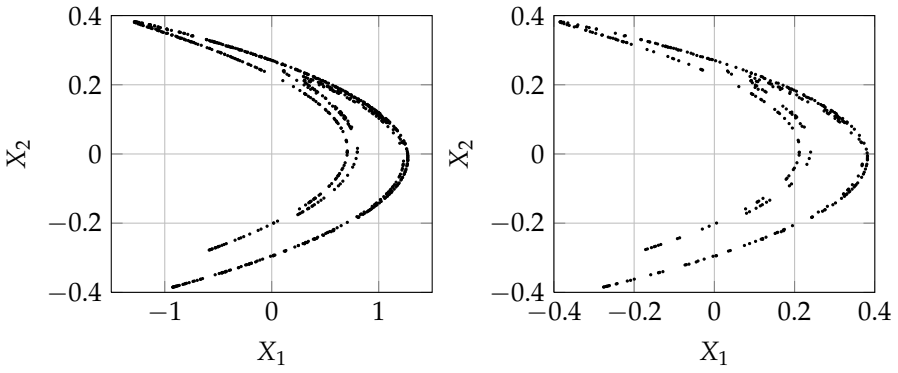
The quantities $[\mathbf{T}_1 \dots \mathbf{T}_L]$ are L points of the alternative trajectory inside the new state space that represents the system evolution. The value m is the target dimension of the approximate space, called the *embedding dimension* [H. 06]. The term τ is a delay and L is chosen such that:

$$L + (m-1)\tau \leq K \tag{3.17}$$

The SSR is a relevant tool as we are about to see considering the following situations:

- Hénon map: we use only one of the two state variables as the basic sequence to construct the new state space,

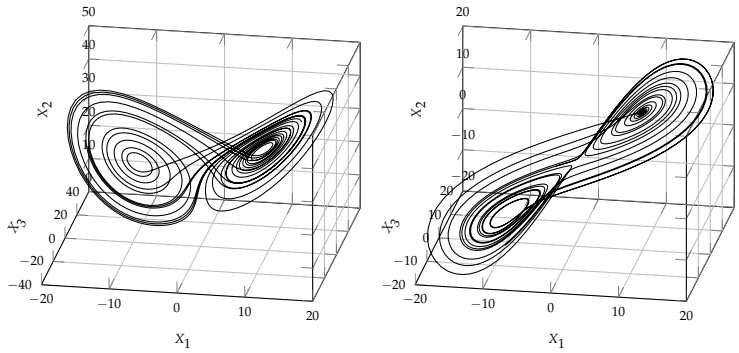
- Lorenz map: we use only one of the three state variables as the basic sequence to construct the new state space.



(a) Hénon map: 2D (b) Hénon map reconstruction

Figure 3.15: *Hénon map state space*

We display the result in Fig.3.15(b) and Fig.3.16(b), with $\tau = 1$, $m = 2$ for the Hénon map, and $\tau = 5$, $m = 3$ for the Lorenz map. These parameters are arbitrarily chosen given the embedding dimension value $m = 3$. We carried out numerous experiments to find the better suited τ, *i.e.* the delays that provide the most faithful trajectory compared with the original one. The shape of the Hénon map is perfectly restored although the x-axis suffers from a contraction comparing with Fig.3.15(a). Concerning the Lorenz map, the reconstruction is not perfect. However, the global shape is recovered, which is the goal of the SSR. Indeed, the alternative trajectory provides more qualitative descriptions of our systems than quantitative descriptions. Finally, it appears that the SSR is a good candidate for the representation of a dynamical system. Therefore the SSR will be a useful tool when we will expose the behaviors of the decoding algorithms that are multidimensional systems.



(a) Lorenz: 3D (b) Lorenz map reconstruction

Figure 3.16: *Lorenz map state space*

§ 3.6 Conclusion

We now have the main tools to describe the dynamics of the message-passing algorithms. One of the main task will be to define the state space, that basically conditions the quantifiers to use. Afterwards, it will be useful to redescribe their computation given that the decoders are less simple than the examples we made use of in this chapter. The goal will be to find the parameters that are responsible for non trivial behaviors of the algorithms, and to evaluate the nature of the associated attractors. We will also introduce other quantifiers dedicated to the decoder themselves.

Conclusion

We now have all the necessary tools to tackle the core of the problem, that is the explanation and the analysis of the Belief Propagation algorithm and its generalization, the Generalized Belief Propagation. The statistical physics and the optimization will help together to construct the message-passing equations of both algorithms. By the use of the probability theory on graphical models that we will also expose, we are able to make merge the different research areas in a single problem.

The chapter about the dynamical systems will be utilized in the last chapter, where we will try to confront the decoding algorithms on their stability and their behaviors according to the SNR of the AWGNC. Thanks to the presentation that we made, it will be possible to construct a relevant and consistent analysis of the dynamics of the Belief Propagation and the Generalized Belief Propagation.

Part II

Graphical models and Belief Propagation algorithms

Introduction

In this second part of the manuscript, we explain in details foundations of the Belief Propagation (BP) and the Generalized Belief Propagation (GBP) algorithms. The previous part helps get accustomed with the different theories needed to tackle this issue. In particular, statistical physics plays a major role in understanding the whole development that leads to the very equations of the decoding algorithms. This is essentially due to the root of this work and many others, that is to say Boltzmann's law. This equation allows us to roll out the whole procedure that concludes on the formulation of the inference problem as a constrained optimization of the free energy.

The BP and the GBP are constructed according to the stationary points of specific formulations of the free energy. As such, they are comparable algorithms, that is why it appears natural to confront them, even though the BP is much more tackled than the GBP in the literature. The relevancy and the originality of our investigation remains in the particular construction of the region-based approximation that we propose to obtain an efficient GBP algorithm. As a matter of fact, we study the topology of a given Tanner graph, that is not usual in the related papers, and we make use of the trapping sets to construct a powerful mean-field generalization, given that they represent one of the most challenging problems in channel coding. In this way, we obtain a relevant GBP algorithm especially for situations where the trapping sets are particularly harmful. We then propose a method to examine and confront the BP and the GBP in these situations.

We organize this part such that the first chapter is a deep investigation on the BP algorithm and its deep connections with the Bethe approximation in statistical physics. We also expose by the use of examples the suboptimality that it is subjected to, due to the loop-like topology of Tanner graphs. In the second chapter, we explain the origin and the functioning of the GBP algorithm based on the region-based approximation. We introduce an original method to factorize the joint distribution that implicitly leads to a specific form of the free energy and then to dedicated message-passing rules. We compare BP and GBP for LDPC codes where trapping sets are responsible for the BER degradation. In the last chapter, we consider the BP and the GBP algorithms as dynamical systems to bring out their properties about their behaviors face to non trivial situations. To this end, we use tools presented in first part, *e.g.* bifurcation diagram and Lyapunov exponent. We also introduce new tools to connect the dynamical behavior to the decoding performance of

both decoders. Thus, we determine sizes and positions of attractors, according to their natures and specific SNR values, provided by the previous dynamical quantifiers.

- Chapter 4 -

Bethe approximation: an approach to the Belief Propagation

§ 4.1 Introduction

BP is a well-known algorithm first introduced by Pearl [Pea88] in 1988 to solve inference problem on Bayesian networks in the framework of artificial intelligence. It swiftly turned out that this method could be applied to many research fields with similar issues. As a matter of fact, problems modeled by factor graphs, Markov Random Fields (MRF), or any other probabilistic graphical models, can be solved or at least approximately solved by BP, *e.g.* neural networks [T. 06b], image processing and video processing [M. 02], spin glasses [M. 06], channel coding [D. 95]. For each of these examples, the goal is to extract either an approximate of the most likely state or approximates of marginals. As a very important example developed in this chapter, it was proved in the 90's that BP had deep links with statistical physics where problems were oriented towards a constrained optimization of the free energy.

In this chapter we show how error correcting codes are closely related to networks handled by BP. We give a few properties of LDPC codes, then we describe the rationale behind the Tanner graph. After that, we introduce a demonstration of the BP equations according to statistical physics. Finally, we present problems linked to BP that will open onto the next chapters.

§ 4.2 LDPC codes and factor graphs

4.2.1 Error correction capability of LDPC codes

In 1963, Gallager created the Low-Density Parity-Check codes during his PhD [R.G63]. Impractical to implement at this time, they were forgotten until 1996 when McKay and Neal re-discovered them [D. 95]. These codes are aimed at providing fast and relevant decoding algorithms for recovering the input codeword $\mathbf{x}$. Tanner graphs of these codes do not contain many edges, they are sparse, making any message-passing algorithm, such as BP, low-complexity

algorithms. To evaluate the error correction capability of a given code, we commonly use d_{min} the minimum distance between any two of its codewords. Coding theory stipulates that the Maximum Likelihood Decoder (MLD) is able to correct a number of errors that equals:

$$\epsilon = E\lfloor \frac{d_{min} - 1}{2} \rfloor \tag{4.1}$$

with $E\lfloor \cdot \rfloor$ the floor function. ϵ is close to an affine behavior in d_{min}. Therefore, as d_{min} is increased, the corresponding code is able to correct more and more errors. Coding theory is aimed at designing error correcting codes which d_{min} is as large as possible. For very structured codes, defined as codes which construction is easily mapped for different values of N, a constraint is that d_{min} has to evolve the most linearly possible with N, that offers very good codes. However, computing d_{min} is not a trivial task as it is closely related to the rank of the parity-check matrix whose length N can reach very large values. That is partly why LDPC codes design is still an open problem.

Decoders, such as BP, do not behave as MLD, *i.e.* nothing ensures that ϵ errors would be practically corrected. This theoretical aspect makes d_{min} a good indicator to describe LDPC codes performance, whereas the BER is a fair indicator of decoders performance. The work we present in the following does not refer to d_{min} as we focus on decoders constructions, but anyone has to keep in his mind that global performance of a transmission partly rely on the couple code+decoder.

4.2.2 Tanner graph

Tanner graph is not only a support for message-passing algorithms, it has a deep meaning in the field of graphical models. Previously we pointed out that any channel could be modeled as a random map on the input codeword $\mathbf{x}$. It means that we cannot obtain any deterministic information concerning $\mathbf{x}$ from the data $\mathbf{y}$ but only a probabilistic information. From the receiver point of view, bits $x_1, \ldots, x_N$ are mapped to a probabilistic set made of the random variables $\mathbf{X} = \{X_1, \ldots, X_N\}$. The only feasible process for the receiver when no knowledge of the LDPC code use to encode information is available, is to estimate the *a posteriori* probability $p(\mathbf{X} = \mathbf{x}, \mathbf{y})$ [1] according to Bayes rule, using the likelihoods, as we mentioned in the previous chapter concerning the naive estimation.

An LDPC code $\mathcal{C}$ transforms the probabilistic set $\mathbf{X}$ into a probabilistic network $(\mathbf{X}, \mathbf{C})$. Any parity-check equation c_a is then a correlation function between its argument bits $\mathbf{X}_a$. In other words, any conditional probability distribution $p(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_N)$ is reduced to:

$$p(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_N) = p(X_i|\mathbf{X}_{\mathcal{N}_i}), \tag{4.2}$$

[1] *a posteriori* probability is originally defined as $p(\mathbf{x}|\mathbf{y})$. As channels are assumed to be memoryless, likelihoods do not depend on the LDPC code used to encode information, *i.e.* they only help get an insight on possible values of bits $x_1, \ldots, x_N$. Bayes rule stipulates that $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y})$, therefore provided observations $y_1, \ldots, y_N$, searching for distribution $p(\mathbf{x}, \mathbf{y})$ is equivalent to searching for distribution $p(\mathbf{x}|\mathbf{y})$. That is why we allow us to call *a posteriori* distribution both distributions.

where $\mathbf{X}_{\mathcal{N}_i} \subseteq \mathbf{X}$ is the set of variables that are linked to X_i by any parity-check equation (distinct from $\mathcal{N}_i$, the set of check nodes linked to X_i), see the example in Fig.4.1.

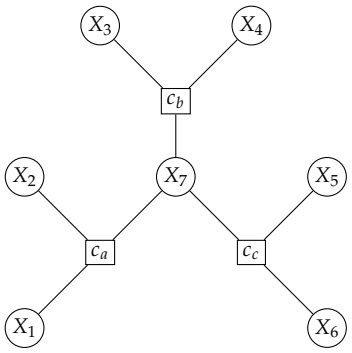


Figure 4.1: *Tanner graph* – $\mathbf{X}_{\mathcal{N}_7} = \{X_1, \dots, X_6\}$, $\mathcal{N}_7 = \{c_a, c_b, c_c\}$

Thus, the network is said to fulfill the *Markov property*. This makes the probabilistic set a Markov Random Field (MRF). We can represent this MRF by a bipartite graph $\mathcal{G} = (\mathbf{X} \cup \mathbf{C}, \mathbf{E})$ where the vertices are the variables $\mathbf{X} = \{X_1, \dots, X_N\}$ and the check nodes $\mathbf{C} = \{c_1, \dots, c_M\}$. Any edge $e = (c_a, X_i) = (X_i, c_a) \in \mathbf{E}$ is the dependence of the variables on the parity-check equations.

4.2.3 Factor graph

The Hammersley-Clifford theorem [P. 90], [J.M71] stipulates that the joint probability distribution of any MRF can be factorized over *cliques* of the graph $\mathcal{G}$, a clique being defined as a fully connected subgraph. In the current study, any check node c_a accompanied by its neighborhood $\mathbf{X}_a$ constitute a clique. Therefore we obtain the following form:

$$\boxed{p(\mathbf{x}) \triangleq p(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \prod_{a=1}^{M} f_a(\mathbf{x}_a)} \tag{4.3}$$

with Z the normalization constant that ensures that $p(\mathbf{x})$ is a genuine probability distribution, *i.e.* $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$. Functions $\{f_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ are symmetric functions called potentials, or kernels, depending on the context:
$\forall c_a \in \mathbf{C}, \forall X_{a_i}, X_{a_j} \in \mathbf{X}_a, \forall x_{a_i}, x_{a_j},$

$$f_a(x_{a_1}, \dots, x_{a_i}, \dots, x_{a_j}, \dots) = f_a(x_{a_1}, \dots, x_{a_j}, \dots, x_{a_i}, \dots) \tag{4.4}$$

Such a factorization (4.3) makes any MRF a *factor graph*. Generally, the factor graph distribution is a product of positive potential functions, but for LDPC codes, potentials are only valued in $\{0, 1\}$ *i.e.* the state of any clique $c_a \cup \mathbf{X}_a$ is strictly valid ($f_a(\mathbf{x}_a) = 1$) or not ($f_a(\mathbf{x}_a) = 0$). In other words, any potential function f_a is the XOR of the associated parity-check equations c_a:

$$\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad f_a(\mathbf{x}_a) = 1 \oplus \bigoplus_{X_i \in \mathbf{X}_a} x_i \tag{4.5}$$

The factor graph of any LDPC codes is called a *Tanner graph* [F.Ro1]. Equation (4.3) is the *a priori* distribution on the variables $X_1, \ldots, X_N$ from the receiver point of view. This expression is quite natural in the sense that:

- for any vector that is not a codeword, there is at least one parity-check equation that is not verified, *i.e.*,

$$\forall \mathbf{x} \in \mathbb{F}_2^N \backslash \mathcal{C}, \exists c_a, \quad f_a(\mathbf{x}_a) = 0 \quad \Longrightarrow \quad p(\mathbf{x}) = 0$$

- only the codewords have non-null probabilities to appear,

$$\forall \mathbf{x} \in \mathcal{C}, \forall c_a, \quad f_a(\mathbf{x}_a) \neq 0 \quad \Longrightarrow \quad p(\mathbf{x}) \neq 0$$

- the distribution over the codewords is uniform and can be computed knowing that $\mathcal{C}$ is built with 2^K codewords (K is the number of information bits encoded by the LDPC code),

$$\forall \mathbf{x} \in \mathcal{C}, \quad p(\mathbf{x}) = \frac{1}{2^K}.$$

According to Bayes rule, we can finally assert that the a posteriori probability distribution is:

$$\boxed{p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \prod_{a=1}^{M} f_a(\mathbf{x}_a) \prod_{i=1}^{N} p(y_i | x_i)} \tag{4.6}$$

From the receiver point of view, the estimate of the input sequence $\hat{\mathbf{x}}$ corresponds to:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}, \mathbf{y}) \tag{4.7}$$

Unfortunately, as it was pointed out about MLD, this distribution is not tractable given that we would need to span through 2^N states to reach $\hat{\mathbf{x}}$, and N is typically very large (*e.g.* $N = 64800$ in LDPC codes used for DVB-S2 standard [2]) making it not practical. To circumvent this problem of computation complexity, we use a trial and tractable probability distribution $\{b(\mathbf{x})\}_{\mathbf{x}}$, called the *joint belief*, that we want to be as close as possible to the true distribution $\{p(\mathbf{x})\}_{\mathbf{x}}$. In the following, we develop questions about the constraints it is subject to, and how to evaluate its accuracy.

§ 4.3 Pairwise Tanner graph

First of all, we study the case of pairwise Tanner graphs, because they form the simplest examples to understand and handle the factorization of the joint distribution. In such graphs, any edge implicitly carries a check node, because any check node only has two variable nodes as neighbors. We start with a tree, *i.e.* a loop-free graph, and we continue with a square lattice that is a graph which drawing forms a regular square tiling.

[2] see documentation on DVB-S2 at `http://www.etsi.org/deliver/etsi_en/302300_302399/302307/01.02.01_60/en_302307v010201p.pdf`

4.3.1 Tree case

We consider the tree represented on Fig.4.2.

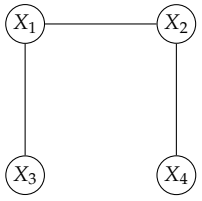


Figure 4.2: *Tanner tree*

Bayes rule and Markov property tell us that the joint distribution is:

$$p(\mathbf{x}) = p_{1|23}(x_1|x_2,x_3)p_{234}(x_2,x_3,x_4) \tag{4.8}$$

$$p(\mathbf{x}) = \frac{p_{123}(x_1,x_2,x_3)}{p_{23}(x_2,x_3)}p_{234}(x_2,x_3,x_4) \tag{4.9}$$

As $\{X_3, X_1, X_2\}$ is a chain, *i.e.* only one path connects these three nodes, we can easily factorize the term p_{123}:

$$p(\mathbf{x}) = \frac{p_{12}(x_1,x_2)p_{13}(x_1,x_3)p_{24}(x_2,x_4)}{p_1(x_1)p_2(x_2)} \tag{4.10}$$

Such a formulation is a product of the couples distributions that are p_{12}, p_{13}, p_{24} divided by the single variable intersections distributions that are p_1, p_2:

$$p(\mathbf{x}) = \frac{\prod\limits_{<X_i,X_j>} p_{ij}(x_i,x_j)}{\prod\limits_{X_i} p_i^{d_i-1}(x_i)} \tag{4.11}$$

where d_i is the number of edges incoming on the variable node X_i.

4.3.2 Loopy case

We consider the graph represented on Fig.4.3.

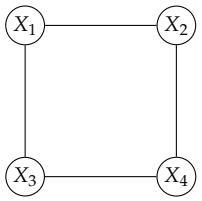


Figure 4.3: *Square Tanner lattice*

The joint distribution is the same as (4.9). However, equation (4.10) cannot be exact on this graph given that X_2 and X_3 are independent conditionally to X_1 and X_4 whereas in the tree case it was only conditioned to X_1.

In other words, if we can find more than one path between two nodes in the graph, *i.e.* if it contains loops, then equation (4.11) holds as an approximation. Most graphs are not trees, therefore factorization (4.11) remains an approximation in most cases. As a famous example in spin glasses theory or image processing, the square lattice of $N = 9$ variable nodes, represented in Fig.4.4, can be approximated with the formula (4.11).

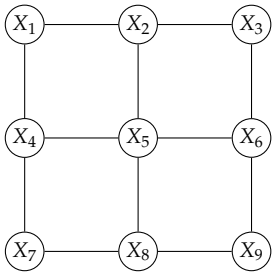
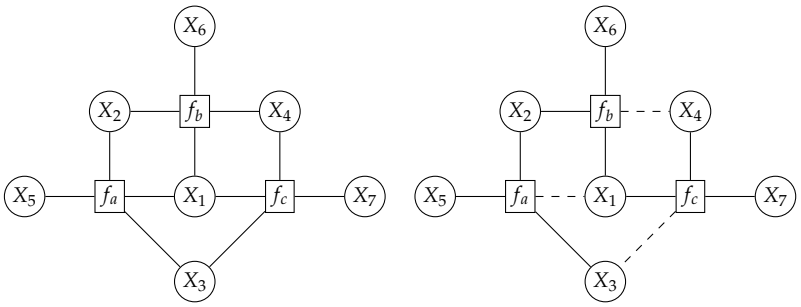


Figure 4.4: *Square lattice of length $N = 9$*

§ 4.4 General Tanner graph

We now consider the example of the Hamming code [R.W50] represented on Fig.4.5(a). We attempt to extend the factorization (4.11) to non-pairwise Tanner graphs.



(a) Tanner graph of the Hamming code

(b) Tanner tree of the modified Hamming code – Dashed edges are zeros in the parity-check matrix H

Figure 4.5: *Hamming code*

First, if we omit the edges $(f_a, X_1), (f_b, X_4), (f_c, X_3)$, see Fig.4.5(b), we obtain a loop-free Tanner graph whose joint distribution is exactly the extension of (4.11):

$$p(\mathbf{x}) = p_{5|123}(x_5|x_1, x_2, x_3)p_{6|124}(x_6|x_1, x_2, x_4)p_{7|134}(x_7|x_1, x_3, x_4) \quad (4.12)$$

$$p(\mathbf{x}) = \frac{p_a(\mathbf{x}_a)p_b(\mathbf{x}_b)p_c(\mathbf{x}_c)}{p_1^2(x_1)p_2(x_2)p_3(x_3)p_4(x_4)} \quad (4.13)$$

Unfortunately, factorization for the true Hamming code cannot be reduced to (4.13):

$$\begin{aligned} p(\mathbf{x}) &= p_{5|123}(x_5|x_1,x_2,x_3)p_{6|124}(x_6|x_1,x_2,x_4)p_{7|134}(x_7|x_1,x_3,x_4) && (4.14)\\ &= \frac{p_a(\mathbf{x}_a)p_b(\mathbf{x}_b)p_c(\mathbf{x}_c)p_{1234}(x_1,x_2,x_3,x_4)}{p_{123}(x_1,x_2,x_3)p_{124}(x_1,x_2,x_4)p_{134}(x_1,x_3,x_4)} && (4.15)\end{aligned}$$

It is due to the loop-like topology of the graph: several paths link any couple of variable nodes. Therefore the general rule of factorization is only an approximate of the joint distribution:

$$\boxed{p(\mathbf{x}) \approx \frac{\prod_{a=1}^{M} p_a(\mathbf{x}_a)}{\prod_{i=1}^{N} p_i^{d_i-1}(x_i)}} \tag{4.16}$$

and so does the belief:

$$b(\mathbf{x},\mathbf{y}) = \frac{\prod_{a=1}^{M} p_a(\mathbf{x}_a) \prod_{X_i \in \mathbf{X}_a} l_i(x_i,y_i)}{\prod_{i=1}^{N} \Big(p_i(x_i) l_i(x_i,y_i) \Big)^{d_i-1}} \tag{4.17}$$

It is a quite hard task to evaluate the influence of a given loop on the previous approximation, mainly because loops are very often interleaved altogether. In addition, we will see that the degradation of the approximation strongly depends on the transmission channel noise, therefore decoders, that we will describe in the following, are the only method to obtain a fair insight.

The factorized form is not the only necessary practical approximation. Indeed, in such a formula, we assume that marginal distributions of potentials $\{p_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ and single variable nodes $\{p_i(x_i)\}_{i,x_i}$ are available. We recall that for any potential f_a, for any state $\mathbf{x}_a$:

$$p_a(\mathbf{x}_a) = \sum_{\mathbf{x} \cup \mathbf{x}_a} p(\mathbf{x}) \tag{4.18}$$

and that for any variable node X_i, for any state x_i:

$$p_i(x_i) = \sum_{\mathbf{x} \cup x_i} p(\mathbf{x}) \tag{4.19}$$

These quantities cannot be practically obtained by these marginalizations as they depend on the joint distribution $p(\mathbf{x})$ we are trying to approximate. Therefore, marginal distributions have to be estimated by trial distributions, the *beliefs*: to any potential f_a is associated a belief $\{b_a(\mathbf{x}_a)\}_{\mathbf{x}_a}$ and to any single variable X_i is associated a belief $\{b_i(x_i)\}_i$.

§ 4.5 Energy

To estimate marginal distributions, we use the form of the joint distribution given by statistical physics. Previously we highlighted that any Tanner graph was a factor graph which probability distribution is:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{a=1}^{M} f_a(\mathbf{x}_a) \tag{4.20}$$

The Hammersley-Clifford theorem stipulates that any factor graph has the distribution of a Boltzmann's field [L. 64]:

$$p(\mathbf{x}) = \frac{1}{Z} e^{-E_I(\mathbf{x})} \tag{4.21}$$

where $E_I(\mathbf{x})$ is an energy function. According to statistical physics, $E_I(\mathbf{x})$ provides links or interactions between variable nodes of the network $\mathbf{X}$ in the state $\mathbf{x}$, *e.g.* the magnetic couplings $\{J_{ij}\}_{i,j}$ between the spins $\{S_i, S_j\}_{i,j}$ of a spin glass $\mathbf{S}$. The function E_I is linearly decomposed as:

$$E_I(\mathbf{x}) = \sum_{a=1}^{M} E_a(\mathbf{x}_a) \tag{4.22}$$

where each function E_a is the interaction energy between variable nodes of the subset $\mathbf{X}_a \subset \mathbf{X}$. This linear decomposition allows us to factorize the joint distribution as:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{a=1}^{M} e^{-E_a(\mathbf{x}_a)} \tag{4.23}$$

that provides, by a simple identification with (4.20), analytical links between potentials $\{f_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ and energy functions $\{E_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$:

$$\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad E_a(\mathbf{x}_a) = -\log f_a(\mathbf{x}_a) \tag{4.24}$$

This formulation carries a physical interpretation of LDPC codes. A configuration, or state, $\mathbf{x}$ such that at least one parity-check equation c_a is not verified, *i.e.* $f_a(\mathbf{x}_a) = 0$, is an infinite energy state. In other words, a forbidden state is made unreachable. In contrast, a state that verifies all parity-check equations is a ground state (state of minimum energy). Thus, to search for the most likely state of the distribution, *i.e.* $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x})$, is equivalent to search for the state that minimizes the energy function, $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E_I(\mathbf{x})$.

Any additive channel noise, *e.g.* the noise from the Additive White Gaussian Noise Channel (AWGNC), is equivalent to an external magnetic field $\mathbf{H}$ in which a spin glass is embedded in. As the AWGNC disturbs independently each bit of a codeword, $\mathbf{H}$ interacts independently with each spin. These magnetic interactions are modeled by another energy function:

$$E_H(\mathbf{x}) = \sum_{i=1}^{N} E_i(x_i) \tag{4.25}$$

which expression is a scalar product between the magnetic field and the spin vectors [3]. The total energy function is then:

$$E(\mathbf{x},\mathbf{H}) = E_I(\mathbf{x}) + E_H(\mathbf{x}) \quad (4.26)$$

$$= \sum_{a=1}^{M} E_a(\mathbf{x}_a) + \sum_{i=1}^{N} E_i(x_i) \quad (4.27)$$

Denoting any local interaction of a "spin" X_i with the magnetic field $\mathbf{H}$ by $E_i(x_i) = -\log l_i(x_i, y_i)$, where $y_i = x_i + n_i$ is the channel output and n_i is a noisy element, we obtain:

$$E(\mathbf{x},\mathbf{y}) = -\sum_{a=1}^{M} \log f_a(\mathbf{x}_a) - \sum_{i=1}^{N} \log l_i(x_i, y_i) \quad (4.28)$$

Then, joint distribution becomes:

$$p(\mathbf{x},\mathbf{y}) = \frac{1}{Z} \prod_{a=1}^{M} e^{-E_a(\mathbf{x}_a)} \prod_{i=1}^{N} e^{-E_i(x_i)} \quad (4.29)$$

or:

$$p(\mathbf{x},\mathbf{y}) = \frac{1}{Z} \prod_{a=1}^{M} f_a(\mathbf{x}_a) \prod_{i=1}^{N} l_i(x_i, y_i) \quad (4.30)$$

that equals (4.6). This helps demonstrate that an LDPC code is somehow equivalent to a spin glass, considering that:

- any coupled spins are bits of a common parity-check equation,
- any spin disturbed by an external magnetic field is a bit disturbed by a channel noise.

§ 4.6 Variational approach

Equations (4.17) and (4.28) allow us to construct a variational approach to solve the inference problem. The variational approach principle is to use a trial distribution, the belief $\{b(\mathbf{x},\mathbf{y})\}$, that we modify to fit the true joint distribution $\{p(\mathbf{x},\mathbf{y})\}$ as accurately as possible. Accuracy of any variational approach is evaluated according to a specific metric, as we will see in the following.

We do not take into account the *a posteriori* probability distribution, but the *a priori* one. Indeed, as the belief is a probability distribution, it satisfies the Bayes rule:

$$b(\mathbf{x},\mathbf{y}) = b(\mathbf{x})b(\mathbf{y}|\mathbf{x}) \quad (4.31)$$

As likelihoods $\{l_i(x_i, y_i) = p_i(y_i|x_i)\}_{i,x_i}$ are completely independent from factor graphs [4], the probability law of $\mathbf{Y}|\mathbf{X}$ does not depend on the inference method. As a matter of fact, it helps to discriminate valid states, or codewords, brought out by $\{b(\mathbf{x})\}_{\mathbf{x}}$, according to a given observation.

[3] denoting $\mathbf{H} = [h_1 \dots h_N]^T$ and $\mathbf{x} = [x_1 \dots x_N]^T$, we obtain that $E_H(\mathbf{x}) = -\sum_{i=1}^{N} h_i x_i$

[4] channels studied in this thesis (AWGNC, BSC) are memoryless: $p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{N} p_i(y_i|x_i)$

This way, we focus on the computation of the *a priori* probability distribution, see (4.16):

$$b(\mathbf{x}) = \frac{\prod_{a=1}^{M} b_a(\mathbf{x}_a)}{\prod_{i=1}^{N} b_i^{d_i-1}(x_i)} \tag{4.32}$$

where $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ and $\{b_i(x_i)\}_{i,x_i}$ are beliefs on potentials and variable nodes, respectively. To evaluate the relevancy of (4.32), a commonly used estimator is the Kullback-Liebler divergence [Kul68]:

$$KL(b||p) = \sum_{\mathbf{x}} b(\mathbf{x}) \log \frac{b(\mathbf{x})}{p(\mathbf{x})} \tag{4.33}$$

Reasons to select this estimator are that it offers a physical interpretation of the variational approach as explained in the following, and also that $KL(b||p)$ has a meaning in information theory: it is the relative entropy between distributions $b\{(\mathbf{x})\}_{\mathbf{x}}$ and $p\{(\mathbf{x})\}_{\mathbf{x}}$. This way, Kullback-Liebler divergence is a measure of the information lost when distribution $b\{(\mathbf{x})\}_{\mathbf{x}}$ is used to approximate distribution $p\{(\mathbf{x})\}_{\mathbf{x}}$. Main properties of this function are:

- $\forall\{p(\mathbf{x})\}_{\mathbf{x}}, \forall\{b(\mathbf{x})\}_{\mathbf{x}}, KL(b||p) \geq 0$,
- $KL(b||p) = 0$ if and only if $\forall\mathbf{x}, b(\mathbf{x}) = p(\mathbf{x})$,
- $KL(\cdot||p)$ and $KL(b||\cdot)$ are convex functions.

We will not detail the fact that it does not define a metric or a distance because we do not need this piece of information but the interested reader could refer to [Kul68] for more details.

A physical interpretation of the variational approach consists in developing $KL(b||p)$ considering $p(\mathbf{x})$ as a Boltzmann's field:

$$\begin{aligned} KL(b||p) &= \sum_{\mathbf{x}} b(\mathbf{x}) \log b(\mathbf{x}) - \sum_{\mathbf{x}} b(\mathbf{x}) \log \frac{e^{-E_I(\mathbf{x})}}{Z} && (4.34)\\ &= U_b - S_b + \log Z && (4.35) \end{aligned}$$

where:

- $U_b = \sum_{\mathbf{x}} b(\mathbf{x}) E_I(\mathbf{x})$ is the expectation value of the energy function called the *variational mean energy* provided that the network is described by the law $\{b(\mathbf{x})\}_{\mathbf{x}}$,
- $S_b = -\sum_{\mathbf{x}} b(\mathbf{x}) \log b(\mathbf{x})$ is the entropy relative to the distribution $\{b(\mathbf{x})\}_{\mathbf{x}}$, called the *variational entropy*.

Properties of the Kullback-Liebler divergence provide that:

$$\begin{aligned} U_b - S_b &\geq -\log Z && (4.36)\\ U_b - S_b &= -\log Z \quad \text{if and only if} \quad \forall\mathbf{x}, b(\mathbf{x}) = p(\mathbf{x}) && (4.37) \end{aligned}$$

We obtain a dual formulation of the inference problem: the variational approach consists in varying the joint belief to minimize the cost function

$$F_b \;=\; U_b \;-\; S_b \tag{4.38}$$

called the *variational free energy*. Therefore it is possible to reach analytic expressions of the marginal distributions replacing the trial distribution $\{b(\mathbf{x})\}_{\mathbf{x}}$ by its factorization (4.32). The expression of the variational entropy is:

$$S_b = -\sum_{a=1}^{M}\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\log b_a(\mathbf{x}_a) - \sum_{i=1}^{N}(1-d_i)\sum_{x_i} b_i(x_i)\log b_i(x_i) \tag{4.39}$$

The variational mean energy is computed thanks to the linear decomposition of the energy function (4.22):

$$\begin{aligned} U_b &= \sum_{a=1}^{M}\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)E_a(\mathbf{x}_a) && (4.40)\\ &= -\sum_{a=1}^{M}\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\log f_a(\mathbf{x}_a) && (4.41)\end{aligned}$$

On one hand, as shown in [J.So5], the variational mean energy is not an approximation of the mean energy $U_p = \sum_{\mathbf{x}} p(\mathbf{x})E(\mathbf{x})$, the only difference is due to the values of the beliefs [5]. On the other hand, even though all beliefs $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}, \{b_i(x_i)\}_{i,x_i}$ equal true marginals, the variational entropy is an approximation of the true entropy $S_p = -\sum_{\mathbf{x}} p(\mathbf{x})\log p(\mathbf{x})$ because the factorization of the joint distribution is only an approximation for most factor graphs (the loopy ones). We obtain the equation of the so-called variational free energy of the *Bethe approximation*:

$$\boxed{F_b = \sum_{a=1}^{M}\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\log\frac{b_a(\mathbf{x}_a)}{f_a(\mathbf{x}_a)} + \sum_{i=1}^{N}(1-d_i)b_i(x_i)\log b_i(x_i)} \tag{4.42}$$

§ 4.7 Optimization

To search for the most accurate belief is equivalent to optimize the cost function F_b *i.e.* to search for stationary points of the variational free energy. These points are distributions $\{b(\mathbf{x})\}_{\mathbf{x}}$ such that the gradient of F_b is zero *i.e.*:

$$\frac{\partial F_b}{\partial b(\mathbf{x})} = 0 \tag{4.43}$$

In the following we present constraints under which a function $\{b(\mathbf{x})\}_{\mathbf{x}}$ is acceptable to minimize F_b.

[5] demonstrations of (4.39) and (4.41) implicitly assume that any marginal distribution, either on a potential or on a variable node, is exact to the associated marginalization on the joint distribution, see next section for details

4.7.1 Constraints

Factorization of belief (4.32) is constituted of variational functions $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ and $\{b_i(x_i)\}_{i,x_i}$ assumed to be distributions, then they have to satisfy local marginalizations:

- (C1) $\forall c_a \in \mathbf{C}, \quad \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) = 1,$
- (C2) $\forall X_i \in \mathbf{X}, \quad \sum_{x_i} b_i(x_i) = 1.$

In addition, for any check node, distributions of neighboring variable nodes have to equal corresponding marginalizations:

- (C3) $\forall c_a \in \mathbf{C}, \forall X_i \in \mathbf{X}_a, \forall x_i, \quad b_i(x_i) = \sum_{\mathbf{x}_a \cup x_i} b_a(\mathbf{x}_a)$

Additional conditions ensure that these functions are non-negative:

- (C4) $\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad b_a(\mathbf{x}_a) \geq 0,$
- (C5) $\forall X_i \in \mathbf{X}, \forall x_i, \quad b_i(x_i) \geq 0.$

We obtain a constrained problem modeled, and sometimes solved, by the Lagrangian formalism [D.P01]. It is accordingly convenient to remove conditions that are linear combinations of other conditions, which helps lighten writings and computations:

- (C1) and (C3) imply constraint (C2) then (C2) is removable,
- (C3) and (C4) entail condition (C5) thus (C5) is voluntary omitted.

One would notice that the update equation of any belief $\{b_i(x_i)\}_{i,x_i}$ (1.14), exhibited in previous chapter, included a normalization constant, whereas condition (C1) guarantees such a property. As a matter of fact, all conditions are guaranteed to be satisfied for any optimum while any other points is not assumed to fulfill such conditions. The BP algorithm is an iterative process that could be interrupted at any iteration k either for time's sake or for any other reason. Thus $\{b_i^{(k)}(x_i)\}_{i,x_i}$, $\{b_a^{(k)}(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$ have to define acceptable solutions, even approximative.

4.7.1.1 Normalization

It is crucial to understand the normalization constraint because failing to do so could lead to a completely wrong optimization. We present here two formulations of the minimization problem that we encountered in the literature. In order to make the explanation quite simple, we reduce the current problem to the following minimization:

$$\begin{array}{rccc} f: & \mathbb{R}_+^n & \longrightarrow & \mathcal{Y} \subset \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) \end{array}$$

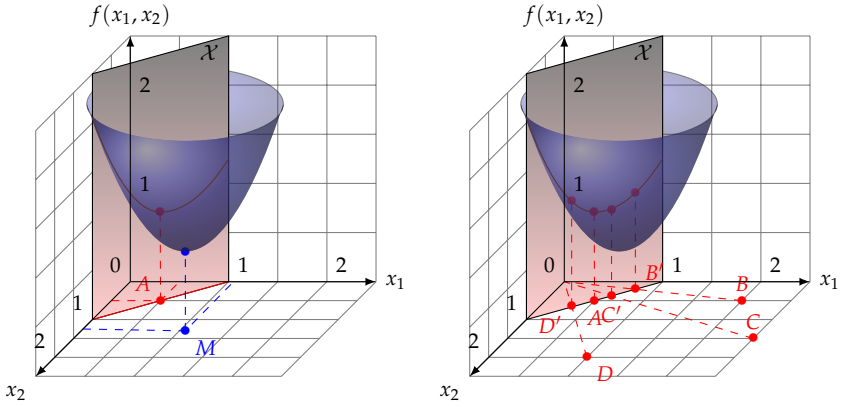
such that $\mathbf{x} \in \mathcal{X} = \{\mathbf{x} \in \mathbb{R}_+^n, ||\mathbf{x}|| = 1\}$, where $||\mathbf{x}|| = \sum_{i=1}^n x_i$. The first formulation of the optimization is:

$$(P1) \quad : \quad \left\{ \begin{array}{rcl} f(\mathbf{x}^*) & = & \min\limits_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \\ \mathbf{x}^* & = & \arg\min\limits_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \end{array} \right. \tag{4.44}$$

A modified formulation consists in relaxing the constraint by making it inherent to the function. We accordingly obtain the second formulation of the optimization:

$$(P2) \quad : \quad \begin{cases} f\left(\frac{\mathbf{x}^*}{||\mathbf{x}^*||}\right) &= \min\limits_{\mathbf{x}\in\mathbb{R}^n_+} f\left(\frac{\mathbf{x}}{||\mathbf{x}||}\right) \\ \mathbf{x}^* &= \arg\min\limits_{\mathbf{x}\in\mathbb{R}^n_+} f\left(\frac{\mathbf{x}}{||\mathbf{x}||}\right) \end{cases} \tag{4.45}$$

As we will now explain, (P1) and (P2) are not strictly equivalent. To this end, we consider the example on Fig.4.6 with $n = 2$, where function $f(x_1, x_2)$ is assumed to be a paraboloid.



(a) Constrained minimization (P1)

(b) Relaxed minimization (P2)

Figure 4.6: *Two formulations of the normalization constraint. (a) We only scan $\mathcal{X}$ to extract point A, the argument of the minimum. (b) Two points that live on the same line (B, B'), (C, C') or (D, D') will have the same mapping by $f\left(\frac{\cdot}{||\cdot||}\right)$ then the argument of the minimum if not a unique point but a line.*

Without any constraint, the minimum of f is reached in $M = [x_1(M), x_2(M)]$. The normalization constraint is the vertical plan $\mathcal{X} = \{(x_1, x_2) | x_1 + x_2 = 1\}$. Points of the paraboloid that match with the constraint constitute the parabola. Problem (P1) consists in spanning through $\mathcal{X}$ to find the couple (x_1^*, x_2^*) such that:

$$\forall (x_1, x_2) \in \mathcal{X} \backslash (x_1^*, x_2^*), \quad f(x_1, x_2) \geq f(x_1^*, x_2^*)$$

Thanks to the convexity of the parabola, it is guaranteed that we find the point $A = [x_1(A), x_2(A)]$, see Fig.4.6(a).

Considering problem (P2) Fig.4.6(b), searching for (x_1^*, x_2^*) is not restricted to $\mathcal{X}$. For each point $(x_1, x_2) \in \mathbb{R}^2_+$, we first compute its normalized version $(z_1 = \frac{x_1}{x_1+x_2}, z_2 = \frac{x_2}{x_1+x_2})$ that lives in $\mathcal{X}$, then we check if $f(z_1, z_2)$ is the minimum. Using this method, the minimum $f(A)$ is always found but the argument of this minimum is not reduced to a unique point. All points on the line that goes from the origin through $(x_1(A), x_2(A))$ are arguments of the

minimum because their normalized version equals point A, *i.e.*:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \subset \arg\min_{\mathbf{x}\in\mathbb{R}_+^2} f\left(\frac{\mathbf{x}}{||\mathbf{x}||}\right) \tag{4.46}$$

Finally, this toy example makes reveal that it is not a mistake not to consider the constraint on the normalization, given that the solution of the optimization is not unique in this case. The current problem of the minimization of the variational free energy F_b can then be formulated as searching for:

$$\arg \min_{\substack{\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}\in\mathbb{R}_+^{2^{d_a}} \\ \{b_i(x_i)\}_{i,x_i}\in\mathbb{R}_+^2}} F_b\left(\left\{\frac{b_a(\mathbf{x}_a)}{\sum_{\mathbf{x}_\alpha} b_a(\mathbf{x}_\alpha)}\right\}_{a,\mathbf{x}_a}, \left\{\frac{b_i(x_i)}{\sum_{x_j} b_i(x_j)}\right\}_{i,x_i}\right) \tag{4.47}$$

This way, enforcing normalization at every iteration does not prevent the BP from searching for and reaching the optimal solution.

Another formulation of the marginals normalizations that provides wrong result is the following one. First of all, we extract $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}, \{b_i(x_i)\}_{i,x_i}$, unnormalized functions that minimize F_b. As a post-treatment, we then normalize these functions. In the former example on the paraboloid, this method leads M to be projected on $\mathcal{X}$ as is exhibited in Fig.4.7.

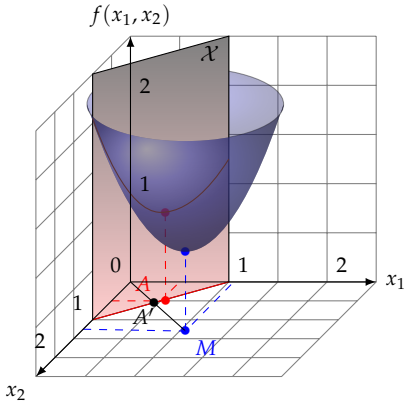


Figure 4.7: *Wrong formulation of the normalization constraint. The projection of M on $\mathcal{X}$ is different from A.*

We observe that point A' differs from point A, the result is wrong. This error is noticeable when inserting post-normalization quantities $\{Z_a\}_a, \{Z_i\}_i$ in the equation of the variational free energy. We obtain another function:

$$\begin{aligned}\tilde{F}_b \quad = \quad & \sum_{a=1}^{M} \frac{1}{Z_a} \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \log \frac{b_a(\mathbf{x}_a)}{f_a(\mathbf{x}_a)} - \sum_{a=1}^{M} \log Z_a \\ & + \sum_{i=1}^{N} \frac{1-d_i}{Z_i} b_i(x_i) \log b_i(x_i) - \sum_{i=1}^{N} (1-d_i) \log Z_i. \end{aligned} \tag{4.48}$$

Signs and values of the new terms strongly affect the variational free energy. Then, there is absolutely no reason that $\tilde{F_b} = F_b$, except if all post-normalization quantities equal 1, that is rarely true without any constraint on F_b. Finally, normalizing marginals cannot be done as a post-treatment.

4.7.1.2 Omitted constraints

We now point out constraints that are not mentioned in the literature, to our knowledge. First of all, the global marginalizations:

$$\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad b_a(\mathbf{x}_a) = \sum_{\mathbf{x} \cup \mathbf{x}_a} b(\mathbf{x}) \tag{4.49}$$

$$\forall X_i \in \mathbf{X}, \forall x_i, \quad b_i(x_i) = \sum_{\mathbf{x} \cup x_i} b(\mathbf{x}) \tag{4.50}$$

According to [T. 06a], the reason not to take into account the global marginalization is that the Bethe approximation does not consist in factorizing the joint belief as (4.17) but in decomposing the variational free energy according to (4.42) using *pseudo-marginals* on check nodes and variable nodes. Indeed, equation of F_b can be deduced according to the following principle: splitting variational mean energy and variational entropy over variable nodes and potentials. This principle is original whereas derivation of the joint distribution is somehow a consequence of this principle that was not originally presented in [H.A35]. The factorization helps introduce definitions to link with the inference problem. This is only an approximation and the values of $\{b(\mathbf{x})\}_{\mathbf{x}}$ are never used in the optimization process then it turns out not to be necessary to consider any joint belief in any inference algorithm.

The second constraint that is not mentioned is the global normalization:

$$\sum_{\mathbf{x}} b(\mathbf{x}) = 1 \tag{4.51}$$

As we mentioned previously, the joint belief is somehow a way to create the pseudo-marginals $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}, \{b_i(x_i)\}_{i,x_i}$. But it turns out possible to define these pseudo-marginals without any joint belief, by asserting that we approximate both the mean energy and the variational entropy by linear decompositions. Then any constraint on the joint belief is irrelevant because this quantity does not truly exist. Furthermore conditioning the minimization to a normalized joint belief does not carry any relevant information, it is equivalent to consider that its true form is:

$$B(\mathbf{x}) = \frac{1}{Z} b(\mathbf{x}) \tag{4.52}$$

$$B(\mathbf{x}) = \frac{1}{Z} \frac{\prod_{a=1}^{M} b_a(\mathbf{x}_a)}{\prod_{i=1}^{N} b_i^{d_i - 1}(x_i)} \tag{4.53}$$

where Z is the normalization such that $\sum_{\mathbf{x}} B(\mathbf{x}) = 1$. The variational free energy becomes:

$$F_B = \frac{1}{Z} F_b - \log Z \tag{4.54}$$

Thus, we observe that Z only implies an affine mapping of F_b to F_B, that does not affect the optimization problem. Finally we can assert that the use of Z is not necessary. Given that it also includes a non-negligible computation time, we do not take it into account in the optimization. Then we do not consider the optimization of F_B but of F_b.

4.7.2 Karush-Kuhn-Tucker conditions

Any problem with constraints of inequalities is generally solved using the Lagrangian formalism [D.P01]. To this end, it is necessary to check that all the constraints fulfill the Karush-Kuhn-Tucker conditions that are:

- continuous differentiability: any constraint and its differentiate are continuous,
- regularity: we can find a solution $\{b_a^*(\mathbf{x}_a)\}_{a,\mathbf{x}_a}$, $\{b_i^*(x_i)\}_{i,x_i}$ such that none of the differentiates of the constraints with these solutions is zero,
- we can find a set of non-infinite Lagrange multipliers $\{\gamma_a^*\}_a$, $\{\lambda_{ai}(x_i)\}_{a,i,x_i}$ such that:
  - primal feasibility: the constraints are all verified,
  - stationarity: the gradient of the Lagrange function is zero.

Given that constraints on the Bethe approximation satisfy KKT conditions, we define Lagrange multipliers $\{\gamma_a\}_a$, $\{\lambda_{ai}(x_i)\}_{a,i,x_i}$ for all constraints such that the Lagrange function is:

$$\begin{aligned}\mathcal{L} = F_b \quad &+ \quad \sum_{c_a \in \mathbf{C}} \gamma_a \left(\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) - 1 \right) \\ &+ \quad \sum_{c_a \in \mathbf{C}} \sum_{X_i \in \mathbf{X}_a} \sum_{x_i} \lambda_{ai}(x_i) \left(\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) - b_i(x_i) \right)\end{aligned} \tag{4.55}$$

KKT conditions are used to define analytic expressions of marginal beliefs. However, one should keep in mind that these KKT conditions are only necessary conditions. The equivalence is not trivial, especially because the variational free energy is rarely convex. Proving equivalence is a very hard task that is not tackled a lot in the literature. In [T. 06a] are introduced a few assumptions that lead to the equivalence but there is no general proof, *e.g.* it is demonstrated for Tanner graphs that only contain one loop, which leads to a perfect optimization in that case. However, many loops are encountered for most LDPC codes. To our knowledge, no proof ensures the equivalence in these cases, and it is very likely that the equivalence is not verified. Despite this drawback, we previously observed that BP has very decent performance, meaning that this omission is not significantly damageable, even though it is still an open problem.

4.7.3 Solution

We now search for positive marginal functions that minimize the Lagrange functions, therefore we compute the gradient of the Lagrange function ac-

cording to the marginal beliefs on check nodes and variable nodes:

$$\begin{aligned}\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad \frac{\partial \mathcal{L}}{\partial b_a(\mathbf{x}_a)} &= 1 + \log \frac{b_a(\mathbf{x}_a)}{f_a(\mathbf{x}_a)} + \gamma_a + \sum_{X_i \in \mathbf{X}_a} \lambda_{ai}(x_i) \\ \forall X_i \in \mathbf{X}, \forall x_i, \quad \frac{\partial \mathcal{L}}{\partial b_i(x_i)} &= (1-d_i)\Big(1 + \log b_i(x_i)\Big) - \sum_{c_a \in \mathcal{N}_i} \lambda_{ai}(x_i)\end{aligned}$$

Arguments that nullify the gradient are called *stationary points* of Bethe approximation. We obtain their equations [6]:

$$\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad b_a^*(\mathbf{x}_a) = f_a(\mathbf{x}_a) e^{-1-\gamma_a} \prod_{X_i \in \mathbf{X}_a} e^{-\lambda_{ai}(x_i)} \tag{4.56}$$

$$\forall X_i \in \mathbf{X}, \forall x_i, \quad b_i^*(x_i) = e^{-1} \prod_{c_a \in \mathcal{N}_i} e^{\frac{\lambda_{ai}(x_i)}{1-d_i}} \tag{4.57}$$

We observe that these functions are positive then it is not useful to consider constraints (C4) and (C5), that is why they do not appear in the equation of the Lagrange function (4.55). As for any variable node X_i, $b_i(x_i, y_i) = b_i(x_i) l_i(x_i, y_i)$ and for any check node c_a, $b_a(\mathbf{x}_a, \mathbf{y}_a) = b_a(\mathbf{x}_a) l_a(\mathbf{x}_a, \mathbf{y}_a)$ with $l_a(\mathbf{x}_a, \mathbf{y}_a) = \prod_{X_i \in \mathbf{X}_a} l_i(x_i, y_i)$ we obtain:

$$\forall c_a \in \mathbf{C}, \forall \mathbf{x}_a, \quad b_a^*(\mathbf{x}_a, \mathbf{y}_a) = l_a(\mathbf{x}_a, \mathbf{y}_a) f_a(\mathbf{x}_a) e^{-1-\gamma_a} \prod_{X_i \in \mathbf{X}_a} e^{-\lambda_{ai}(x_i)} \tag{4.58}$$

$$\forall X_i \in \mathbf{X}, \forall x_i, \quad b_i^*(x_i, y_i) = l_i(x_i, y_i) e^{-1} \prod_{c_a \in \mathcal{N}_i} e^{\frac{\lambda_{ai}(x_i)}{1-d_i}} \tag{4.59}$$

Beliefs are solutions to what is called *primal problem* [D.P01]. At this point, Lagrange multipliers are not valued, their assignments is the *dual problem*. Using constraints, it is possible to obtain a system of equations that does not depend on beliefs anymore but only on Lagrange multipliers, *e.g.* constraint (C3) is equivalent to:

$$e^{\frac{\lambda_{ai}(x_i)}{1-d_i}} = \frac{e^{-\gamma_a} \sum_{\mathbf{x}_a \cup x_i} f_a(\mathbf{x}_a) \prod_{X_j \in \mathbf{X}_a \setminus X_i} e^{-\lambda_{aj}(x_j)}}{e^{\lambda_{ai}(x_i)} \prod_{\mathbf{X}_b \ni X_i, f_b \neq f_a} e^{\frac{\lambda_{bi}(x_i)}{1-d_i}}} \tag{4.60}$$

However, the system is not linear and no method in the literature indicates how to handle equations to reach a solution. Partly tackled in [T. 06a] and [A.L03], computation of multipliers is very hard to handle but it emerged an alternative algorithm, the CCCP 1.6 to solve it. In spite of encouraging results [J.-13a], CCCP does not success for any LDPC code, especially due their loop-like topologies.

[6] any variable node X_i of a unit degree $d_i = 1$ does not appear in (4.42). Therefore, Lagrangian formalism cannot offer any equation of associated belief. Such a belief is obtained according to constraint (C3).

Another approach to this dual problem is offered by the BP algorithm. First of all, we assume that:

$$e^{-\lambda_{ai}(x_i)} = \prod_{\mathbf{X}_b \ni X_i, c_b \neq c_a} e^{\frac{\lambda_{bi}(x_i)}{1-d_i}} \tag{4.61}$$

Afterwards, we can reduce (4.60):

$$e^{\frac{\lambda_{ai}(x_i)}{1-d_i}} = e^{-\gamma_a} \sum_{\mathbf{x}_a \cup x_i} f_a(\mathbf{x}_a) \prod_{X_j \in \mathbf{X}_a \setminus X_i} e^{-\lambda_{aj}(x_j)} \tag{4.62}$$

These two equations are messages equations of the BP algorithm provided that messages and Lagrange multipliers are linked such that:

$$m_{ai}(x_i) = e^{\frac{\lambda_{ai}(x_i)}{1-d_i}} \tag{4.63}$$

$$n_{ia}(x_i) = e^{-\lambda_{ai}(x_i)} \tag{4.64}$$

Using an iterative process to solve these Lagrange multipliers is performing the message-passing of BP (1.12) and (1.13). We then obtain beliefs equations of BP:

$$b_a(\mathbf{x}_a, \mathbf{y}_a) = \frac{1}{Z_a} l_a(\mathbf{x}_a, \mathbf{y}_a) f_a(\mathbf{x}_a) \prod_{X_i \in \mathbf{X}_a} n_{ia}(x_i) \tag{4.65}$$

$$b_i(x_i, y_i) = \frac{1}{Z_i} l_i(x_i, y_i) \prod_{c_a \in \mathcal{N}_i} m_{ai}(x_i) \tag{4.66}$$

where Z_i, Z_a are normalization constants. Identifying messages to Lagrange multipliers provides the crucial result that stationary points of the variational free energy of Bethe are fixed points of BP. We summarize three equivalences that we have encountered until now:

- searching for the most likely state of Tanner graphs is searching for the ground state of energy functions,
- searching for the most accurate belief is minimizing the variational free energy,
- searching for BP fixed points is searching for stationary points of the variational free energy of Bethe.

§ 4.8 BP assumption

Many experiments have shown in the literature that BP is quite efficient to decode LDPC codes, and more generally to infer any factor graph. However, its equations are based on a particular assumption that prevents it from being optimal. To understand it well, we use two very simple examples in which we extract the belief on a given variable node.

4.8.1 Tree-like Tanner graph

A Tanner graph has a tree-like topology if and only if any two nodes are linked by a unique path, *e.g.* Tanner graph in Fig.4.8.

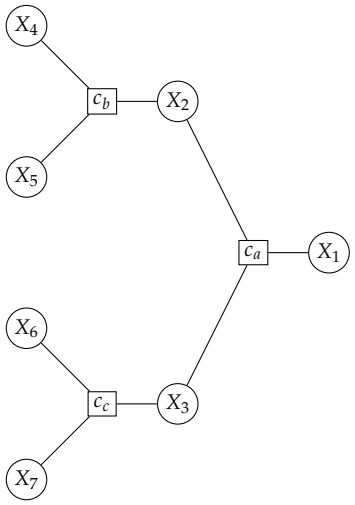


Figure 4.8: *Tree-like Tanner graph*

In this case, messages and beliefs of BP reach *canonical forms*, that is to say expressions that only depend on parity-check equations and likelihoods. In previous figure, after few iterations, the marginal belief of variable node X_1 becomes:

$$\begin{aligned} b_1^{(k)}(x_1, y_1) &= \frac{l_1(x_1, y_1)}{Z} \sum_{x_2,x_3,x_4,x_5,x_6,x_7} l_2(x_2, y_2) l_3(x_3, y_3) l_4(x_4, y_4) \\ &\quad \times l_5(x_5, y_5) l_6(x_6, y_6) l_7(x_7, y_7) \\ &\quad \times f_a(x_1, x_2, x_3) f_b(x_2, x_4, x_5) f_c(x_3, x_6, x_7) \end{aligned} \tag{4.67}$$

We observe that it does not depend on messages or initialization quantities due to the iterative process but only on intrinsic parameters of the problem. As a matter of fact, anyone can verify that for any other variable node X_i:

$$\forall x_i, \quad b_i(x_i, y_i) = \sum_{\mathbf{x} \cup x_i} \frac{1}{Z} \prod_{a=1}^{M} f_a(\mathbf{x}_a) \prod_{i=1}^{N} l_i(x_i, y_i) \tag{4.68}$$

which is the exact marginalization of the true joint distribution (4.6). It finally confirms that BP reaches optimal performance for tree-like LDPC codes.

4.8.2 Non tree-like Tanner graph

We use the previous Tanner graph considering variable nodes X_5 and X_6 merged, see Fig.4.9. Follow BP update equations, we obtain at any iteration k:

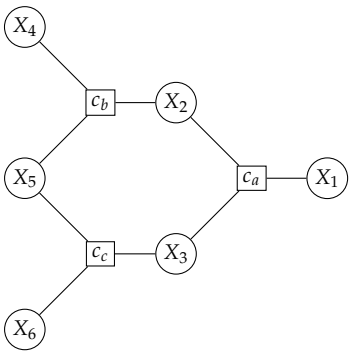


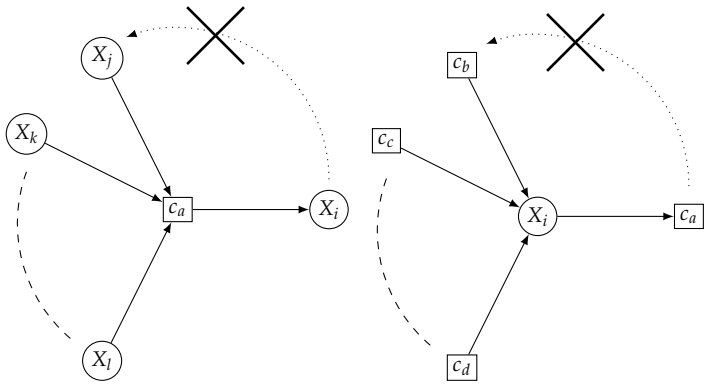
Figure 4.9: *Non tree-like Tanner graph*

$$\begin{aligned} m_{2a}^{(k)}(x_2,y_2) &= \frac{l_2(x_2,y_2)}{Z} \sum_{x_1,x_2,x_3,x_4,x_5,x_6} f_b(x_2,x_4,x_5) f_c(x_3,x_5,x_6) \\ &\quad \times l_3(x_3,y_3) l_4(x_4,y_4) l_5(x_5,y_5) l_6(x_6,y_6) \\ &\quad \times m_{1a}^{(k-3)}(x_1,y_1) m_{2a}^{(k-3)}(x_2,y_2) \end{aligned} \tag{4.69}$$

Message $m_{2a}^{(k)}$ depends on its value three iterations before $m_{2a}^{(k-3)}$, due to the loop $(c_a) - (X_2) - (c_b) - (X_5) - (c_c) - (X_3) - (c_a)$. It is then impossible to reach canonical equation (4.68), making BP a suboptimal algorithm.

4.8.3 Loops

We show in Fig.4.10 local Tanner graphs used to compute BP equations (1.12) and (1.13).



(a) Message from c_a to X_i (b) Message from X_i to c_a

Figure 4.10: *Local Tanner graph for messages computations*

Crosses indicate that BP assumes that there is a unique path between two nodes of the Tanner graph, we call this the *BP assumption*. Therefore, this assumption makes us temporarily ignore few edges when computing messages. Anyone can show that this prevents us from obtaining canonical equations of

messages and then beliefs, as we previously saw (4.69). Thus, tree-like graphs result in optimal performances, contrary to loopy graphs.

In addition, iterative process involves that errors due to loops are spread out in the graph depending on loops sizes: the shorter a loop is, the more harmful it is for the approximation as its effect appears in a number of iterations equal to the number of variable nodes it contains.

Designing LDPC codes of a high minimum distance needs sophisticated links between variable nodes to make them robust against channel noise. Regularly, robustness is relevant when any bit takes part in more than two parity-check equations. The other side of the coin is that it creates loops in Tanner graphs. Challenge of coding theory is then to design codes with a high minimum distance and with loops as large as possible. Optimal solution then appear unreachable by the BP algorithm.

4.8.4 Physical interpretation

Considering the equivalence with statistical physics, such a result is not surprising. Indeed, Bethe approximation is a MF approach, see in first chapters 2.4, which is obvious when confronting both fundamental equations:

$$(MF) \ : \ b(\mathbf{x}) = \prod_{i=1}^{N} b_i(x_i), \quad (Bethe) \ : \ b(\mathbf{x}) = \frac{\prod_{a=1}^{M} b_a(\mathbf{x}_a)}{\prod_{i=1}^{N} b_i^{d_i-1}(x_i)} \tag{4.70}$$

It means that instead of ignoring couplings when computing marginal distributions on single variable nodes, as originally done in MF equations, Bethe approximation is aimed at neglecting longer range correlations between spins, that are loops. In other words, Bethe approximation extends MF approach. First of all, we will see in next chapter that this note considerably help improving again MF approaches. In addition, this also helps not to consider the BP assumption as a theoretical mistake but as a limit that goes further than the MF one.

4.8.5 Oscillation

An effect due to the loop-like topology of Tanner graphs is the oscillation of messages, and then of the beliefs. A simple example helps understand well this phenomenon. We consider the Tanner graph in Fig.4.11.

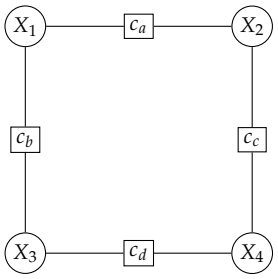


Figure 4.11: *Tanner graph of a loop*

The only acceptable states are $\mathbf{x} \in \{\mathbf{0}, \mathbf{1}\}$. We now assume that likelihoods are:

$$l_1(x_1, y_1) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}, \quad l_2(x_2, y_2) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix},$$
$$l_3(x_3, y_3) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix}, \quad l_4(x_4, y_4) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}.$$

In this case, we can show, according to BP equations, that beliefs oscillate such that for any iteration k:

$$b_1^{(k)}(x_1, y_1) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix}, \quad b_1^{(k+1)}(x_1, y_1) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix},$$
$$b_2^{(k)}(x_2, y_2) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}, \quad b_2^{(k+1)}(x_2, y_2) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix},$$
$$b_3^{(k)}(x_3, y_3) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}, \quad b_3^{(k+1)}(x_3, y_3) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix},$$
$$b_4^{(k)}(x_4, y_4) = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix}, \quad b_4^{(k+1)}(x_4, y_4) = \begin{bmatrix} 0.75 \\ 0.25 \end{bmatrix}.$$

The estimate of the input codeword then also oscillates:

$$\mathbf{x}^{(k)} = [1001], \quad \mathbf{x}^{(k+1)} = [0110]$$

Both states 1001 and 0110 are unacceptable because they do not satisfy all parity-check equations. More precisely, for a given variable node, neighboring check nodes carry conflicting information about its most likely state. It is then impossible to simultaneously satisfy all parity-check equations. Such a behavior prevents BP from reaching any steady state, which clearly damages its performance. As we saw in chapter 2.3, oscillations also appear in statistical physics for spin glasses and are referred to as frustration.

4.8.6 Trapping-sets

The BP assumption can seriously damage loopy LDPC codes decoding. Worse than loops are combinations of loops, also known as *trapping sets* [T.J03]. A trapping set is a nonempty set of variable nodes in a Tanner graph that are not eventually corrected by the BP decoder [7]. An (a, b) trapping set, shortly written TS(a, b), is a set of a variable nodes such that the induced subgraph has b odd-degree, or unverified, check nodes. A trapping sets ontology has been created for LDPC codes which variable nodes have three checks neighbors, and which shortest loops contain eight nodes [B. 09]. In this work, we focus on TS$(5, 3)$ represented in Fig.4.12.
Trapping sets are especially investigated on discrete channels, particularly BSC, because in these cases, they match low-weight errors events, *i.e.* sequences of erroneous bits that BP cannot eventually decode. They are responsible for the *error floor* [T.J03], an abrupt degradation of the error-rate slope that occurs for low crossover probabilities of BSC, see Fig.4.13. Researchers deeply investigate trapping sets and their relations to failure of iterative decoders.

[7] strictly speaking, the trapping sets definition does not depend on BP. As a matter of fact, these are not eventually corrected by any arbitrary iterative decoder, as Gallager-B, Bit-flipping,

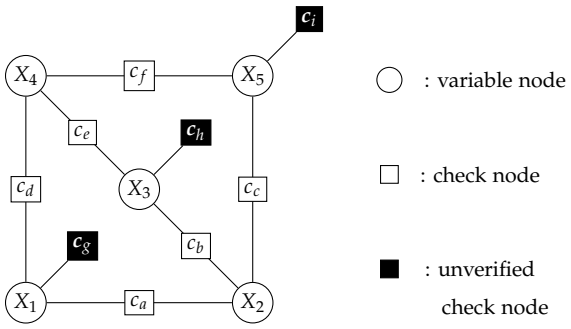


Figure 4.12: *Trapping-set of $a = 5$ variable nodes and $b = 3$ unverified check nodes*

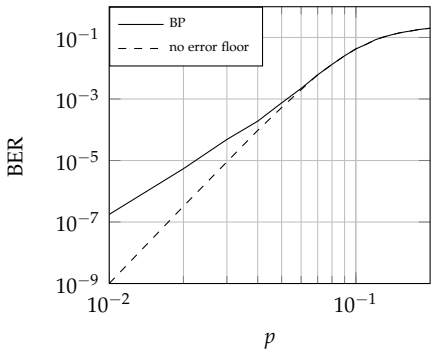


Figure 4.13: *BP error floor when decoding the Tanner code covered by TS(5, 3).*

Two points of view remain to circumvent the error-floor:

- either we construct LDPC codes that contain neither small loops nor combinations of them [D.V12],
- or we design decoders able to manage trapping sets.

First point of view has already been mentioned when we introduced loops. In the second one, Finite Alphabet Iterative Decoders (FAID), strongly studied in [S.K11] and [D. 12], can significantly improve error-rate for very low values of the crossover probability. Two drawbacks of these algorithms in our study are that they are not defined on LDPC codes with arbitrary node degrees and they do not deal with any continuous channels, as the AWGNC. In our study, we choose not to discriminate between any of these channels, therefore we focus on a generalization of BP, described in next chapter, studied on BSC and AWGNC. This decoder would help improve the BP decoding performance, annihilating loops and often trapping sets effects.

BP, etc. In this thesis, we focus on failures of BP, that is why we orient definition towards BP

§ 4.9 Conclusion

BP and Bethe approximation are demonstrated to be equivalent, therefore they are two different points of view on a common problem. This result help interpret any Tanner graph as a physical network, *e.g.* spin glasses, which can be studied according to a variational approach. Indeed, minimizing the variational free energy of Bethe using trial distributions is searching for marginal beliefs as close as possible to exact marginals. We saw that BP – to some extent Bethe approximation too – was based on a crucial assumption telling that loops must be ignored to get messages update equations. According to statistical physics, this is somehow an extension of the MF approach that even ignores couplings between spins (only when factorizing joint distribution). Literature on coding theory indicates that most LDPC codes are full of loops and trapping sets which then prevent BP from being an optimal decoder. Thus, searching for a more general MF approach turns out to be necessary to overcome this very important problem.

- Chapter 5 -

Region-based approximation: beyond the Belief Propagation

§ 5.1 Introduction

In this chapter, we present an extension of the BP algorithm, the Generalized Belief Propagation (GBP) algorithm. Stemming from the cluster variation method that gave birth to the Kikuchi approximation, also known as region-based approximation, in statistical physics, this method provides a more general approach to factorize the joint distribution. This is actually the general form that the Bethe approximation and the Mean-Field are particular cases of. It helps deal with the loops problem inside Tanner graphs of LDPC code extending nodes to the concept of regions, where regions are defined as subgraphs, or gathering of nodes. This allows us to build a mapping of any Tanner graph to a region-graph where harmful loop-like structures are absorbed in regions.

We begin this chapter introducing the region-based formalism for any factor graph. The region-based approximation is then tackled from the point of view of the inference and statistical physics, that makes emerge a new variational free energy called the region-based free energy. This stands for a new cost function which stationary points provide equations of a new message-passing, the GBP algorithm. We also describe general damping rules that help the decoder converge, contrary to its original version. Afterward, we present our original rules of the region-graph construction based on trapping sets. In addition, we exhibit results that reflect significant performance improvements compared with the BP, according to an original method that emphasizes on error-events based on trapping sets.

A part of the following results are published in [J.-12c] and [J.-12a].

§ 5.2 The region-based formalism

Bethe approximation, presented in previous chapter, mostly relies on an approximate factorization of the joint distribution:

$$p(\mathbf{x}) \approx \frac{\prod_{a=1}^{M} p_a(\mathbf{x}_a)}{\prod_{i=1}^{N} p_i^{d_i-1}(x_i)} \tag{5.1}$$

which is exact if and only if the corresponding factor graph is a tree. We recall that this formulation stems from Bayes rule and Markov condition [1]. As a matter of fact, Markov condition does not restrict the factorization of the joint distribution to be only (5.1). More precisely, we are allowed to extend Markov condition asserting that the distribution of any set of variable nodes only depends on the neighborhoods of all these nodes. This way, we find other factorizations that can be more accurate or even exact. To illustrate such tellings, we consider the factor graph in Fig.5.1.

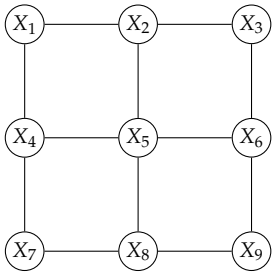


Figure 5.1: *Square lattice of length $N = 9$*

Bayes rule allows us to formulate the joint distribution as:

$$p(\mathbf{x}) = \frac{p_{124578}(x_1, x_2, x_4, x_5, x_7, x_8)\, p_{235689}(x_2, x_3, x_5, x_6, x_8, x_9)}{p_{258}(x_2, x_5, x_8)} \tag{5.2}$$

Considering BP assumption[2], the Markov condition offers us that:

$$p(\mathbf{x}) \approx \frac{p_{1245}(x_1, x_2, x_4, x_5)\, p_{2356}(x_2, x_3, x_5, x_6)\, p_{4578}(x_4, x_5, x_7, x_8)\, p_{5689}(x_5, x_6, x_8, x_9)}{p_{25}(x_2, x_5)\, p_{45}(x_4, x_5)\, p_{56}(x_5, x_6)\, p_{58}(x_5, x_8)}\, p_5(x_5) \tag{5.3}$$

Both factorizations, either exact or approximative, are also distributions of other graphical representations, see Fig.5.2 and Fig.5.3. For these factorizations, marginal distributions are computed over subgraphs of the factor graph. These subgraphs are called *regions*, a region r being made of:

- a set of variable nodes $\mathbf{X}_r \subseteq \mathbf{X}$,
- a set of check nodes $\mathbf{C}_r \subseteq \mathbf{C}$[3] that can be an empty set,

[1] that stipulates that the distribution of any variable node only depends on its neighborhood

[2] any two nodes are only linked by a unique path

[3] provided that any check node c_a of $\mathbf{C}_r$ is accompanied by its neighborhood $\mathbf{X}_a \subseteq \mathbf{X}_r$ because a region has to be a subgraph (associated edges are implicitly included in r)

The regions set used to write down the factorization is denoted by $\mathcal{R}$. Joint distribution of a factor graph is then approximated by:

$$\boxed{p(\mathbf{x}) \approx \prod_{r \in \mathcal{R}} p_r^{c_r}(\mathbf{x}_r)} \tag{5.4}$$

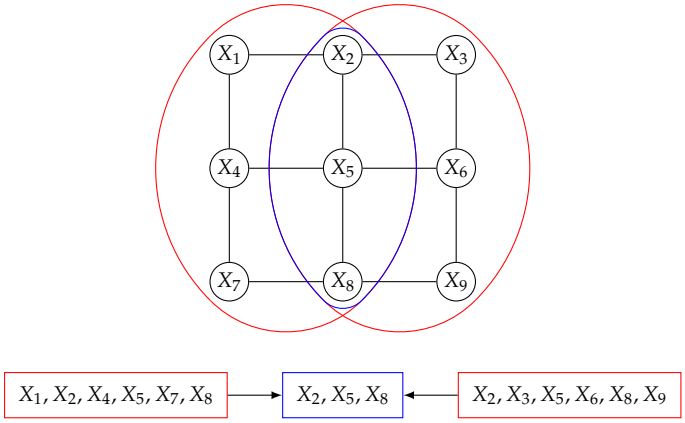


Figure 5.2: *A Bayesian network as a graphical representation of* (5.2)

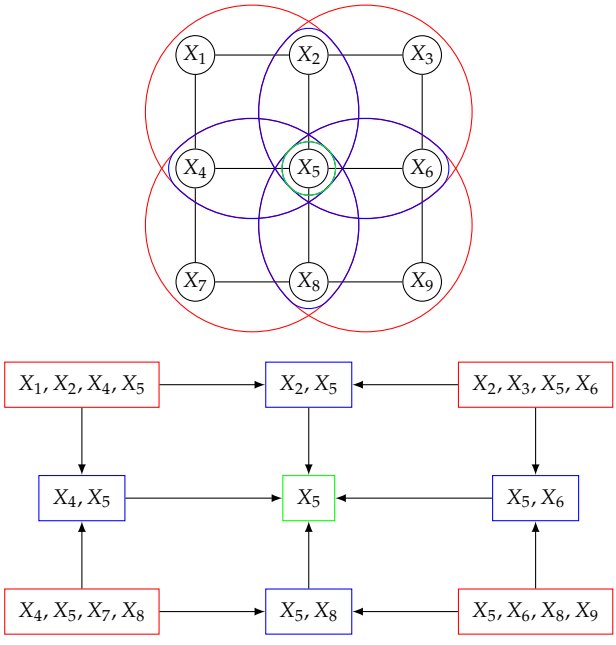


Figure 5.3: *Another Bayesian network as a graphical representation of* (5.3)

where c_r denotes what is called the *counting number* of r, that is the weight of its distribution inside the joint distribution, *e.g.* in (5.3) region $\{X_1, X_2, X_4, X_5\}$

has a unit weight, and region $\{X_2, X_5\}$ has a weight of -1. This formulation is a *region-based* formulation of the distribution.

5.2.1 Links with MF approach

At this point appears a deep link with the mean-field approach we exposed in chapter 2.4. We recall that MF consists in approximating the joint distribution by the following factorization:

$$p_{MF}(\mathbf{x}) = \prod_{i=1}^{N} p_i(x_i) \tag{5.5}$$

This corresponds to (5.4) with any region being a single variable node with a unit weight. The rationale behind the MF is that any single variable node considers all the other variable nodes as a unique global field or node. The Bethe approximation, linked to the factorization (5.1), is a first extension of the MF. In this case, cliques of variable nodes consider the other cliques, and to some extent the other variable nodes, as a unique global node. In this way, the joint belief could approach the exact form of the joint distribution, and therefore it could be more accurate than the MF. As an example, the Bethe approximation is exact if the factor graph is a tree whereas the MF cannot be. In the current case, we are given more freedom to write the factorization. It allows us to deal with bigger subsets of variable nodes, and to some extent to deal with harmful topological structures *e.g.* loops and trapping sets, as we will see in the following.

5.2.2 Counting numbers

In Bethe approximation, factorization of the joint distribution is:

$$b(\mathbf{x}) = \prod_{a=1}^{M} b_a(\mathbf{x}_a) \prod_{i=1}^{N} b_i^{1-d_i}(x_i) \tag{5.6}$$

Comparing with the region-based factorization, we observe that any potential has a unit weight, and any variable node X_i has a weight of $1 - d_i$. These weights directly stem from Bayes rule and Markov condition, such that any single variable node – and to some extent any potential – has to contribute only once to the factorization. According to the Bethe approximation, considering an arbitrary variable node X_i, we observe that:

- X_i appears in d_i marginal functions on potentials,
- X_i appears once in marginal $\{b_i(x_i)\}_{x_i}$ of multiplicity $1 - d_i$.

As $d_i + (1 - d_i) = 1$, we are ensured that X_i only "participates" once to the factorization. It is somehow as chemical equations where each element appears the same number of times on both equation sides, that we can model:

$$b(x_1, x_2, \ldots, x_N) \quad \longrightarrow \prod_{a=1}^{M} b_a(\mathbf{x}_a) \prod_{i=1}^{N} b_i^{1-d_i}(x_i) \tag{5.7}$$

In [J.S05], $\{c_r\}_r$ are called counting numbers, and authors introduced the concept of *validity* of a region-based approximation when it satisfies this principle. In the following, all region-based approximations will fulfill this rule.

5.2.2.1 Computation rule

For any couple of regions (r, s) in a given regions set $\mathcal{R}$, $s \subseteq r$ means that $\mathbf{X}_s \subseteq \mathbf{X}_r$ and $\mathbf{C}_s \subseteq \mathbf{C}_r$. We previously mentioned that regions distributions have to be weighted not to over count inner variable nodes and potentials in the factorization of the joint distribution (5.4). This is modeled according to following equations:

$$\forall X_i \in \mathbf{X}, \qquad \sum_{\substack{r \in \mathcal{R} \\ X_i \in \mathbf{X}_r}} c_r = 1, \tag{5.8}$$

$$\forall c_a \in \mathbf{C}, \qquad \sum_{\substack{r \in \mathcal{R} \\ c_a \in \mathbf{C}_r}} c_r = 1 \tag{5.9}$$

These equations can be rephrased for regions, such that any region only contributes once to the region-based factorization:

$$\forall r \in \mathcal{R}, \quad \sum_{s \supseteq r} c_s = 1 \tag{5.10}$$

which can be rephrased as:

$$\forall r \in \mathcal{R}, \quad c_r = 1 - \sum_{s \supset r} c_s = 1 \tag{5.11}$$

This helps compute counting numbers. First of all, any region r that is not included in any other region of the same regions set $\mathcal{R}$ has a unit counting number according to the previous equation. Afterward, we can compute any region that is included in other regions of unit counting numbers, and so on. We iteratively compute counting numbers for all regions according to (5.11).

§ 5.3 Region-based approximation

Contrary to Bethe approximation that provides a unique joint belief, region-based approximation is not so restrictive. Constructed from cluster variation method of Kikuchi [R. 51], it offers us numerous choices of the regions set $\mathcal{R}$ to factorize $b(\mathbf{x})$. As indicated in previous chapter, variable nodes distributions are not available, that is still true for regions. The joint distribution is accordingly approximated by a joint belief computed with a family of functions $\{b_r(\mathbf{x}_r)\}_{r,\mathbf{x}_r}$ that are marginal beliefs on regions:

$$b(\mathbf{x}) = \prod_{r \in \mathcal{R}} b_r^{c_r}(\mathbf{x}_r) \tag{5.12}$$

Region-based approximation allows us to extend MF approach to longer ranges than Bethe approximation. Any family of functions $\{b_r(\mathbf{x}_r)\}_{r,\mathbf{x}_r}$ is then expected to provide better performance than a family $\{b_a(\mathbf{x}_a)\}_{a,\mathbf{x}_a}, \{b_i(x_i)_{i,x_i}\}_{i,x_i}$.

5.3.1 Example

We present here an evaluation of the accuracy of Mean-Field (MF), Bethe Approximation (BA) and Region-Based Approximation (RBA). Following the experimental protocol of [J.S05], we consider a square lattice of size $N = 9$, see Fig.5.1, with soft factors. More precisely, as is commonly regarded in physics of spin glasses, we consider that for each node c_a:

$$f_a(x_i, x_j) = \begin{bmatrix} e^{J_{ij}} & e^{-J_{ij}} \\ e^{-J_{ij}} & e^{J_{ij}} \end{bmatrix}$$

where $\{J_{ij}\}_{<i,j>}$ are centered Gaussian variables of variance σ used as the control parameter of the experiment. We display in Fig.5.4 the average error:

$$\epsilon = \frac{\sum_{\mathbf{x}} |b(\mathbf{x}) - p(\mathbf{x})|}{2^N} \tag{5.13}$$

for every approximation. First of all, we see that the MF error is the largest one. It is consistent with the fact that MF does not consider any graphical link between variable nodes, *i.e.* it considers them independent although they are correlated[4]. We observe then that RBA is globally closer to the true distribution than other approximations. We also observe that the BA reaches a maximum near $\sigma = 1.4$ that is not an expected phenomenon, but we do not investigate it here because it does not deal with the current problem.

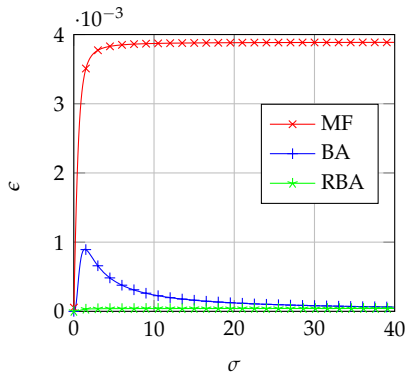


Figure 5.4: *Average errors of MF, Bethe and region-based approximations on square lattice $N = 9$*

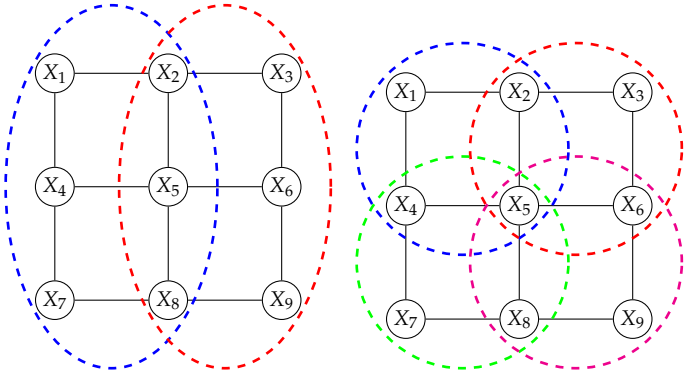
5.3.2 Region graph

Any factorization stemming from RBA (5.12) is graphically represented by a Bayesian network, *e.g.* Fig.5.2 and Fig.5.3, which is called *region-graph*, any region r being one of its node. To present construction rules of such a graph, specific subsets have to be defined. Given a regions set $\mathcal{R}$ and $r \in \mathcal{R}$:

[4] more precisely, couplings $\{J_{ij}\}_{<i,j>}$ are not ignored but probabilistic correlations are, *i.e.* any $< X_i, X_j >$ is approximated by $< X_i >< X_j >$

- $\mathcal{F}_r = \{q \in \mathcal{R} | q \subseteq r\}$ is family of r,
- $\mathcal{D}_r = \mathcal{F}_r \backslash r$ are descendants of r,
- $\mathcal{E}_r = \{q \in \mathcal{D}_r | \ \nexists s \in \mathcal{D}_r, q \subset s \subset r\}$ are children of r,
- $\mathcal{P}_r = \{q \in \mathcal{R} \backslash \mathcal{F}_r | r \in \mathcal{E}_q\}$ are parents of r,
- $\mathcal{A}_r = \{q \in \mathcal{R} \backslash \mathcal{F}_r | r \in \mathcal{D}_q\}$ are ancestors of r,

Any region-graph $\mathcal{R} = \{\mathcal{R}_l\}_{1 \leq l \leq L}$ is constructed level by level, or generation by generation, a level $\mathcal{R}_l$ being a set of regions that have the same number of variable nodes and check nodes. First level contains regions called *clusters*, arbitrary selected according to one rule: all variables nodes and check nodes have to be included in these clusters. We say that clusters, or the *clustering*, entirely *cover* the factor graph, see Fig.5.5.



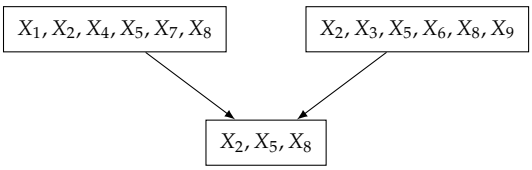
(a) Clustering by adjacent squares (b) Clustering by squares

Figure 5.5: *Two clusterings for two factorizations*

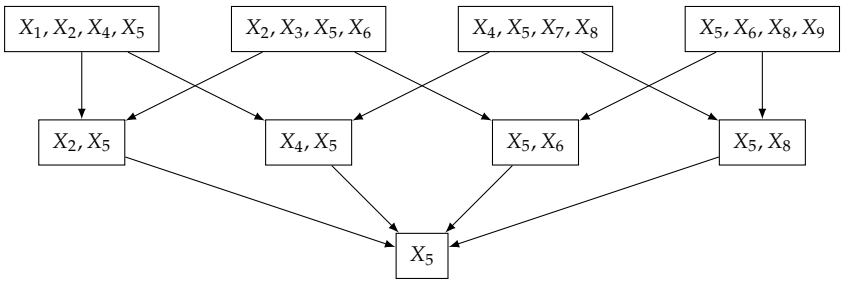
A level $\mathcal{R}_l$ distinct from the first level is constructed according to the Bayes rule: we search for all intersections between regions of $\mathcal{R}_{l-1}$. In other words, we exhibit common children between regions of $\mathcal{R}_{l-1}$:

$$r = \mathbf{X}_r \cup \mathbf{C}_r \in \mathcal{R}_l \quad \Longleftrightarrow \quad \exists (p,q) \in \mathcal{R}_{l-1}^2 \quad \text{s.t.} \quad \begin{array}{rcl} \mathbf{X}_r & = & \mathbf{X}_p \cap \mathbf{X}_q \\ \mathbf{C}_r & = & \mathbf{C}_p \cap \mathbf{C}_q \end{array}$$

Edges of the region-graph represent links between any region and its parents. The degree of freedom in a region-graph construction is then the Tanner graph clustering. Considering example in Fig.5.5, we can exhibit at least two region-graphs Fig.5.2 and Fig.5.3 constructed from two different clusterings. The way clusters are picked out widely influences RBA performance. In previous chapter we highlighted that the BP algorithm was suboptimal because of loop-like topology of most LDPC codes. RBA is aimed at absorbing harmful topological structures, *e.g.* loops and trapping sets, in clusters in such a way that they are quenched: no messages and then no information run on their Tanner graph edges. Thus, effects of topological damageable structures are hoped to be reduced or even annihilated, making region-based approximation a more efficient method than Bethe approximation.



(a) Region-graph by clustering with adjacent squares



(b) Region-graph built by clustering with smallest squares

Figure 5.6: *Two region-graphs for two factorizations*

As we already mentioned, the selection of clusters is not unique, the only conditions that must be fulfilled are:

- the whole Tanner graph has to be covered *i.e.* any variable node and any check node has to be included in at least one region,
- any check node cannot be separated from the variable nodes of its neighborhood.

A Tanner graph can then be clustered by a large variety of region-graphs. The difficult task is to single out the region-graph with the most accurate marginal beliefs. It is by no means a trivial task, as already mentioned in [J.S05] "how to optimally choose regions [...] remains at this point more an art than a science". In our study, we investigate this "art" for particular cases given that, to our knowledge, there is no general and efficient method for any graph.

§ 5.4 Generalized Belief Propagation

In this section, we make emerge the message-passing equations for any region-graph following the same method as Bethe approximation.

5.4.1 Variational free energy

As mentioned concerning Bethe approximation, the variational free energy is a cost function which minimization provides beliefs and messages equations. We recall that this function, denoted by F_K in the case of region-based approximation, is decomposed as the sum of the variational mean energy U_K and the variational entropy S_k:

$$F_K = U_K - S_K \tag{5.14}$$

with:

$$U_K = \sum_{\mathbf{x}} b(\mathbf{x}) E(\mathbf{x}) \tag{5.15}$$

$$S_K = -\sum_{\mathbf{x}} b(\mathbf{x}) \log b(\mathbf{x}) \tag{5.16}$$

We recall that the joint belief has the following expression:

$$b(\mathbf{x}) = \prod_{r \in \mathcal{R}} b_r^{c_r}(\mathbf{x}_r) \tag{5.17}$$

Therefore we obtain for the variational entropy:

$$S_K = -\sum_{r \in \mathcal{R}} c_r \sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) \log b_r(\mathbf{x}_r) \tag{5.18}$$

provided that the family is defined such that:

$$\forall r \in \mathcal{R}, \forall \mathbf{x}_r, \quad b_r(\mathbf{x}_r) = \sum_{\mathbf{x} \cup \mathbf{x}_r} b(\mathbf{x}) \tag{5.19}$$

Indeed, the equation (5.18) does not equal (5.16) if this condition is not respected. However, this condition is hard to handle, as mentioned in 4.7.1.2. To be consistent with the construction of Bethe approximation, we consider that the entropy (5.16) is approximated by a linear decomposition over regions entropies (5.18), that is the *region-based entropy*. The *region-based mean energy* is computed using the expression of the energy function from the previous chapter (4.26). It is possible expand it over regions:

$$E(\mathbf{x}) = \sum_{r \in \mathcal{R}} c_r E_r(\mathbf{x}_r) \tag{5.20}$$

with:

$$E_r(\mathbf{x}_r) = \sum_{c_a \in \mathbf{C}_r} E_a(\mathbf{x}_a) \tag{5.21}$$

We assign:

$$E_r(\mathbf{x}_r) = -\log f_r(\mathbf{x}_r) \tag{5.22}$$

that amounts to define a potential function of the region:

$$f_r(\mathbf{x}_r) = \prod_{c_a \in \mathbf{C}_r} f_a(\mathbf{x}_a) \tag{5.23}$$

Then the region-based mean energy is:

$$U_K = -\sum_{r \in \mathcal{R}} c_r \sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) \log f_r(\mathbf{x}_r) \tag{5.24}$$

We recall that this linear decomposition is exact as it stems from the energy function expression and not from the belief factorization, contrary to

the region-based entropy. Finally the region-based free energy becomes:

$$F_K = \sum_{r \in \mathcal{R}} c_r \sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) \log \frac{b_r(\mathbf{x}_r)}{f_r(\mathbf{x}_r)} \tag{5.25}$$

5.4.2 Optimization

As mentioned in previous chapter, we know that minimizing F_K under the following constraints provides marginal beliefs equations of regions:

- (C1) $\forall r \in \mathcal{R}, \quad \sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) = 1$,
- (C2) $\forall r \in \mathcal{R}, \forall s \in \mathcal{E}_r, \forall \mathbf{x}_s, \quad b_s(\mathbf{x}_s) = \sum_{\mathbf{x}_r \cup \mathbf{x}_s} b_r(\mathbf{x}_r)$.

We include neither inequality constraints on the beliefs, nor global normalizations and marginalizations, see previous chapter for details. We obtain the Lagrange function:

$$\mathcal{L}_K = F_k \quad + \quad \sum_{r \in \mathcal{R}} \gamma_r \left(\sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) - 1 \right) \tag{5.26}$$

$$+ \quad \sum_{r \in \mathcal{R}} \sum_{s \in \mathcal{E}_r} \sum_{\mathbf{x}_s} \lambda_{rs}(\mathbf{x}_s) \left(\sum_{\mathbf{x}_r \cup \mathbf{x}_s} b_r(\mathbf{x}_r) - b_s(\mathbf{x}_s) \right) \tag{5.27}$$

where $\{\gamma_r\}_r$ and $\{\lambda_{rs}(\mathbf{x}_s)\}_{r,s,\mathbf{x}_s}$ are Lagrange multipliers associated with constraints. Anyone can check that RBA satisfies KKT assumptions, as Bethe approximation. Lagrange multipliers are found such that critical points, *i.e.* beliefs that nullify the gradient of $\mathcal{L}_K$, are:

$$\boxed{b_r^*(\mathbf{x}_r) = f_r(\mathbf{x}_r) e^{-1} e^{-\frac{1}{c_r}\gamma_r} \prod_{s \in \mathcal{E}_r} e^{-\frac{1}{c_r}\lambda_{rs}(\mathbf{x}_s)} \prod_{q \in \mathcal{P}_r} e^{\frac{1}{c_r}\lambda_{qr}(\mathbf{x}_r)}} \tag{5.28}$$

Considering observations $\mathbf{Y}_r = \{Y_i | X_i \in \mathbf{X}_r\}$ and Bayes rule, we obtain the *a posteriori* distribution:

$$b_r^*(\mathbf{x}_r, \mathbf{y}_r) = f_r(\mathbf{x}_r) l_r(\mathbf{x}_r, \mathbf{y}_r) e^{-1} e^{-\frac{1}{c_r}\gamma_r} \prod_{s \in \mathcal{E}_r} e^{-\frac{1}{c_r}\lambda_{rs}(\mathbf{x}_s)} \prod_{q \in \mathcal{P}_r} e^{\frac{1}{c_r}\lambda_{qr}(\mathbf{x}_r)} \tag{5.29}$$

where:

$$l_r(\mathbf{x}_r, \mathbf{y}_r) = \prod_{X_i \in \mathbf{X}_r} l_i(x_i, y_i) \tag{5.30}$$

5.4.3 Rotation of the Lagrange multipliers

Bethe approximation stipulated that the belief $\{b_a(\mathbf{x}_a)\}_{\mathbf{x}_a}$ of a check node $c_a \in \mathbf{C}$ was the product of all incoming messages in any of its neighboring variable nodes (4.58). We also saw that these messages were directly related to Lagrange multipliers. Thus, generalizing it for a region would be that the belief $\{b_r(\mathbf{x}_r)\}_{\mathbf{x}_r}$ of a region $r \in \mathcal{R}$ is the product of all incoming messages in any region q of its family $\mathcal{F}_r$, messages being equivalently related to the Lagrange multipliers $\{\lambda_{qr}(\mathbf{x}_r\}_{q,\mathbf{x}_r}$. However, this assumption is not consistent

with equation (5.28). Here we demonstrate that an equivalent formulation of the constraints may make emerge the right generalization of the Bethe approximation. First of all, we define Lagrange multipliers $\{\mu_{pr}(\mathbf{x}_r)\}_{p,r,\mathbf{x}_r}$ associated with the alternative constraint:

$$\text{(C2')}\ \forall r \in \mathcal{R}, \forall p \in \mathcal{P}_r, \forall \mathbf{x}_r, \quad c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{A}_r \setminus p \cup \mathcal{A}_p} c_q \sum_{\mathbf{x}_q \cup \mathbf{x}_r} b_q(\mathbf{x}_q) = 0$$

Then we prove that constraint (C2) associated with multipliers $\{\lambda_{pr}(\mathbf{x}_r)\}_{p,r,\mathbf{x}_r}$ is equivalent to constraint (C2').

5.4.3.1 From λ to μ

We consider an arbitrary region $r \in \mathcal{R}$. We recall that its counting number c_r is given by:

$$c_r = 1 - \sum_{q \in \mathcal{A}_r} c_q \tag{5.31}$$

We rephrase it such that:

$$c_r + \sum_{q \in \mathcal{A}_r} c_q = 1 \tag{5.32}$$

We consider a parent region p of r, then:

$$c_p + \sum_{q \in \mathcal{A}_p} c_q = 1 \tag{5.33}$$

Subtracting (5.33) to (5.32), we obtain:

$$c_r + \sum_{q \in \mathcal{A}_r \setminus p \cup \mathcal{A}_p} c_q = 0 \tag{5.34}$$

Then (5.34)$\times b_r(\mathbf{x}_r)$ gives:

$$c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{A}_r \setminus p \cup \mathcal{A}_p} c_q b_r(\mathbf{x}_r) = 0 \tag{5.35}$$

We recall that constraint (C2) associated with $\{\lambda_{qr}(\mathbf{x}_r)\}_{q,\mathbf{x}_r}$ is assumed to be true. Therefore we obtain:

$$c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{A}_r \setminus p \cup \mathcal{A}_p} c_q \sum_{\mathbf{x}_q \cup \mathbf{x}_r} b_q(\mathbf{x}_q) = 0 \tag{5.36}$$

that is constraint (C2').

5.4.3.2 From μ to λ

This part of the equivalence is induction. We consider a region $r \in \mathcal{R}_2$, and two of its parents p_1, p_2. We assume that constraint (C2') is true:

$$c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{P}_r \setminus p_1} \sum_{\mathbf{x}_q \cup \mathbf{x}_r} b_q(\mathbf{x}_q) = 0 \tag{5.37}$$

and:

$$c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{P}_r \setminus p_2} \sum_{\mathbf{x}_q \cup \mathbf{x}_r} b_q(\mathbf{x}_q) = 0 \tag{5.38}$$

Subtracting (5.38) to (5.37) gives:

$$\sum_{\mathbf{x}_{p_1} \cup \mathbf{x}_r} b_{p_1}(\mathbf{x}_{p_1}) = \sum_{\mathbf{x}_{p_2} \cup \mathbf{x}_r} b_{p_2}(\mathbf{x}_{p_2}) \tag{5.39}$$

In other words, marginalization does not depend on the parent of r. The fact that $r \in \mathcal{R}_2$ involves that:

- $c_r = 1 - N_p$, with N_p the number of parents of r,
- $\forall p \in \mathcal{P}_r, \quad c_p = 1$.

Considering a parent p of r, equation (5.37) becomes:

$$(1 - N_p) b_r(\mathbf{x}_r) + (N_p - 1) \sum_{\mathbf{x}_p \cup \mathbf{x}_r} b_p(\mathbf{x}_p) = 0 \tag{5.40}$$

that implies:

$$b_r(\mathbf{x}_r) = \sum_{\mathbf{x}_p \cup \mathbf{x}_r} b_p(\mathbf{x}_p) \tag{5.41}$$

provided that $N_p > 1$[5]. Finally we obtain constraint (C2) associated with $\lambda_{pr}(\mathbf{x}_r)$.

First part of the induction is demonstrated, now we have to work on a region r of an arbitrary generation $\mathcal{R}_l$. We can prove by another induction [J.S05],[R.J02] that:

$$(c_r - 1) b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{A}_r} c_q \sum_{\mathbf{x}_q \setminus \mathbf{x}_r} b_q(\mathbf{x}_q) = 0 \tag{5.42}$$

Equations (5.42) and (C2') imply that for any ancestor t of r:

$$c_r b_r(\mathbf{x}_r) + \sum_{q \in \mathcal{A}_r} c_q \sum_{\mathbf{x}_q \cup \mathbf{x}_r} b_q(\mathbf{x}_q) - \sum_{\mathbf{x}_t \cup \mathbf{x}_r} b_t(\mathbf{x}_t) = 0 \tag{5.43}$$

For any couple of ancestors t_1, t_2 of r, subtracting (5.43) for t_1 to (5.43) for t_2 results in::

$$\sum_{\mathbf{x}_{t_1} \cup \mathbf{x}_r} b_{t_1}(\mathbf{x}_{t_1}) = \sum_{\mathbf{x}_{t_2} \cup \mathbf{x}_r} b_{t_2}(\mathbf{x}_{t_2}) \tag{5.44}$$

In words, it means that marginalization does not depend on the selected ancestor, for any region in $\mathcal{R}$. It implies in (C2') that for any ancestor (to some extent for any parent) p:

$$b_r(\mathbf{x}_r) = \sum_{\mathbf{x}_p \cup \mathbf{x}_r} b_p(\mathbf{x}_p) \tag{5.45}$$

which is constraint associated with the Lagrange multiplier $\lambda_{pr}(\mathbf{x}_r)$.

[5] which is true, given that if $N_p \leq 1$, r cannot exist

As we proved that constraints (C2) and (C2') are equivalent, the Lagrangian function of F_K is rephrased:

$$\begin{aligned}\mathcal{L}_K = F_k \quad &+ \quad \sum_{r\in\mathcal{R}} \gamma_r \left(\sum_{\mathbf{x}_r} b_r(\mathbf{x}_r) - 1 \right) \\ &+ \quad \sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \sum_{\mathbf{x}_r} \mu_{pr}(\mathbf{x}_r) \left(c_r b_r(\mathbf{x}_r) + \sum_{q\in\mathcal{A}_r\setminus p\cup\mathcal{A}_p} c_q \sum_{\mathbf{x}_q\cup\mathbf{x}_r} b_q(\mathbf{x}_q) \right)\end{aligned} \tag{5.46}$$

whose stationary points are:

$$\boxed{b_r^*(\mathbf{x}_r) = f_r(\mathbf{x}_r) e^{-1} e^{-\frac{\gamma_r}{c_r}} \prod_{p\in\mathcal{P}_r} e^{-\mu_{pr}(\mathbf{x}_r)} \prod_{q\in\mathcal{D}_r} \prod_{s\in\mathcal{P}_q\setminus\mathcal{F}_r} e^{-\mu_{sq}(\mathbf{x}_q)}} \tag{5.47}$$

which correspond to a generalization of Bethe approximation we searched for. Considering observations $\mathbf{Y}_r = \{Y_i | X_i \in \mathbf{X}_r\}$ and Bayes rule, we obtain the *a posteriori* distribution:

$$b_r^*(\mathbf{x}_r, \mathbf{y}_r) = l_r(\mathbf{x}_r, \mathbf{y}_r) f_r(\mathbf{x}_r) e^{-1} e^{-\frac{\gamma_r}{c_r}} \prod_{p\in\mathcal{P}_r} e^{-\mu_{pr}(\mathbf{x}_r)} \prod_{q\in\mathcal{D}_r} \prod_{s\in\mathcal{P}_q\setminus\mathcal{F}_r} e^{-\mu_{sq}(\mathbf{x}_q)} \tag{5.48}$$

5.4.4 Message-passing

Belief equation (5.48) on a region r is, in words, the product of all information concerning it and its family. This information comes from:

- the code itself represented by $f_r(\mathbf{x}_r)$,
- the channel represented by $l_r(\mathbf{x}_r, \mathbf{y}_r)$,
- the rest of the region-graph represented by all the Lagrange multipliers $\{\mu_{pr}(\mathbf{x}_r)\}_{p\in\mathcal{P}_r,\mathbf{x}_r}$, $\{\mu_{sq}(\mathbf{x}_q)\}_{q\in\mathcal{D}_r,s\in\mathcal{P}_q\setminus\mathcal{F}_r,\mathbf{x}_q}$

In other words we can summarize it in the following formula:

$$b_r(\mathbf{x}_r, \mathbf{y}_r) = l_r(\mathbf{x}_r, \mathbf{y}_r) f_r(\mathbf{x}_r) m_{\mathcal{R}\to r}(\mathbf{x}_r) \tag{5.49}$$

where $m_{\mathcal{R}\to r}(\mathbf{x}_r)$ is information coming from the region-graph that can be re-written as:

$$b_r(\mathbf{x}_r, \mathbf{y}_r) = l_r(\mathbf{x}_r, \mathbf{y}_r) f_r(\mathbf{x}_r) \prod_{\substack{q\in\mathcal{R}\setminus\mathcal{F}_r \\ s\in\mathcal{F}_r}} m_{qs}(\mathbf{x}_s) \tag{5.50}$$

where the product implies an implicit condition that regions q is a parent of region s, *i.e.* (q, s) is an edge of the region-graph. Quantity m_{qs} is a message from q to s, closely related to Lagrange multiplier on the same edge such that:

$$m_{qs}(\mathbf{x}_s) = e^{-\mu_{qs}(\mathbf{x}_s)} \tag{5.51}$$

However, values of Lagrange multipliers are not available, this is the dual problem of the optimization. Not solved in the literature, to our knowledge, we shall deal with an iterative solution, as with Bethe approximation.

We make use of the local marginalization constraint (C2) to extract an update equation of a message between two regions (r, s) linked by an edge. We get:

$$\boxed{m_{rs}(\mathbf{x}_s, \mathbf{y}_s) = \frac{\sum_{\mathbf{x}_r \cup \mathbf{x}_s} l_{r\setminus s}(\mathbf{x}_{r\setminus s}, \mathbf{y}_{r\setminus s}) f_{r\setminus s}(\mathbf{x}_r) \prod_{\substack{u \in \mathcal{R}\setminus\mathcal{F}_r \\ v \in \mathcal{F}_r\setminus\mathcal{F}_s}} m_{uv}(\mathbf{x}_v, \mathbf{y}_v)}{\prod_{\substack{u \in \mathcal{D}_r\setminus\mathcal{F}_s \\ v \in \mathcal{D}_s}} m_{uv}(\mathbf{x}_v, \mathbf{y}_v)}} \tag{5.52}$$

where:

$$l_{r\setminus s}(\mathbf{x}_{r\setminus s}, \mathbf{y}_{r\setminus s}) = \prod_{X_i \in \mathbf{X}_r\setminus\mathbf{X}_s} l_i(x_i, y_i) \tag{5.53}$$

The message-passing that runs on the region-graph is a generalization of the BP algorithm that runs on the factor graph, thus it is called the *Generalized Belief Propagation* algorithm (GBPA).

5.4.4.1 Example

We consider the Tanner graph in Fig.5.7[6]. We cluster it with squares of four check nodes, that provides us the region-graph in Fig.5.8.

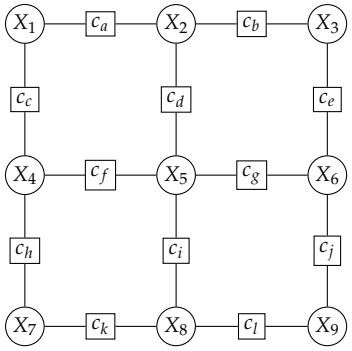


Figure 5.7: *Explicit square lattice of length $N = 9$*

According to equation (5.52), message from region r, with $\mathbf{C}_r = \{c_b, c_d, c_e, c_g\}$ and $\mathbf{X}_r = \{X_2, X_3, X_5, X_6\}$, to region s, with $\mathbf{C}_s = c_g$ and $\mathbf{X}_s = \{X_5, X_6\}$, is:

$$m_{2356\to 56}(\mathbf{x}_{56}, \mathbf{y}_{56}) = \frac{\sum_{\mathbf{x}_{23}} c_b(\mathbf{x}_{23}) c_d(\mathbf{x}_{25}) c_e(\mathbf{x}_{36}) l_2(x_2, y_2) l_3(x_3, y_3) m_{1245\to 25}(\mathbf{x}_{25}, \mathbf{y}_{25})}{m_{25\to 5}(x_5, y_5)} \tag{5.54}$$

[6] which is the explicit representation of the square lattice, *i.e.* we exhibit the check nodes that belong to the edges

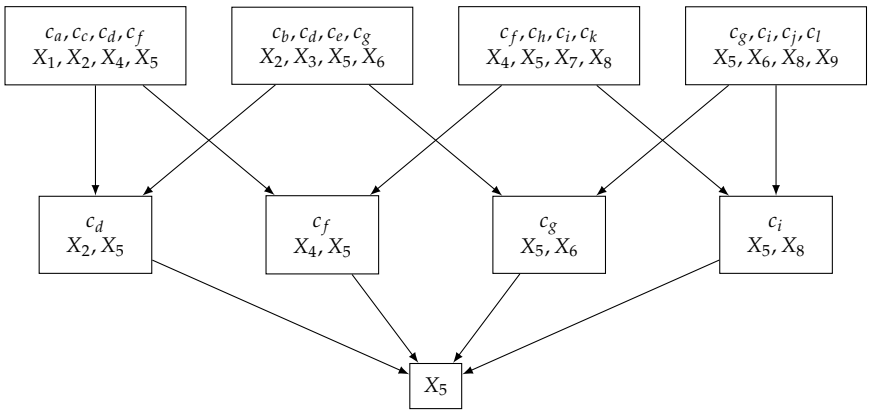


Figure 5.8: *Region-graph of the explicit square lattice*

5.4.4.2 Complexity

For a given couple $(r, s \in \mathcal{E}_r)$, the number of elementary operations required to compute message $m_{rs}(\mathbf{x}_s)$, for a given $\mathbf{x}_s$, is near $2^{|\mathbf{X}_r|-|\mathbf{X}_s|}$. The total number of elementary operations to compute all messages on edge $(r \to s)$ is accordingly $n_{eo}(r) = 2^{|\mathbf{X}_r|}$. Therefore, it is necessary to cluster any Tanner graph whose clusters are small enough such that $n_{eo}(r)$ is reasonable for any region r. This is the common trade-off of coding theory:

- too large clusters allow us to absorb and annihilate most loops but they induce a GBP with a very high complexity,
- too small clusters do not help reducing loops effects but they induce a GBP with a competitive complexity.

The GBP algorithm is not currently of a practical interest. To our knowledge, there is no paper in the literature that deals with its complexity. Update equations cannot be as easily reduced as BP ones, because messages are not two-dimensional vectors with binary arguments but multi-dimensional vectors. Nevertheless, the GBP algorithm must be investigated firstly because there are many of its properties that are still concealed behind update equations, then because it reflects the difficulty to solve inference on most factor graphs. Even though its implementation would ask for unreasonable hardware components, it provides us very relevant information about the effects of topology.

5.4.5 Region-graph reduction

Any loopy region-graph cannot perform optimally, only loopfree ones offer optimal approximation as marginal belief on any region r can be derived as:

$$b_r(\mathbf{x}_r, \mathbf{y}_r) = l_r(\mathbf{x}_r, \mathbf{y}_r) f_r(\mathbf{x}_r) \sum_{\mathbf{x} \cup \mathbf{x}_r} \prod_{X_i \notin \mathbf{X}_r} l_i(x_i, y_i) \prod_{c_a \notin \mathbf{C}_r} f_a(\mathbf{x}_a) \tag{5.55}$$

that is precisely the a posteriori probability distribution $p_r(\mathbf{x}_r, \mathbf{y}_r)$. In [P. 03], Pakzad proposed a method to modify the region-graph removing few edges

according to a specific rule. We call this method the *Pakzad reduction* of the region-graph. The method is based on the fact that both Lagrange functions (5.26) (5.46) contain few linearly dependent constraints. In optimization theory [D.P01] is stipulated that all constraints must be linearly independent to avoid redundancy. Pakzad then removed corresponding redundant constraints in Lagrange functions. As an example, we consider the small region-graph in Fig.5.9.

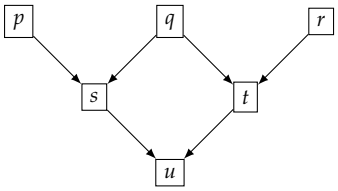


Figure 5.9: *Small region-graph*

Part of the Lagrange function dedicated to local marginalization constraints (C2) is made with following equations:

$$\forall \mathbf{x}_s, \quad \sum_{\mathbf{x}_p \cup \mathbf{x}_s} b_p(\mathbf{x}_p) = b_s(\mathbf{x}_s) \tag{5.56}$$

$$\forall \mathbf{x}_s, \quad \sum_{\mathbf{x}_q \cup \mathbf{x}_s} b_q(\mathbf{x}_q) = b_s(\mathbf{x}_s) \tag{5.57}$$

$$\forall \mathbf{x}_t, \quad \sum_{\mathbf{x}_q \cup \mathbf{x}_t} b_q(\mathbf{x}_q) = b_t(\mathbf{x}_t) \tag{5.58}$$

$$\forall \mathbf{x}_t, \quad \sum_{\mathbf{x}_r \cup \mathbf{x}_t} b_r(\mathbf{x}_r) = b_t(\mathbf{x}_t) \tag{5.59}$$

$$\forall \mathbf{x}_u, \quad \sum_{\mathbf{x}_s \cup \mathbf{x}_u} b_s(\mathbf{x}_s) = b_u(\mathbf{x}_u) \tag{5.60}$$

$$\forall \mathbf{x}_u, \quad \sum_{\mathbf{x}_t \cup \mathbf{x}_u} b_t(\mathbf{x}_t) = b_u(\mathbf{x}_u) \tag{5.61}$$

Constraints (5.57) and (5.60) imply that:

$$\forall \mathbf{x}_u, \quad \sum_{\mathbf{x}_q \cup \mathbf{x}_u} b_q(\mathbf{x}_q) = b_u(\mathbf{x}_u) \tag{5.62}$$

However, constraints (5.58) and (5.61) also imply (5.62). As $t \in \mathcal{P}_u$, we affirm that (5.57), (5.60) and (5.58) imply (5.61). In other words, constraints (5.61) has to be removed from Lagrange function for it to satisfy the linear independence.

Removing local marginalization constraints between regions t and u means that the resulting GBP algorithm does not compute any message between them. The region-graph is then reduced to the one depicted in Fig.5.10. We obtain a loopfree region-graph with an exact RBA *i.e.* with an optimal GBP algorithm. Unfortunately, Pakzad reduction hardly results in loopfree region-graphs, *e.g.* we performed extensive experiments on Hamming, square lattice and Tanner codes, and reduction does not provide loopfree region-graphs. Simulations demonstrated that Pakzad reduction did not significantly improve RBA.

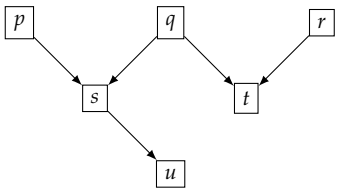


Figure 5.10: *Reduced region-graph*

Eventually, Pakzad reduction is a well-suited method for specific region-graphs, but not for all of them. However, it helps decrease computation time of GBP as it removes a few message computations. As it does not damage the performance but still improves computation time at least, we will use it in the following of the work.

5.4.6 Damping factor

A inherent problem of the GBPA is the lack of convergence it suffers from. To our knowledge, the literature does not provide any explanation concerning this phenomenon, but all experiments tend to prove it. In [J.S05] is introduced a damping on messages to avoid this problem, a well-known solution for many iterative algorithms. We denote by F_{rs} the implicit update function that computes the value of $m_{rs}^{(k)}$ for any k such that:

$$m_{rs}^{(k)}(\mathbf{x}_s, \mathbf{y}_s) = \frac{1}{2} F_{rs} + \frac{1}{2} m_{rs}^{(k-1)}(\mathbf{x}_s, \mathbf{y}_s) \tag{5.63}$$

As a matter of fact this equation is a uniform blend of memory $m_{rs}^{(k-1)}$ and update F_{rs}. There is still no visible reason to restrict the modification to a uniform damping, that is why we introduce a general equation:

$$\boxed{m_{rs}^{(k)}(\mathbf{x}_s, \mathbf{y}_s) = w_k F_{rs} + (1 - w_k) m_{rs}^{(k-1)}(\mathbf{x}_s, \mathbf{y}_s)} \tag{5.64}$$

where $0 \leq w_k \leq 1$ is the weight or damping factor. Currently, damping of the GBPA is not much tackled in the literature therefore we do not have many clues to determine properties it should verify. In [P. 03] is assumed that convergence of the GBP decoder is mostly ensured if w_k converges to zero as k is getting larger. It contradicts (5.63) but as no general theory has been written, we cannot testify the veracity of any assumption. Investigations may only be made of experiments.

This way, first idea is to use the simplest damping laws, that is to say we consider that w_k is constant, $w_k = w_0$ for any k. This assumption is against the one proposed in [P. 03] because it does not converge to zero but it is consistent with [J.S05]. After that, second idea is to use decreasing laws of w_k, which match with assumption of [P. 03] and not really with (5.63). It is a hard task to extract a global theory of a damped GBP, a very large set of codes should be used to highlight inherent properties of the decoder. In our work that is not dedicated to this problem, we have tested our own different damping laws on Hamming and Tanner codes.

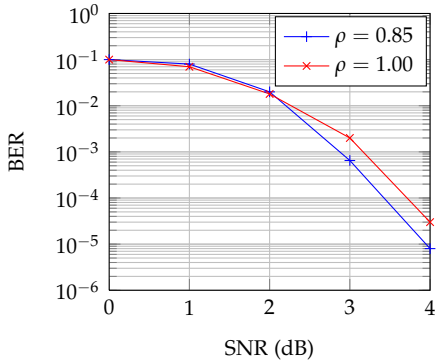


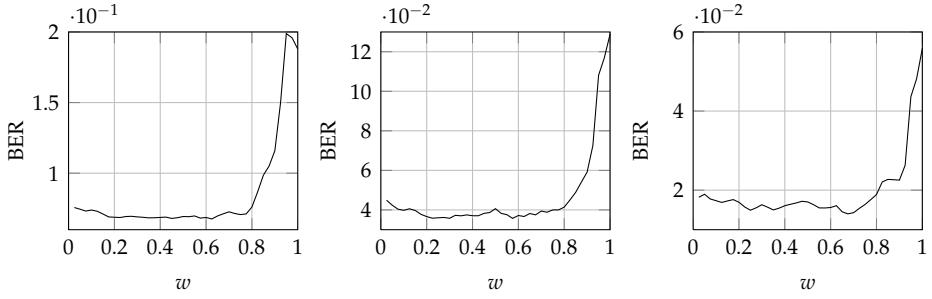
Figure 5.11: *Uniformly reweighted BP ($\rho = 1.00$ is BP)*

We highlight an important assumption that w_k does not depend on messages, that is not told in the literature whereas it is not so obvious. As an example, a very interesting work on a reweighted BPA (RBPA) has been presented in [T.Go8]. To each edge of the factor graph is assigned a particular weight, according to a particular method based on spanning trees, detailed in the paper. This improves convergence of the decoder. In [H. 12] and [H. 11] is introduced that a uniformly RBPA (URBPA), *i.e.* a BPA with all edges equally weighted by a value ρ. To extract the weight that offers the lowest BER, the only method is to run decoder for many values of ρ. Displayed in Fig.5.11 for an LDPC code of length $N = 256$ with $M = 128$, results show that the URBPA may significantly surpass the BPA for specific values of ρ. This experimental protocol provides the optimal weight, then we use it in our study to select the constant damping factor.

5.4.6.1 Constant damping factor

We consider that any message m_{rs} is updated according to the following equation:

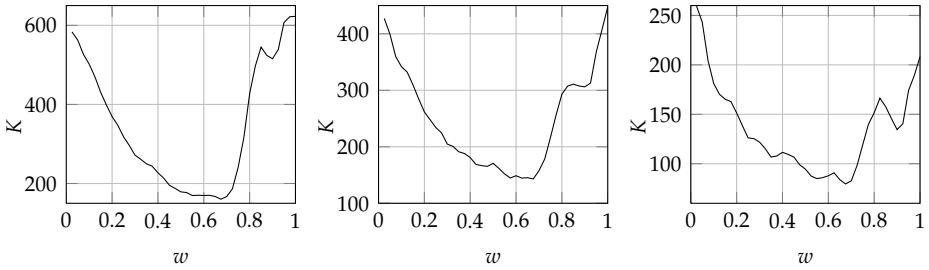
$$m_{rs}^{(k)}(\mathbf{x}_s, \mathbf{y}_s) = wF_{rs} + (1-w)m_{rs}^{(k-1)}(\mathbf{x}_s, \mathbf{y}_s) \tag{5.65}$$



(a) SNR = 1.0dB (b) SNR = 1.5dB (c) SNR = 2.0dB

Figure 5.12: *BER of the GBP with changing w_0*

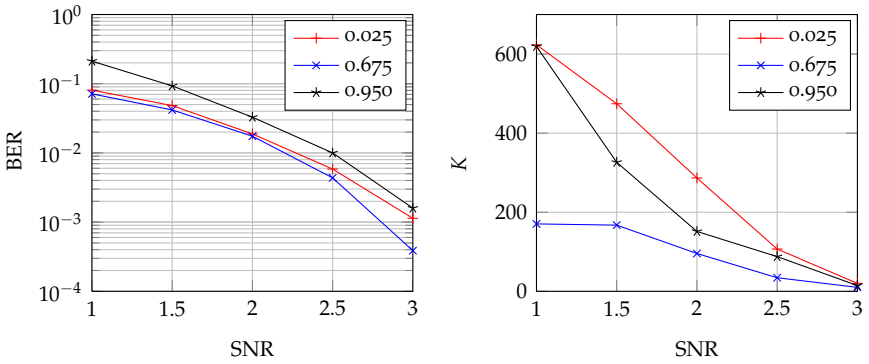
For low values of w_0, memory $m_{rs}^{(k-1)}$ takes the advantage over update F_{rs} whereas for high values of w_0 it is the other direction. We display in Fig.5.12 the BER evolution as a function of w, on Tanner code. Results are averaged over 1000 simulations with 1000 iterations maximum. First of all, we observe that value $w = 1.0$, *i.e.* pure GBP algorithm without any blend with memory, provides the worse BER. As a significant example, we see that BER at $w = 0.6$ for the three SNR values is three times less than BER at $w = 1.0$. After that, it turns out that the most relevant weight is given by the evolution of the number of iterations K before GBP convergence or appearance of a null syndrome, represented in Fig.5.13. Indeed for the three SNR values, the shape of K is similar, K reaches a minimum when $w \approx 0.675$.



(a) SNR = 1.0dB (b) SNR = 1.5dB (c) SNR = 2.0dB

Figure 5.13: *Number of iterations of the GBP with changing w_0*

We represent in Fig.5.14 the BER as a function of the SNR for three values of w. We see significant improvement in the BER, for a damping $w = 0.675$, as much as the number of iterations K.

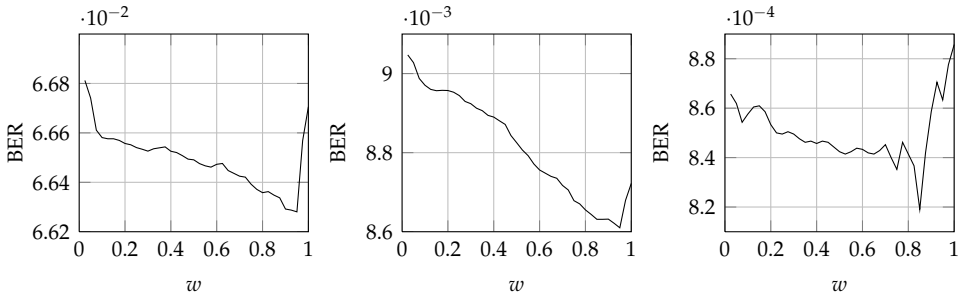


(a) BER (b) Number of iterations K

Figure 5.14: *Performance of the GBP for three weights*

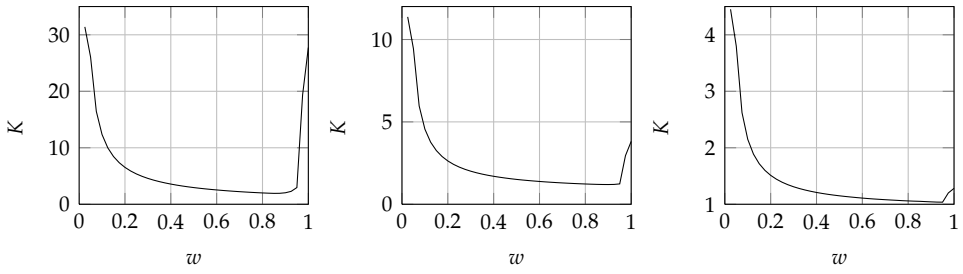
Coming back to the figures Fig.5.12 and Fig.5.13, a noteworthy observation is that low values of w do not provide the worst BER, even though corresponding values of K are very high. Furthermore, BER is quite unchanged when $w \geq 0.675$. In other words, pure update F_{rs} is less accurate than memory.

This leads us to consider a varying damping factor with k, that offers smoother relations between update and memory. We display in Fig.5.15 and Fig.5.16 the BER and the number of iterations of GBP, respectively, on Hamming code with a varying damping factor. We observe that the minimum of the BER is reaches for $w \approx 0.85$, that also corresponds with the minimum number of iterations K. Contrary to Tanner code, there is no obvious evidence that would validate the assumption for the Tanner code that stipulates that pure update is less accurate than memory.



(a) SNR = 0.5dB (b) SNR = 3.5dB (c) SNR = 5.5dB

Figure 5.15: *BER of the GBP with changing w_0*



(a) SNR = 0.5dB (b) SNR = 3.5dB (c) SNR = 5.5dB

Figure 5.16: *Number of iterations of the GBP with changing w_0*

5.4.6.2 Decreasing damping factor

Now we consider that the damping factor follows a non-constant evolution that converges to zero. We selected three different profiles depicted in Fig.5.17:

- the parabolic factor (P) of order n is: $w_k = w_0(1 - \frac{k}{K})^n$,
- the inverse parabolic factor (IP) of order n is: $w_k = w_0 \left(1 - (\frac{k}{K})^n\right)$,
- the affine factor (A) is: $w_k = w_0 - \frac{k}{K} w_0$,

The affine factor (A) is only a particular case of the parabolic one, or the inverse parabolic one, with $n = 1$.

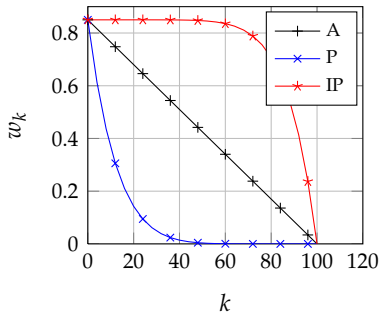
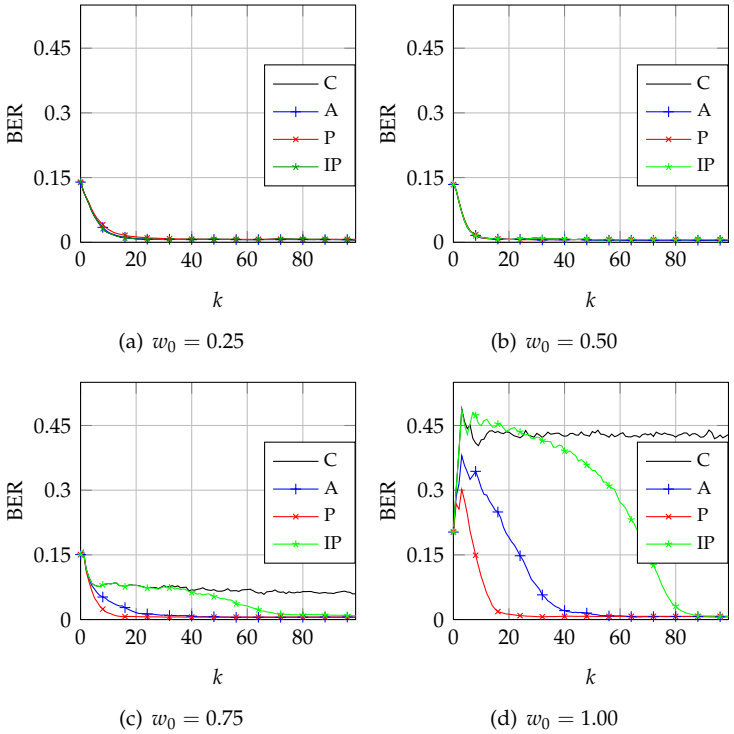


Figure 5.17: *Damping factor with initial value* $w_0 = 0.85$

The key point of w_k is the balance between memory $m_{rq}^{(k-1)}$ and update F_{rq}. For functions we introduce, this balance is different from case to case. In the case of a parabolic function, memory is quickly favored at the expense of update, whereas in the case of an inverse parabolic function, memory is almost ignored for a long time. The affine damping coefficient offers a quite balance distribution, in terms of k, between memory and update. On figures Fig.5.18 are displayed the BER on Tanner code for the different damping laws with an SNR $= 2.90$dB and order $n = 2$ of P and IP factors.



(a) $w_0 = 0.25$ (b) $w_0 = 0.50$

(c) $w_0 = 0.75$ (d) $w_0 = 1.00$

Figure 5.18: *BER of the GBP algorithm on Tanner code with decreasing damping factors*

We observe that all the laws, except the constant one, allows GBP to converge to the same BER, the difference remains in the convergence speed. The fact is that the parabolic coefficient provides the fastest GBP algorithm, as it is significantly visible for high values of w_0. This property is quite surprising because it means that update functions of GBP has to be ignored as quickly as possible for the benefit of memory. This process seems unnatural because any decoding algorithm is aimed at keeping control on information that pass along edges of the associated graph. Finally, it means that the GBP algorithm with the parabolic damping can be considered as a "starter" decoding algorithm.

On Fig.5.19 we show the BER for Hamming code with a parabolic damping factor corresponding to $n = 2$ and $w_0 = 0.855$ and the BER any without damping.

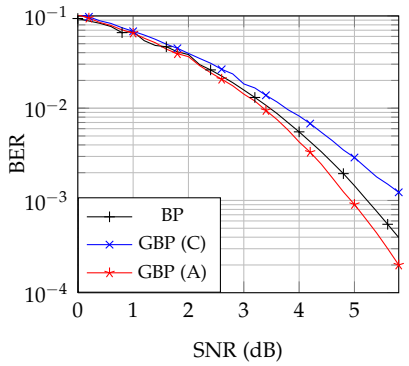


Figure 5.19: *BER of the GBP algorithm on the Hamming code*

It turns out that the original version is even worse than what the BP algorithm provides. On the contrary, the parabolic damping factor helps the GBP algorithm be more accurate. It should be recommended to carry the same investigation for many other LDPC codes to extract the best damping law. It appears in the current case that the parabolic damping factor with an initial value greater than 0.5 is a very good candidate to improve GBP performance.

5.4.7 Iteration index

Equation (5.52) is not practical because it does not carry any iterative data. According to local marginalization constraints, for any couple of regions $r \in \mathcal{R}$, $s \in \mathcal{E}_r$:

$$b_s(\mathbf{x}_s) = \sum_{\mathbf{x}_r \cup \mathbf{x}_s} b_r(\mathbf{x}_r) \tag{5.66}$$

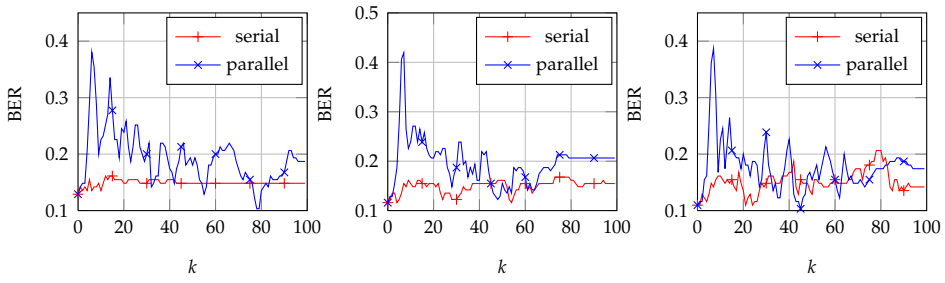
that means that left side is computed by means of right side of the equation. More precisely, this equation can be rephrased as a sequence:

$$b_s^{(k+1)}(\mathbf{x}_s) = \sum_{\mathbf{x}_r \cup \mathbf{x}_s} b_r^{(k)}(\mathbf{x}_r) \tag{5.67}$$

where k is an iterative index. Thus equation (5.52) is practically:

$$m_{rs}^{(k+1)}(\mathbf{x}_s, \mathbf{y}_s) = \frac{\sum_{\mathbf{x}_r \cup \mathbf{x}_s} l_{r\setminus s}(\mathbf{x}_{r\setminus s}, \mathbf{y}_{r\setminus s}) f_{r\setminus s}(\mathbf{x}_r) \prod_{\substack{u \in \mathcal{R}\setminus\mathcal{F}_r \\ v \in \mathcal{F}_r\setminus\mathcal{F}_s}} m_{uv}^{(k)}(\mathbf{x}_v, \mathbf{y}_v)}{\prod_{\substack{u \in \mathcal{D}_r\setminus\mathcal{F}_s \\ v \in \mathcal{D}_s}} m_{uv}^{(k+1)}(\mathbf{x}_v, \mathbf{y}_v)} \tag{5.68}$$

It is not a strict update from k to $k+1$ given that on the right side of the equation are left messages at iteration k and messages at iteration $k+1$. From a practical point of view, it is advantageous to search for a way to parallelize the GBP as it would speed it up. At first sight, we could think of approximating (5.68) forcing the whole right side of the equation to be at iteration k. Doing so would allow us to implement a version where all messages at iteration k are computed in parallel, which would dramatically reduce the computation time. We show in Fig.5.20 the BER of the GBP algorithm from the parallel version, denoted by GBP_p, and the original one, also called the serial one, denoted by GBP_s, on Tanner code with the same initialization.



(a) SNR = 2.00dB (b) SNR = 2.25dB (c) SNR = 2.50dB

(d) SNR = 2.75dB (e) SNR = 3.00dB (f) SNR = 3.20dB

Figure 5.20: *Parallel version performance*

Without any ambiguity, GBP_p offers poor results compared with GBP_s, the BER is significantly damaged. Furthermore, except for SNR = 2.75dB, the GBP_p algorithm does not tangibly converge whereas the GBP_s algorithm quickly converges towards the final state it will end in. We observe for SNR = 3.20dB, which is a quite high SNR value for this code, that GBP_p has not converged yet whereas GBP_s has converged in less than 20 iterations.

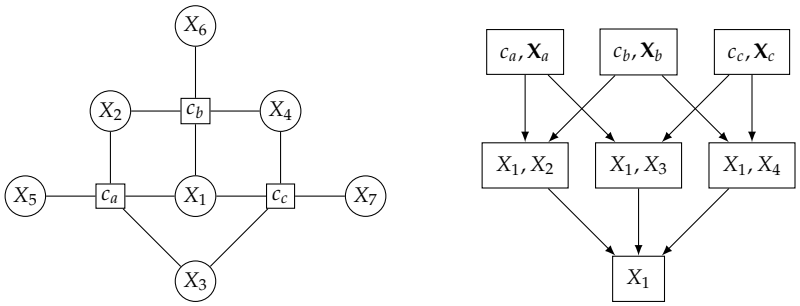
Finally, experiments show that the parallelization of the GBP algorithm should not be done at once on the whole graph. Actually, equation (5.68) indicates the scheduling of GBP: it requires messages to be computed bottom up in the region-graph. Thus, the best choice is to compute messages generation after generation, with a parallelization inside each generation, beginning by the bottom of $\mathcal{R}$. In this case, parallelization makes sense as we obtain same results as original GBP.

§ 5.5 Region-graph construction

Constructing a loop free region-graph is not a trivial task. Nothing makes sure that a loopfree region-graph exists for a given Tanner graph[7]. To our knowledge, there is no general method that provides the region-graph with the optimal GBP algorithm for a Tanner graph because there is no criterion that helps discriminate them.

5.5.1 Systematic construction

In [J.S05] is explained a systematic process that provides a region-graph for any Tanner graph, without using any criterion. Each cluster is made of a single check node accompanied by its neighborhood[8], someway as in Bethe approximation. Regions of next generations are constructed as explained before, searching for common children. The advantage of this construction is that it is systematic, *i.e.* we may use it for any LDPC code. The region-graph in Fig.5.21 is an example of this method applied on Hamming code.



(a) Tanner graph (b) Region-graph

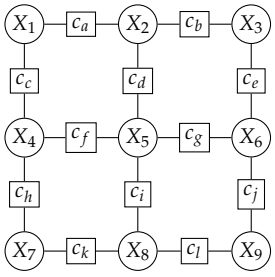
Figure 5.21: *A systematic region-graph for the Hamming code*

In spite of the ease of this construction, it does not offer a very relevant choice for the GBP algorithm according to the goal of the region-based approximation. Previously, we mentioned that it is aimed at absorbing harmful topological structures of the Tanner graph. Therefore, its construction has to be based on the shape of the Tanner graph, which is not the case here.

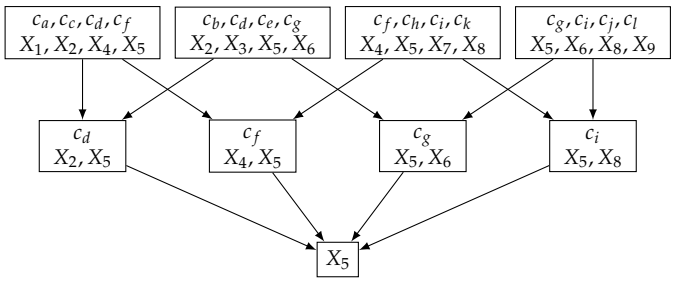
[7] except if we select the region-graph made of a unique region being the Tanner graph itself
[8] we recall that a region must be a subgraph of the Tanner graph

5.5.2 Square lattice

The square lattice Tanner graph in Fig.5.22(a) has the great advantage to be topologically regular. It is a pavement made with a constant square basis, then it is very easy to build such a code, even for any other length N. But squares are also the shortest loops, *i.e.* they are the most harmful topological structures for BP decoding. They accordingly appear to be natural choices for clusters, see Fig.5.22(b).



(a) Tanner graph



(b) Region-graph

Figure 5.22: *Suited region-graph for the square lattice code $N = 9$*

Absorbing short loops offers a better correction capability, even if the resulting region-graph is not loopfree[9]. We display in Fig.5.23 BER performance, representing pure GBP and uniformly damped GBP compared with the BP algorithm. We observe that even without blending update and memory, the GBP algorithm is more efficient than BP. As an example, at SNR= 5dB, the BER of BP is 2.10^{-2}, whereas undamped GBP reaches a BER lower than 2.10^{-3} and damped GBP has a BER lower than 2.10^{-4}: there is ten times less errors between them, which is a significant result.

One would suggest to use concatenations of squares as basic clusters but these clusters would be so large that computation complexity would skyrocket, which is not acceptable. Smallest squares succeed in fairly weighting good performance and competitive speed.

[9]short loops are more harmful than large ones as the iterative aspect of BP brings out their effect in very few iterations

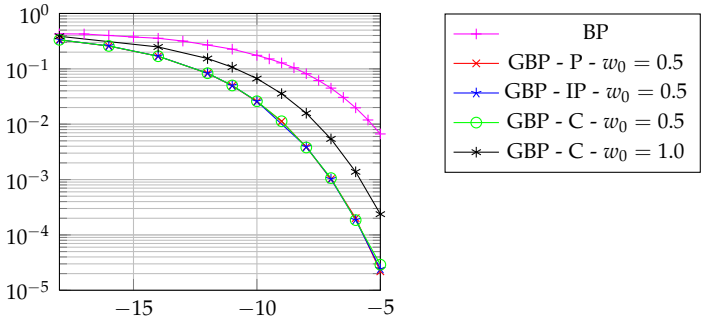


Figure 5.23: *Bit-error rate of the BP and the GBP algorithms on the square lattice code $N = 36$*

5.5.3 Trapping-sets

For most LDPC codes, the Tanner graph topology is not as simple as the square lattice one. It is then close to impossible to construct associated relevant region-graphs[10]. In contrast, in spite of our lack of knowledge on codes topology, significant advances have been made to bring out specific subgraphs, that are trapping sets, responsible for BP failures, see chapter 4 for more details. GBP may be consequently useful provided that region-graphs are constructed taking them into account.

Each code needs a long and deep study to extract a relevant region-graph with a GBP algorithm better than its BP equivalent, the set of trapping sets that compose a given Tanner graph being not always available. In this thesis, we focus on a specific code, the Tanner code of length $N = 155$ with check nodes and variable nodes degrees $d_c = 5$ and $d_v = 3$, respectively. The reason to choose it is that its Tanner graph is entirely covered by a set of 155 trapping sets TS(5, 3), see Fig.5.24, that is available[11]. Therefore, clusters that build the first generation of the region-graph are naturally selected.

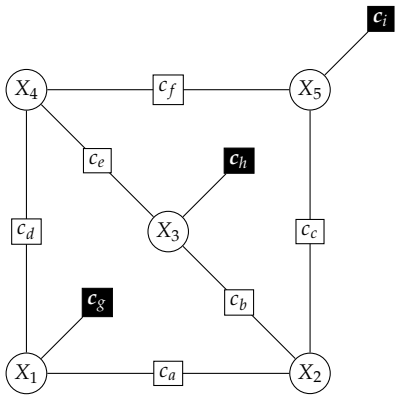


Figure 5.24: *Trapping set TS(5, 3)*

[10] notion of relevancy is hard to define, goal being to absorb harmful topological structures
[11] the fact that N is also the number of TS(5, 3) is not useful in our study

The best way to neutralize TS(5, 3) is to absorb each trapping set in a cluster. Problem is that it is absolutely not practical. We recall a region-graph construction rule that any check node $c_a \in \mathbf{C}$, included in an arbitrary region r, has to be accompanied by its neighborhood $\mathbf{X}_a \subset \mathbf{X}$. Counting the number of neighbors of all check nodes inside a TS(5, 3), we obtain that such a cluster would contain 9 check nodes and 35 variable nodes. According to the summation in equation (5.68), any message from a cluster would then need at least 2^{35} elementary operations, which is not acceptable. Eventually, trapping sets need splitting to create clusters of practical size.

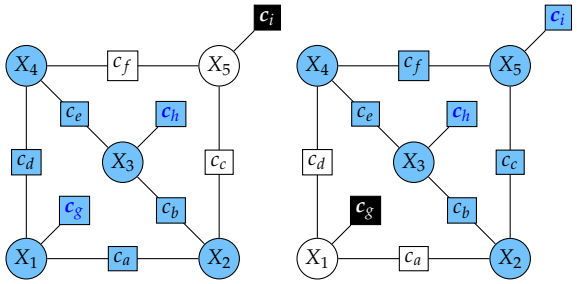
5.5.3.1 Local optimality

As it is very hard to ensure optimality on the whole region-graph, we orient a clustering by considering local optimality. This way, we introduce a basic rule for a region-graph construction.

Local loopfree principle: *the local region-graph, made of clusters and their children, created splitting a harmful topological structure, has to be loopfree such that the associated local RBA is exact.*

Following this principle, we perform a clustering such that local region-graphs made by splitting trapping sets are loopfree. Unfortunately, when considering all trapping sets, the whole region-graph is not loopfree anymore[12], then RBA remains an approximation. If we do not respect the local loopfree principle, RBA, and so GBP, are likely to provide very bad performance. For a given TS(5, 3), we show as examples two different splittings so that a cluster may be:

- either the subgraph made by one of the three shortest loops with two unverified connected check nodes, see Fig.5.25 where concerned nodes are colored (cluster is called *quadruplet*):
  - variable nodes X_1, X_2, X_3, X_4 and check nodes $c_a, c_b, c_d, c_e, c_g, c_h$,
  - variable nodes X_2, X_3, X_4, X_5 and check nodes $c_b, c_c, c_e, c_f, c_h, c_i$,
  - variable nodes X_1, X_2, X_4, X_5 and check nodes $c_a, c_c, c_d, c_f, c_g, c_i$,

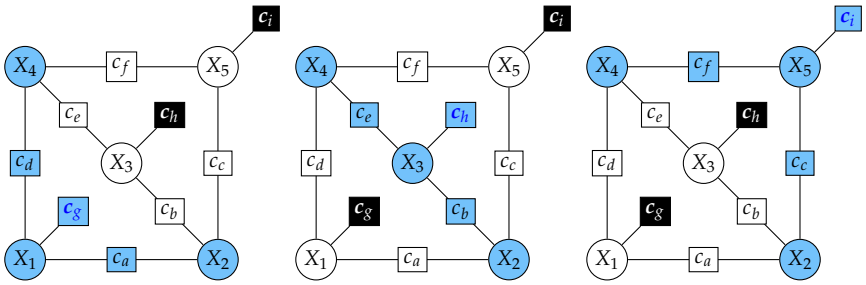


(a) First cluster (b) Second cluster

Figure 5.25: *Split of the TS(5, 3) – Quadruplets construction*

[12] trapping sets TS(5, 3) are strongly connected in Tanner code

- either the subgraph made by a triplet of variable nodes and one unverified connected check nodes, see Fig.5.26 where concerned nodes are colored in blue (cluster is called *triplet*):
  - variable nodes X_1, X_2, X_4 and check nodes c_a, c_d, c_g,
  - variable nodes X_2, X_3, X_4 and check nodes c_b, c_e, c_h,
  - variable nodes X_2, X_4, X_5 and check nodes c_c, c_f, c_i.



(a) First cluster (b) Second cluster (c) Third cluster

Figure 5.26: *Split of the TS(5, 3) – Triplets construction*

Splitting a trapping set in quadruplets fulfills the local loopfree principle, see Fig.5.27. However, there is no balance between all the variable nodes as a TS(5, 3) contains three quadruplets and not only two, see the third cluster in Fig.5.28.

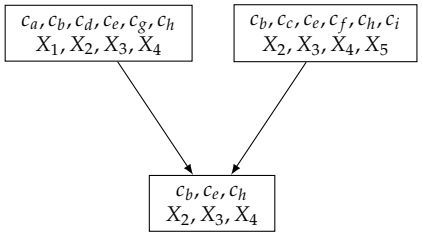


Figure 5.27: *Region-graph resulting from the split of a TS(5, 3) in two quadruplets*

The whole region-graph is not balanced too, but between all TS(5, 3). As a matter of fact, all variables nodes equally participate to several trapping sets[13]. The split in two quadruplets then involves that some variable nodes are included in more clusters than other variable nodes. This makes the GBP algorithm biased in the sense that noise realizations, and then likelihoods, are not fairly introduced to initialize the region-graph. More precisely, variable nodes that appear in a lot of clusters would wield more influence than variable nodes included in less clusters, whereas nothing either in the Tanner code or the channel discriminates between them. Finally, it turns out to be impossible to determine any criterion to select either the first and the second clusters, or the first and the third clusters, or the second and the third clusters.

[13]this property is inherent to Tanner code

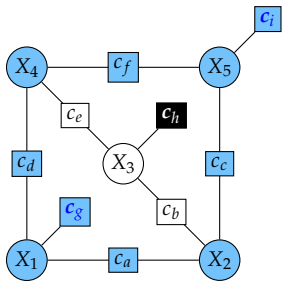


Figure 5.28: *Third quadruplet as a cluster*

Eventually, we should consider the three quadruplets to build the local region-graph. However, this would violate the local loopfree principle, as the region-graph is not loopfree, see Fig.5.29. Furthermore, quadruplets contain 24 variable nodes that is quite too large to consider a GBP algorithm of reasonable complexity, as a message from a cluster would need $2^{24} = 16777216$ elementary operations.

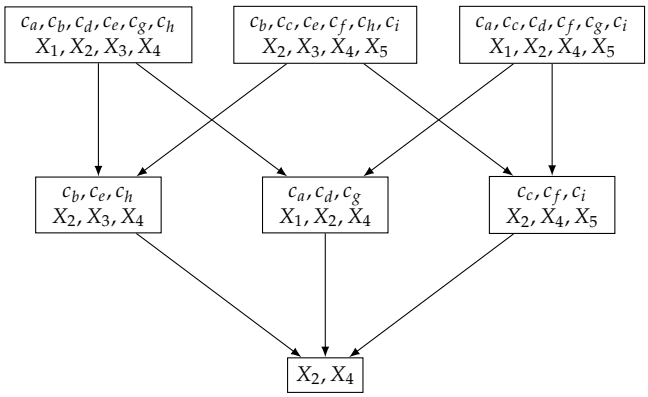


Figure 5.29: *Region-graph resulting from the split of a TS*$(5, 3)$ *in quadruplets*

The second split of trapping sets in triplets results in local region-graphs, see Fig.5.30, that satisfied the local loopfree principle. This way, RBA is locally exact, *i.e.* the GBP algorithm provides local optimal performance. In addition, triplets only contain 13 variable nodes, a message would then need $2^{13} = 8192$ elementary operations, that is considerably less than quadruplets.

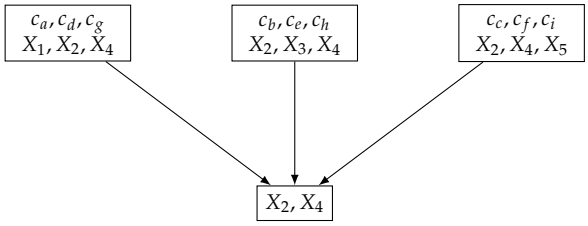


Figure 5.30: *Region-graph resulting from the split of a TS*$(5, 3)$ *in triplets*

Embedded inside the whole Tanner graph, any TS(5, 3) has many other connections than the edges exhibited in Fig.5.24. The complete trapping-set in this case is Fig.5.31. Empty circles in this figure are outer variable nodes, supposed to make the parity-check equations satisfied without assignment to the a inner variable nodes. Construction of the region-graph then results in a loopy Bayesian network, whose first generation is made of the clusters that we have created, the triplets.

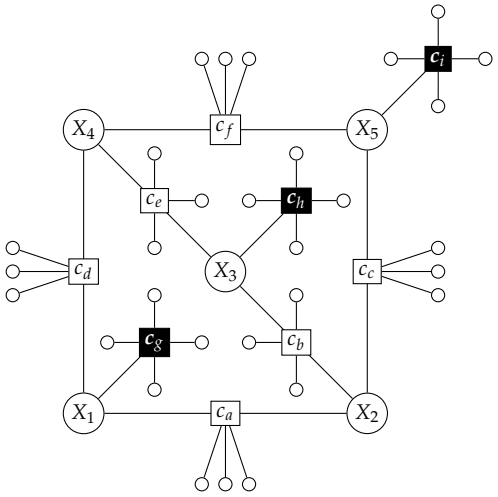


Figure 5.31: *A TS*(5, 3) *with all variable nodes connected to its check nodes*

A noteworthy property of the triplets construction is that the two last generations are exactly generations of the Tanner graph:

- any of the 93 regions of the penultimate generation $\mathcal{R}_{L-1}$ is only made of one check node and its neighborhood of variable nodes[14], L being the number of generations (3 in the current case),
- any of the 155 regions of the last generation $\mathcal{R}_L$ is only made of a single variable node.

As a matter of fact, this particular point makes our constructions relevant according to the work made in [M. 04]. In this paper, Welling assumed that a relevant region-graph, *i.e.* a region-graph whose GBP algorithm performs well, has to be constructed "above" the corresponding Tanner graph[15]: bottom generation $\mathcal{R}_3$ is made of regions each containing a single variable node such that all variable nodes are represented, any region of the upper generation $\mathcal{R}_2$ is made of a check node in such a way that all check nodes are represented, and regions of the upper generation $\mathcal{R}_1$, called *super nodes*, are clusters that would allow us to break or absorb harmful topological structures of the two last generations. Our triplets construction satisfies the Welling assumption as we break inner loops of trapping sets.

[14] we recall that a check node has to be accompanied by its neighborhood, as a region is a subgraph of the Tanner graph

[15] or more generally, factor graph

The Welling assumption is perfectly in agreement with our local loopfree principle provided that super nodes are added to Tanner graphs, and not to region-graphs. As an example, one would note that in the quadruplets construction, in Fig.5.29, children of quadruplets are triplets, *i.e.* quadruplets are super nodes of triplets. We recall that a balanced region-graph made with quadruplets contains loops, that means that adding super nodes to the region-graph in triplets damage its local optimality. Finally, the Welling assumption is relevant for Tanner graphs only.

5.5.4 Experiments

Considering that quadruplets construction implies a very large number of elementary operations, we decide not to hold it for computation time's sake. Experiments we have performed accordingly only concerned triplets construction. All previous results concerning the damped GBP, the iteration index for the scheduling were realized on Tanner code with triplets construction. We now present other comparisons between BP and GBP. Usually, decoders performance are delivered exhibiting the averaged BER as a function of the SNR. Our own experiments, not displayed here, showed that BER of BP and GBP were merged for low and middle values of the SNR on AWGNC[16]. This is an encouraging result as it confirms that the special construction of the region-graph with triplets does not damage the BER, in average[17]. As the triplets construction is based on trapping sets, and as trapping sets are known to damage BP decoding for weakly noisy channels, performance of GBP will distinguish themselves from BP ones for high SNR values on AWGNC and low p values on BSC.

As a channel noise is lowered, the number of simulations needed to get a trustworthy BER is increased. We previously mentioned that GBP suffers from a large computation complexity compared with BP one, which prevents us from computing any average BER in this case. Instead of considering random noise realizations, we then orient experiments on channel realizations that are deeply linked with trapping sets.

5.5.4.1 BSC

We deal with low-weight error events[18] on BSC. We only take into account specific low-weight error events that prevent BP from correctly decoding because of TS(5, 3). To keep a reasonable amount of results, we select a single error event, EE, that fairly represents all error events that deals with TS(5, 3). We roll out an experimental protocol for a given crossover probability p:

- for any variable node X_i that is not in EE, we force the channel to copy the input: $y_i = x_i$,
- for any variable node X_i that is in EE, we force the channel to flip the input: $y_i = \bar{x}_i$.

[16] and of the crossover probability p on BSC

[17] local loopfree principle is an assumption without proof, then experiments are the only means to check it

[18] a low-weight error event is a codeword that BSC has slightly disturbed, *i.e.* very few bits have been flipped

EE is made with 7 errors, a low value compared with the codewords length $N = 155$. This makes our protocol realistic as getting 7 corrupted bits is much more encountered for low p values than for large ones.

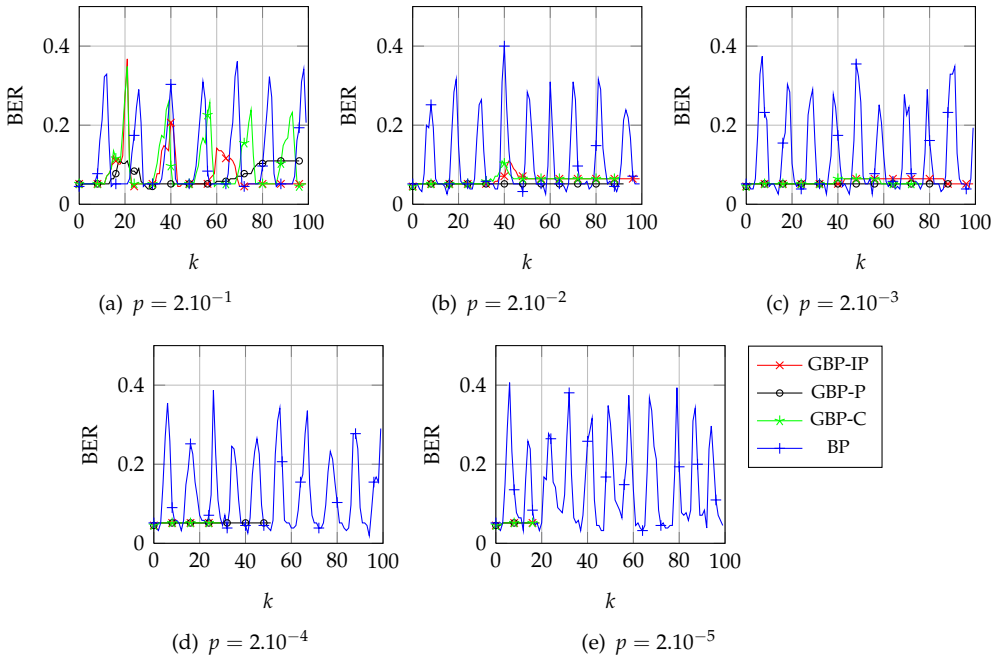


Figure 5.32: *BER on the Tanner code for EE completely dependent on TS*$(5, 3)$

On Fig.5.32 are represented iterative evolutions of BP and damped GBP BERs, with different damping factors[19] and decreasing values of p. We denote by GBP-C, GBP-P and GBP-IP the GBP algorithm with constant, parabolic and inverse parabolic damping factors, respectively. For any p, the BP BER oscillates within interval $[0.03; 0.4]$, whereas GBPs present different evolutions. First of all, for $p = 0.2$, GBP BERs do not oscillate as strongly as BP except for GBP-C which local maximum though decreases along k contrary to BP: local maximum values are $0.34, 0.26, 0.25, 0.24, 0.23$ for $k = 21, 39, 57, 75, 93$, respectively. These pseudo-oscillations may be hoped to end to a very low BER, but experiments, not presented here, showed that it is not true, BER of GBP-C always oscillates.

GBP-IP seems to behave in the same way, but as for GBP-P, the decreasing law of w_k makes the algorithm converge to a single value. This helps local maximum values be dramatically reduced compared with GBP-C: we obtain $0.38, 0.21, 0.14$ for $k = 21, 40, 60$. GBP-P exhibits a small bound around $k = 20$, as other decoders, but with a lower value 0.1096774. Then during thirty iterations it keeps a constant value unfortunately increased around $k = 70$. This phenomenon emphasizes that our triplets construction is dedicated to situations where p is significantly low. If p is too high then EE is not realistic, and the experimental protocol too.

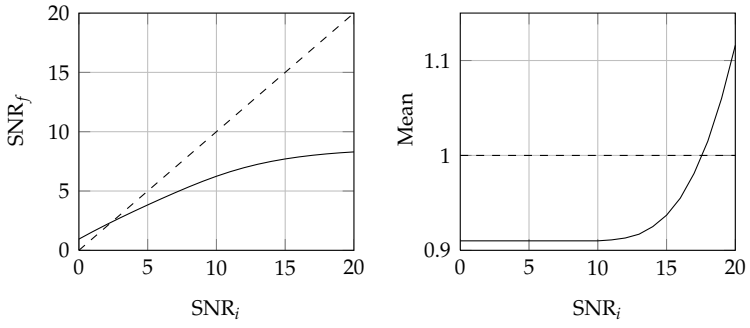
[19] $w_0 = 0.675$ for all of them

As p is getting smaller, behaviors of GBPs merge into a single behavior. All of them converge faster and faster to the same steady state as p decreases[20]. For $p = 2.10^{-2}$ and $p = 2.10^{-3}$, we observe bounds of no significance for GBP-C and GBP-IP around $k = 40$, as they disappear for lower values of p. For $p = 2.10^{-5}$, no more than 20 iterations are needed for all GBPs to converge, and BERs reach a value of 0.04, which corresponds to 8 errors out of 155 bits. We may interpret this result telling that GBPs did not succeed in obtaining as many corrected bits as BP could do[21], in addition it has added an error to EE. From another point of view, when comparing both iterative evolutions, we easily see that reaching 0.03 is close to a random process: BP oscillations are not perfectly predicted, then stopping the decoder to an exact local minimum makes the experimenter very lucky. On the contrary, all GBPs reach a unique stable and convergent evolution which allows us to ensure these decoders output, GBP is more reliable and trustworthy than BP.

Finally, we may assume that GBP is able to decrease TS(5, 3) effects, which makes it able to decrease LDPC error floor on BSC 4.8.6. We cannot verify this assumption as computation complexity of GBP is too high to simulate it for very low p values, but the experimental protocol is a relevant substitute.

5.5.4.2 AWGNC

The experimental protocol on AWGNC is similar to the previous one, but the error event notion has to be redefined as it is originally defined on BSC. As a matter of fact, flipping bits does not make sense on AWGNC as noise is a continuous elements independently added on all bits. The concept of error event on this channel is treated using bits corruption: a bit X_i of value x_i is corrupted if likelihood, $p_i(y_i|x_i)$, deduced from channel observation, y_i, is maximized for $\bar{x}_i$. A low-weight error event EE is then defined as a codeword that AWGNC has slightly disturbed, *i.e.* only very few bits have been corrupted.

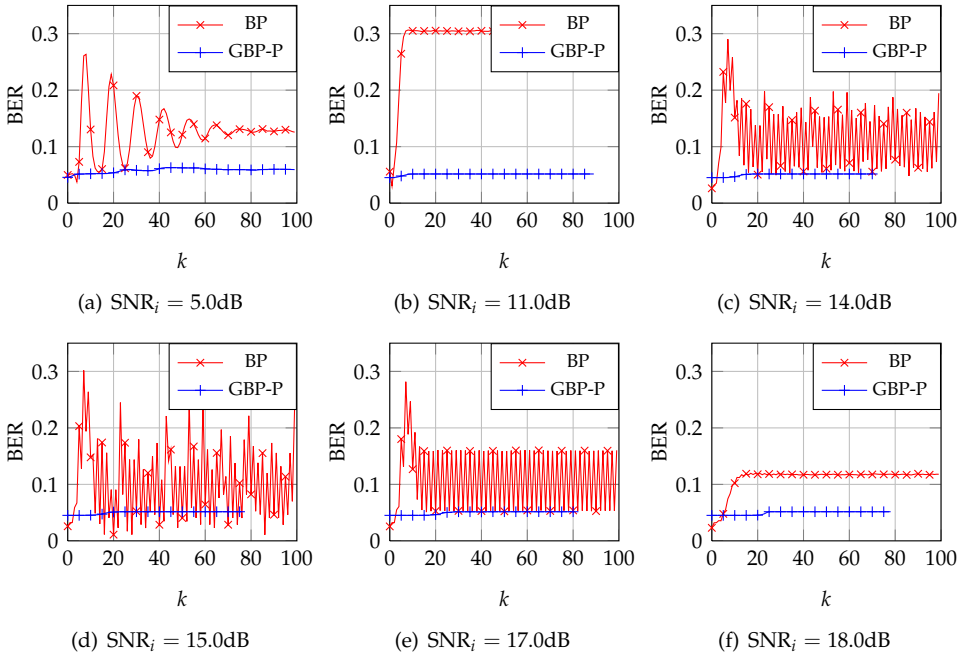


(a) SNR changes (b) Mean changes

Figure 5.33: *Modification of moments – Dashed lines are original SNR and mean, solid lines are their practical estimates according to EE*

[20] any decoder is stopped as soon as it has converged
[21] for BP, a BER of 0.03

Practically, on BSC, we forced a bit to be erroneous flipping it if channel did not. On AWGNC, we make a bit X_i corrupted multiplying the channel output y_i by -1: multiplying y_i by -1 is equivalent to swapping likelihoods $l_i(0, y_i)$ and $l_i(1, y_i)$[22]. As SNR is increased any error event is then more and more emphasized. The drawback of this protocol is that moments, as mean and variance, of random variable $\mathbf{Y}|\mathbf{X}$ are modified, as depicted in Fig.5.33. This is due to the fact that random realizations are discriminated to only get specific error events[23]. However, as the same protocol is used for BP and GBP, comparing between both decoders still holds, we only have to keep in mind that SNR values on next figures are not SNR_f but SNR_i, *i.e.* not estimated ones but original ones. We display iterative evolutions of the BER of BP and GBP-P in Fig.5.34[24]. We specify that GBP-IP and GBP-C are not represented in this figures as they behave in the same way that GBP-P, which means that results stem from the triplets clustering, not from damping.



(a) $\text{SNR}_i = 5.0\text{dB}$ (b) $\text{SNR}_i = 11.0\text{dB}$ (c) $\text{SNR}_i = 14.0\text{dB}$

(d) $\text{SNR}_i = 15.0\text{dB}$ (e) $\text{SNR}_i = 17.0\text{dB}$ (f) $\text{SNR}_i = 18.0\text{dB}$

Figure 5.34: *BER of the BP and the GBP algorithms on a EE with an AWGNC*

For low values of SNR_i, the BP BER pseudo-oscillates up to a steady state of 0.13 at 5dB, that is a substantial value compared with GBP one that quickly reaches 0.06 at 5dB. As SNR_i is decreased, the BER of BP stops oscillating by converging in less than 20 iterations to a non-negligible increasing steady value of 0.305 at 11dB, whereas the GBP BER converges to a dramatically lower value of 5.10^{-2} at 11dB *i.e.* about six times less errors than BP.

[22] on BSC, forcing a bit flipping is also swapping likelihoods

[23] but we obtain a subset of these genuine random realizations, *i.e.* they are still Gaussian variables

[24] BER is averaged over 10^4 simulations, which is enough to obtain reliable results that reflect behaviors of the decoders with our protocol

For $\text{SNR}_i \geq 13$dB, BP does not converge anymore: it either oscillates, see at 14dB and 17dB, or it exhibits an erratic evolution, see at 15dB[25]. We see that the range of values of the BER is large enough such that the choice of the last practical iteration widely influences the performance of the algorithm. On the contrary, GBP still convergences in nearly 30 iterations to the same value of 5.10^{-2} for any SNR_i value. Eventually, for $\text{SNR}_i \geq 18$dB, the BP BER reaches steady and decreasing values. For very high SNR_i, it is stuck to the same non-zero value as GBP around 3.10^{-2}. We summarize the steady values of the BER of both algorithms in Fig.5.35.

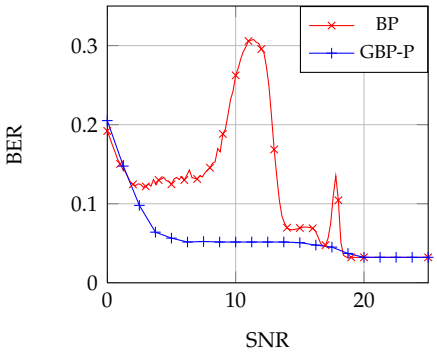


Figure 5.35: *Average BER of the BP and the GBP-P algorithms*

We observe that the triplets clustering provides a robust GBP algorithm against TS(5, 3) compared with BP. For very high values of SNR_i, likelihood of any variable node X_i of EE is almost:

$$l_i(0, y_i) \approx 0, \quad l_i(1, y_i) \approx 1 \tag{5.69}$$

therefore the corresponding bit is too corrupted to be correctly decoded by either GBP or BP. That is why for both decoders converge to a non-zero value of the BER, *e.g.* at $\text{SNR}_i = 20$dB, we obtain a BER of 3.10^{-2} for the two algorithms. As EE is of weight 7, such a value of the BER is equivalent to 4 errors, we conclude that BP and GBPs can only correct 3 errors.

§ 5.6 Conclusion

The Bethe approximation is a particular case of the more general region-based approximation, as the BP algorithm is the GBP algorithm when the region-graph is exactly made of the Tanner graph[26]. We observed that generalizing the BP message-passing needs a generalization of the node definition, to create region-graphs which edges are support for the GBP algorithm. In addition, new damping rules for GBP help converge, as its original version is unstable, and obtain better results in terms of BER.

[25] we will deal with this peculiar phenomenon in next chapter
[26] clusters are only check nodes and their neighborhood

We saw that contrary to the Tanner graph, we can find a multitude of region-graphs for any LDPC code, since the clustering is not unique. We then introduced the local loopfree principle and an associate specific construction based on particular clusters, the triplets, stemming from the split of trapping sets. GBP is then aimed at dealing with error events deeply connected with trapping sets. Exhibited simulation results, according to a dedicated protocol, showed that the triplets clustering provided a GBP algorithm more accurate than BP in these cases. These results tend to investigate more on the GBP algorithm, and particularly the construction of the region-graph. This would help bring out other clustering rules and new principles to better understand GBP and RBA. In addition, the study of the damping factor is not closed, we have raised an open problem.

The specific construction in triplets is a technique that can be used as a basis to cluster any LDPC code, provided that known and harmful topological structures entirely cover its Tanner graph. In the literature, clustering the square grid code using squares follows the same idea, and we also observed that GBP outperformed BP in this case. To our knowledge, no other clustering has been proposed on any other code, therefore our construction is likely a new lead in the research field on graphical models.

- Chapter 6 -

Dynamical behavior of iterative decoders

§ 6.1 Introduction

Messages $m^{(k)}$ of BP and the GBP are computed at any iteration k, according to update rules (1.12), (1.13), and (5.52). These rules are functions:

- all messages at the previous iteration $\{m_e^{(k)}\}_{e\in E}$, where E is the set of edges either of a Tanner graph or a region-graph,
- initialization parameters: SNR and likelihoods.

It is accordingly legitimate to consider update rules of these algorithms as *iterated maps*. This way, it is possible to describe BP and GBP with tools of dynamical systems theory. It allows us to introduce explanations of non-trivial phenomenon, *e.g.* non monotonous evolution of the BP convergence rate exhibited in 1.7. The goal of this chapter is to extract a few typical behaviors of BP and GBP and to examine them as functions of SNR. More specifically we want to emphasize:

- the variety of behaviors encountered with these decoders,
- how these behaviors wield influence on decoding performance.

First of all, this work needs the definition of a mathematical framework, known as the *state space*. After that, we will present dynamical quantifiers to describe the decoders. Among the large set of estimators commonly encountered in the literature, we chose a few of them to characterize BP and GBP dynamics. This investigation is aimed at highlighting particular SNR values for which their behaviors are non-trivial, describing the nature of these behaviors and their connections with decoding performance. This would help understand and predict, or even control, decoders behaviors. All experiments in this chapter are performed on Tanner code, to specify dynamics of GBP coupled with the triplets construction. In addition, from now on, the channel used is only the AWGNC.

A part of the following results on BP are published in [J.-12b],[J.-12d],[J.-13b], on GBP in [J.-12c],[J.-12e].

§ 6.2 Preliminaries

BP and GBP are dynamical systems that sometimes behave very differently when initialization parameters are changed. Even though iterated maps that update messages are deterministic, we observe significant dependence on likelihoods, see Fig.6.1.

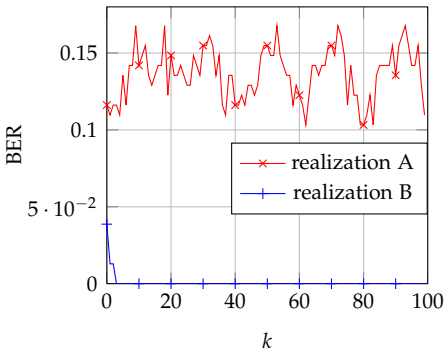


Figure 6.1: *Iterative BER of BP with two different sets of likelihoods at the same SNR*

In this figure are depicted iterative evolutions of the BP BER for two channel realizations at the same SNR. It clearly appears two behaviors: realization B makes the BER converge to a zero value in 5 iterations whereas realization A prevents the BER from converging, it oscillates. The space created by likelihoods is accordingly not uniform: two likelihoods computed at the same SNR do not systematically lead to the same decoding.

We represent in Fig.6.2 the space generated by likelihoods of two variable nodes ($X_i = 100$, $X_j = 102$), randomly selected in Tanner code. For a given channel realization at SNR = 2.3dB, we quench all likelihoods of other variable nodes and we make vary likelihoods $l_i(0, y_i)$ and $l_j(0, y_j)$ from their minimum value, zero, to their maximum value, one. We then run a BP decoding for each couple of values $\{l_i(0, y_i) \in [0; 1], l_j(0, y_j) \in [0; 1]\}$ and we pick out the BER to plot it in the likelihoods space. We observe that two neighboring couples can reach very different BER, depending on their relative position in the space. As an example, points A and B are defined such that variable nodes X_i and X_j are:

$$\begin{aligned} l_i^A(0, y_i) &= 0.48, \quad l_j^A(0, y_j) = 0.89, \\ l_i^B(0, y_i) &= 0.87, \quad l_j^B(0, y_j) = 0.54. \end{aligned}$$

Computing BER variances σ_A^2, σ_B^2 in disks of radius 0.1, centered on A and B, respectively, offers that:

$$\begin{aligned} \sigma_A^2 &= 8.10^{-6}, \\ \sigma_B^2 &= 3.10^{-4}. \end{aligned}$$

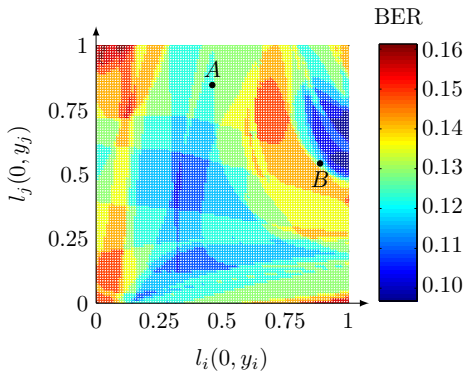


Figure 6.2: *Likelihoods space at SNR* $= 2.3dB$ *on the Tanner code for quenched likelihoods except on variable nodes* $X_i = 100$, $X_j = 102$ *– BER of the BP*

Ratio between both variances is around 10^2, *i.e.* BER may change ten times more around point B than around point A, which is a considerable difference. One would link this result to the specific couple ($X_i = 100$, $X_j = 102$) but it turns out possible to observe such a behavior for other couples and noise realizations[1]. It then appears that message-passing algorithms are strongly dependent on initialization, *i.e.* when slight changes are induced.

The likelihoods space provides a *BER map* for a given couple of decoder and LDPC code. Such a map would give access to the whole description of the error correction capability[2]. Unfortunately, to compute a complete map is needed not only two likelihoods but N likelihoods, N being the number of variable nodes. Even for low values of N[3], computing the BER map is then impossible, as it would need $\mathcal{A}^N$ configurations with $\mathcal{A}$ the number of values to assign values to a single likelihood. Furthermore, storing this map would ask for unrealistic hardware requirements. Using less informative but tractable estimators finally turns out necessary to understand decoders behaviors.

§ 6.3 State space

The study of any iterated map needs state space definition, *i.e.* a mathematical space where state variables evolve. Usually, state variables are defined as variables handled by the iterated map, *e.g.* the logistic map transforms $x^{(k)}$ in $x^{(k+1)}$ by function $f(x) = ax(1-x)$ therefore $x^{(k)}$ is the state variable. In the same way, update rules of decoding algorithms may be summarized as iterate maps according to the following rules:

$$\text{BP}: \quad \forall (X_i, c_a)\text{an edge}, \quad m_{ai}^{(k)} = F_{BP}(\{m_{bj}^{(k-1)}\}_{b,j}, \{l_j\}_j), \tag{6.1}$$

$$\text{GBP}: \quad \forall (r,s)\text{an edge}, \quad m_{rs}^{(k)} = F_{GBP}(\{m_{uv}^{(k)}\}_{u,v}, \{m_{uv}^{(k-1)}\}_{u,v}, \{l_j\}_j), \tag{6.2}$$

[1] exhibiting all simulation would not be reasonable to keep a fair paper length
[2] whether it is the code or the decoder
[3] Tanner code of length $N = 155$ is considered in coding theory as a small code

where update rules F_{BP}, F_{GBP} are provided in previous chapters by equations (1.12),(1.13) and (5.52). F_{BP} maps exactly messages from an iteration to the next one whereas F_{GBP} computes $k-th$ iterated messages by former and current messages. Furthermore, BP messages are two-dimensional functions while GBP messages are multi-dimensional. Comparing both algorithms involves that iterated maps and state variables used to describe them are comparable, which is not the case here. To confront both decoders to same situations, we need common quantities.

Beliefs on single variable nodes are common quantities between BP and GBP. We then choose them to stand for state variables. In the following, we consider that any belief of a node X_i refers only to $b_i(0, y_i)$[4]. Expanding explicit update rules that compute beliefs as functions of former iteration beliefs, either for BP or GBP, proves to be a hard task therefore we do not present it. Nevertheless, we assume that such functions G_{BP}, G_{GBP} exist, which allows us to define their related dynamical systems:

- the dynamical system related to BP is the state space built from beliefs on the N single variable nodes computed from G_{BP},
- the dynamical system related to GBP is the state space built from beliefs on the N single variable nodes computed from G_{GBP}.

We note that in the case of the GBP algorithm, the bottom generation $\mathcal{R}_L$ of the region-graph is not always made of single variable node regions, sometimes it does not even contain any single variable node region. For such cases, beliefs on missing single variable nodes are computed using the local marginalization on regions of the upper generation $\mathcal{R}_{L-1}$, or still upper if necessary. Without any loss of generality, we now refer to any decoder and its related dynamical system by the same denomination, in order to lighten writings[5].

We now have to choose behavior descriptors for decoders. BER is quite poor according to the quantity of information it offers as it firstly maps beliefs, *i.e.* continuous values in $[0;1]$, to estimated bits, *i.e.* discrete values in $\{0,1\}$, and then it associates an average value over these bits. As an example, considering an arbitrary bit X_i, an indecisive belief $b_i(0, y_i) = 0.49, b_i(1, y_i) = 0.51$ is mapped to $\hat{x}_i = 1$ in the same way as the belief $b_i(0, y_i) = 0.01, b_i(1, y_i) = 0.99$. It is a very rough estimate of the information contained in the belief. Using BER finally deletes a large part of information lying on graph edges, either Tanner graphs or region-graphs. The only advantage of BER is that it delivers any decoder performance with only one value, that makes it easy to store. In this investigation, searching for other descriptors appears absolutely necessary to obtain accurate information on decoders behaviors.

[4]for any variable node X_i, $b_i(0, y_i)$ is enough to describe the distribution of X_i as $b_i(1, y_i) = 1 - b_i(0, y_i)$

[5]strictly speaking, defining dynamical systems with beliefs is not correct as update rules G_{BP} and G_{GBP} are not bijective, *i.e.* many different beliefs vectors at iteration $k-1$ may produce the same beliefs vector at iteration k

§ 6.4 Mean square beliefs

6.4.1 Definition

The mean-square-belief (MSB) introduced in [X. 05] is an interesting alternative to BER, as it offers a good compromise: it is a one-dimensional map that reflects decoding performance using directly beliefs. We may compute it at any iteration k for a given channel realization as:

$$E(k) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(b_i^{(k)}(x_i)\right)^2} \tag{6.3}$$

where $\mathbf{x} = [x_1, \ldots, x_N]$ is the emitted codeword, *e.g.* the null codeword that we always use in our experiments. This function has the following properties:

- $E(k) = 1.00$: correct decoding,
- $E(k) = 0.25$: any variable node entropy is maximum, no information is available on the emitted codeword,
- $E(k) = 0.00$: faulty decoding.

This function implicitly depends on SNR and likelihoods. MSB provides smoother information about a given decoder than BER as it does not include any threshold. Hence, it is a more faithful estimator.

6.4.2 Experimental protocol

MSB reveals performance of a given decoder according to a specific noise realization. To cover a large range of SNR values, we firstly generate N noisy samples $\mathbf{n} = [n_1, \ldots, n_N]$, all computed from a Gaussian law $\mathcal{N}(0, 1)$. After that, these samples are scaled on the SNR multiplying them by the associated standard deviation[6]. This way, we obtain a sequence of channel realizations:

$$\mathbf{y}_1 = \mathbf{x}_1 + \sigma_1\mathbf{n} \tag{6.4}$$
$$\mathbf{y}_2 = \mathbf{x}_2 + \sigma_2\mathbf{n} \tag{6.5}$$
$$\mathbf{y}_3 = \mathbf{x}_3 + \sigma_3\mathbf{n} \tag{6.6}$$
$$\vdots \tag{6.7}$$

where $\mathbf{y}_j = [y_{j,1}, \ldots, y_{j,N}]$ and $\mathbf{x}_j = [x_{j,1}, \ldots, x_{j,N}]$. Samples $\mathbf{y}$ provide us likelihoods that initialize decoding algorithms. In order to obtain a consistent investigation, we use the same noise realizations for BP and GBP. We recall that the goal of this study is to focus on non-trivial decodings, *i.e.* decodings where the BER does not converge to any steady value[7]. To our knowledge, there is no results in the literature that help find such realizations.

[6] if X and Y are two random variables s.t. $Y = \sqrt{a}X$, then $\sigma_X = a\sigma_Y$

[7] a realization, the corresponding error event and the likelihoods are said *non-trivial* when decoding results in a non-trivial BER

The method we used in this thesis is to run the desired decoder for many simulations, each one with a different basic noise realization $\mathbf{n}$. Plotting MSB for all of them allows us to discriminate non-trivial error events. In the following of this chapter, we use a specific error event EE[8] to exhibit dynamical properties of the dynamical systems related to BP and GBP. This error event perfectly reflects all typical behaviors that we encountered with other error events in our experiments, not displayed here, *i.e.* it is a reliable error event to describe non-trivial evolutions of decoders.

6.4.2.1 Attractor

We now present essential definitions from [Hiloo]. First of all, the *trajectory* of a decoder is the sequence of points:

$$\mathbf{T} = [[b_1^{(1)} \dots b_N^{(1)}]^T, \dots, [b_1^{(k)} \dots b_N^{(k)}]^T] \tag{6.8}$$

of the related dynamical system in its state space as the iteration k is increased. In addition, an *attractor* is a set of points, inside the state space, to which the system, related to any decoder, approaches as the number of iterations goes to infinity. Moreover, the *basin of attraction* of a particular attractor is the set of initial conditions, *i.e.* likelihoods, that give rise to trajectories approaching the attractor as the number of iterations is increased to infinity.

The difficult task in the study of decoders dynamics is that the state space is so immense that we can list neither all attractors nor their basins of attraction. This is why scanning a large part of the state space is required to extract typical non-trivial error events.

6.4.3 Experiments

6.4.3.1 BP

We display in Fig.6.3 the iterative evolution of $E(k)$ as a function of the SNR, considering BP decoding on EE. Four typical behaviors may be distinguished, ordered in the table 6.1.

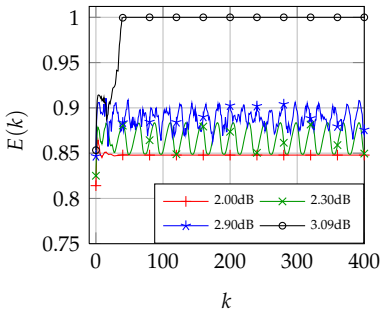


Figure 6.3: *BP MSB of EE: four SNR values, to each one is found a typical behavior*

[8] it is different from the error event EE of previous chapter

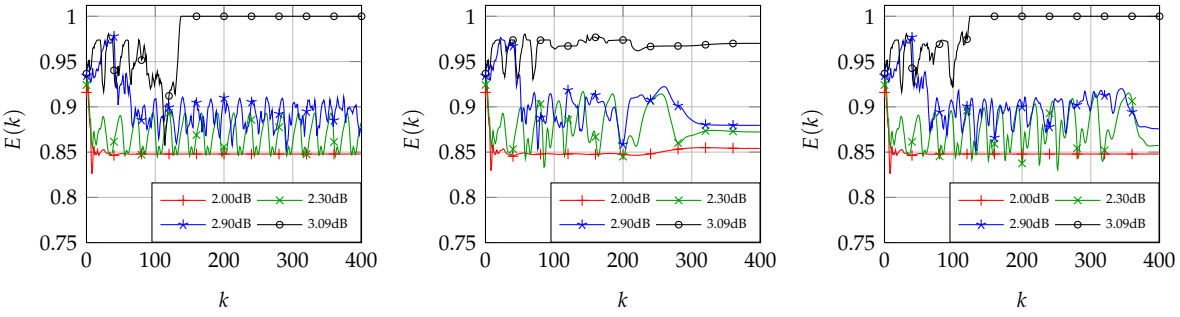
Behaviors of $E(k)$	**SNR**
convergence to a faulty decoding	2.00dB
oscillation	2.30dB
unknown evolution	2.90dB
convergence to the right decoding	3.09dB

Table 6.1: *BP behaviors in terms of SNR values*

These four behaviors are not specific to EE, we encountered them for most other non-trivial error events. Difference remains in the SNR values where the behavior changes. We note that continuously increasing SNR values does not make BP behavior change as continuously, but abruptly for specific SNR values. We call these SNR values *critical SNR values*. To study rough behavior changes, it is relevant to use a bifurcation diagram.

6.4.3.2 GBP

We display on Fig.6.4 the iterative evolution of $E(k)$ on EE as a function of the SNR with GBP decoding.



(a) GBP-C (b) GBP-P (c) GBP-IP

Figure 6.4: *GBP MSB on EE*

GBP-C and GBP-IP similarly behave to BP for the four critical SNR values. A slight difference remains in transient phases that last longer. Moreover, GBP-IP eventually converges[9]. In contrast, the GBP-P algorithm does not exhibit such a vicinity. We do not observe clear oscillations at SNR $= 2.30$dB, even though values of $E(k)$ are similar in average. In addition, it reaches an imperfect steady state at 3.09dB. However, as GBP-IP, it is due to converge in the last iterations by means of the damping factor.

More generally, we notice that GBP behaviors, either damped or not, can be categorized in the same way that BP behaviors, see 6.1. Therefore, we use the critical SNR values $\{2.00, 2.30, 2.90, 3.09\}$dB in the following of the study to describe decoders dynamics. The investigation is now oriented to highlight SNR intervals in which typical behaviors live.

[9]thanks to the decreasing damping factor

§ 6.5 Bifurcation diagram

6.5.1 Definition

Analyzing decoders behaviors for any error event and for any SNR value is a substantial work as it results in a tremendous amount of information. Computing the *bifurcation diagram* is a lead towards a summary of information. This descriptor is a one-dimensional function that provides the evolution of the MSB steady state as a function of the SNR, for any decoder. Strictly speaking, steady states are not systematically found, as MSB often oscillates, see previous figures as examples. We accordingly define the steady state, at an arbitrary SNR, as the MSB value at the last iteration K[10]. The goal is to bring out SNR values for which decoders radically change in their behavior. These changes are called *bifurcations* [K.T96].

6.5.2 Experiments

6.5.2.1 BP

Bifurcation diagram of BP for the selected error event EE is displayed in Fig.6.5. Five obvious SNR intervals may be emphasized in this figure, each one corresponding to a specific BP behavior. We summarize corresponding SNR intervals in table 6.2.

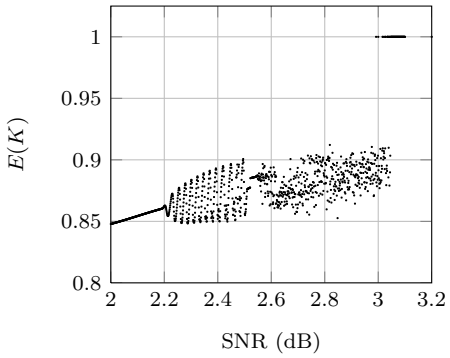


Figure 6.5: *Bifurcation diagram of BP*

Behavior of $E(K)$	**SNR**
smooth increasing	[0.00 dB; 2.20 dB]
pseudo-oscillations	[2.20 dB; 2.51 dB]
erratic evolutions	[2.51 dB; 2.99 dB]
convergence jumps	[2.99 dB; 3.04 dB]
optimal convergence	[3.04 dB; ∞ dB]

Table 6.2: *BP behaviors in terms of SNR intervals*

[10] arbitrary fixed by experimenters

Numerous other experiments, performed with other non-trivial error events, but not displayed here, made reveal that lengths and bounds of these intervals are quite constant. In addition, these behaviors are always ordered in the same way, which indicates a regularity in BP behaviors for Tanner code. When changing LDPC code, BP similarly behaves but with different lengths and bounds of the SNR values, *e.g.* McKay code of length $N = 504$ which bifurcation diagram is exhibited in Fig.6.6. Even though critical SNR values and amplitudes of $E(K)$ are different from Tanner code, the five intervals are clearly distinguishable.

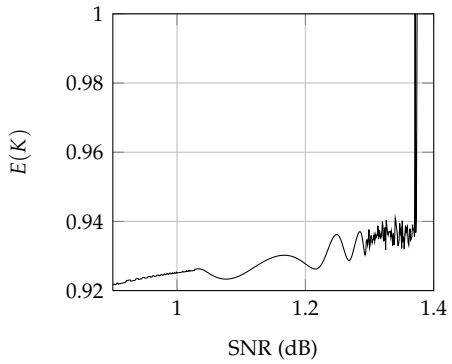
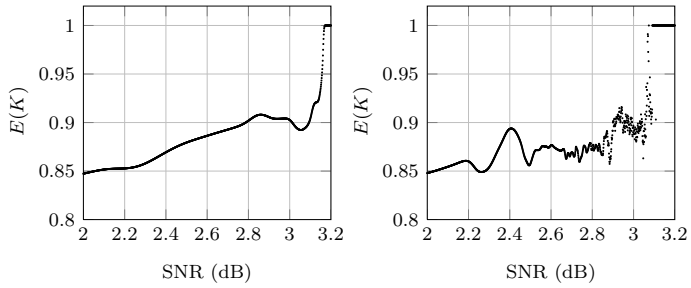


Figure 6.6: *Bifurcation diagram of the BP on the Mac Kay code*

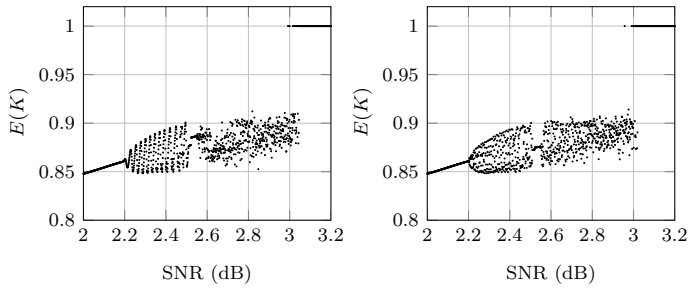
An important practical point is that the bifurcation diagram shape depends on the last iteration K[11]. We display in Fig.6.7 this dependence on K computing the bifurcation diagram, using Tanner code, for four values of K: $30, 100, 1000, 10000$. We observe that SNR intervals do not change, but increasing K makes their bounds emerge more significantly. For $K = 30$, it is hard to split SNR values into five intervals, $E(K)$ does not change abruptly, contrary to the cases in which $K \geq 100$. As K is increased, we may distinguish a Hopf bifurcation [Hiloo], at SNR=2.20dB, that brings out two steady states. In other words, increasing K makes BP closer to its steady behaviors, choosing low K values limit BP to its transient phases.

We recall that BP is practically run for less than one hundred iterations, *i.e.* the BP algorithm is not fairly used. As a matter of fact, the lack of convergence may often be solved considering larger K so that BP can reach a steady behavior. Within the state space, the BP dynamical system takes a very long time before falling into an attractor[12]. For $K = 100004$, the attractor is obvious as the fork of the Hopf bifurcation is clearly visible, even if transient points still appear between the two fork branches. In addition, an extra behavior arises, for $K \in \{1000, 10000\}$, in interval [2.50 dB;2.54 dB], where $E(K)$ stops oscillating but does not begins an erratic evolution, it seems to start up again a smooth increasing as for first interval. In spite of this noteworthy result, we shall not consider such an extra behavior, as selecting a large value of K takes us away from any practical interest. Thus, we use $K = 100$ to stay close to realistic situations, at the expense of the decoders long-term reliability.

[11] $E(K)$ is assumed to be the steady state
[12] for middle SNR values



(a) $K = 30$ (b) $K = 100$

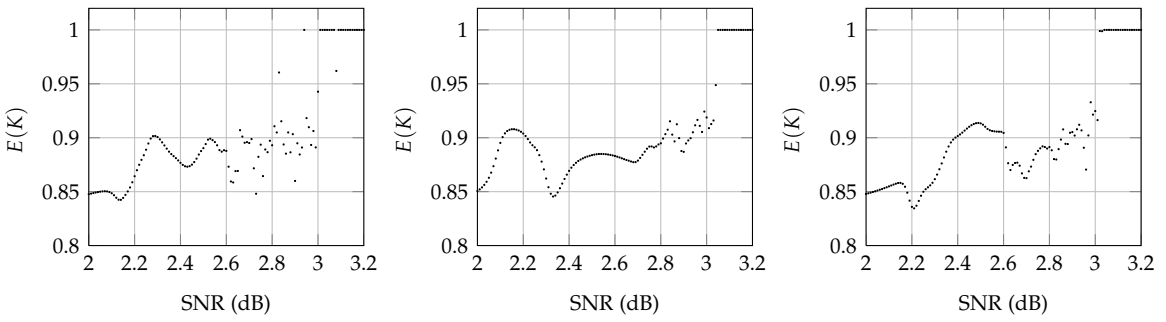


(c) $K = 1000$ (d) $K = 10000$

Figure 6.7: *Influence of K on the bifurcation diagram of the BP*

6.5.2.2 GBP

We present in Fig.6.8 the bifurcation diagrams of GBP and damped GBPs for $K = 100$.



(a) Constant damping – $K = 100$ (b) Parabolic damping – $K = 100$ (c) Inverse parabolic damping – $K = 100$

Figure 6.8: *Bifurcation diagram of GBPs*

Constant damping triggers a behavior similar to the BP one as $E(K)$ pseudo-oscillates before presenting erratic evolutions. GBP-C firstly differs from BP through the fact that a bifurcation distinctly occurs at SNR = 2.6dB whereas it is less visible for BP[13].

[13]for the same K value

Secondly, GBP-C presents convergence jumps from SNR = 2.87dB, and GBP-IP too, which is lower than for BP as it has to reach larger SNR values than 2.99dB. We note that GBP-P behavior is slightly different: pseudo-oscillations take place earlier than BP and other GBPs, no visible bifurcation occurs around 2.6dB and it erratically behaves only from 2.8dB, *i.e.* 0.2dB later than other ones.

We note that for SNR values in [2.6 dB; 2.87 dB], BP and damped GBPs with constant and inverse parabolic laws present peculiar evolutions of $E(K)$. The shape of bifurcation diagrams is close to the logistic map one: a cloud of points without particular consistency. It is known [Hiloo] that the logistic map trajectory falls into a chaotic attractor for $a \geq 3.44$, see Fig.6.9.

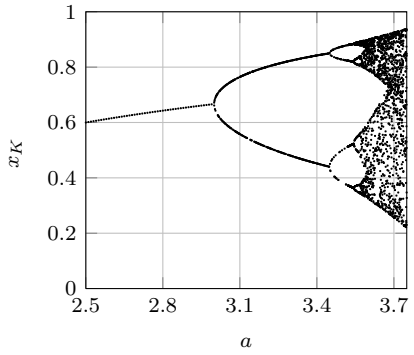


Figure 6.9: *Chaos in the logistic map* $x_{k+1} = ax_k(1 - x_k)$*, for* $a \geq 3.44$

We accordingly assume that the erratic evolution of $E(K)$ is chaos for both message-passing algorithms. Eventually, the bifurcation diagrams of GBPs show the same properties that we brought out for BP on the same error event EE. All of these decoders are similarly influenced by same likelihoods. In other words, the distinction between them is not easy to make, based on their bifurcation diagrams only[14].

The bifurcation diagram underlines critical SNR values and associated intervals in which decoders have particular behaviors. In addition, it helps us introduce assumptions concerning behaviors nature, that we now attempt to describe in depth.

§ 6.6 Reduced trajectory

Iterative BER and MSB are means to draw decoders evolutions according to SNR values. Visualizing bifurcations and peculiar behaviors allows us to conjecture nature of associated attractors that trap decoders. Usually, if state spaces are one, two or three-dimensional, nothing prevents us from observing trajectories evolution, *e.g.* logistic and Hénon map, Lorenz system lie in humanly observable state spaces of one, two and three dimensions, respectively.

[14] larger values of K would make the comparison easier but as pointed out earlier, it would not reflect a realistic point of view

In the current study, trajectories of decoders lie in N-dimensional state spaces, with an often very large N[15]. Visualizing decoders trajectories is then absolutely impossible. BER and MSB are useful to pre-investigate on decoders behaviors but they are too restrictive as they are average of N beliefs. Another method is therefore essential to observe and analyze decoders trajectories.

6.6.1 State space reduction

We introduce an alternative space which number of dimensions is considerably lower than for the original state space. We build it through the State Space Reconstruction (SSR), exposed in chapter 3.5. We recall that SSR helps faithfully recover a state space only with a subset of its state variables. We interpret this property such that a subset of all state variables is enough to fairly describe any dynamical system. In other words, SSR allows us to construct a space of an arbitrary number of dimensions m. This way, we are able to construct three-dimensional *reduced state spaces* such that dynamical systems related to decoders are now humanly observable.

Decoders state spaces are generated by beliefs assumed to be state variables. We cannot use any beliefs subset to construct a reduced space because we do not have any criterion to discriminate three beliefs among N. One would suggest to take into account only "very variant" beliefs but this would mean that other parameters (SNR, EE, LDPC code) would discriminate three other beliefs, making reduced state spaces hardly comparable.

Therefore, we have to find a way to gather all beliefs without any discrimination in three quantities. As a matter of fact, this is perfectly possible using MSB. We practically compute a sequence, called *time series*, of K points $[E(1) \dots E(K)]$ that we map to a matrix such that each column is a state variable of the reduced state space:

$$T = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \\ \vdots \\ \mathbf{T}_L \end{bmatrix} = \begin{bmatrix} E(1) & E(1+\tau) & \dots & E(1+(m-1)\tau) \\ E(2) & E(2+\tau) & \dots & E(2+(m-1)\tau) \\ \vdots & \vdots & & \vdots \\ E(n) & E(n+\tau) & \dots & E(n+(m-1)\tau) \\ \vdots & \vdots & & \vdots \\ E(L) & E(L+\tau) & \dots & E(L+(m-1)\tau) \end{bmatrix}$$

Vector $[\mathbf{T}_1 \dots \mathbf{T}_L]$ contains L points of a trajectory, called *reduced trajectory*, inside the reduced space. The value m is the target dimension of the reduced space, called the *embedding dimension*. Here, m must be three to get a humanly observable space. The term τ is a delay and L is chosen such that:

$$L + (m-1)\tau \leq K \tag{6.9}$$

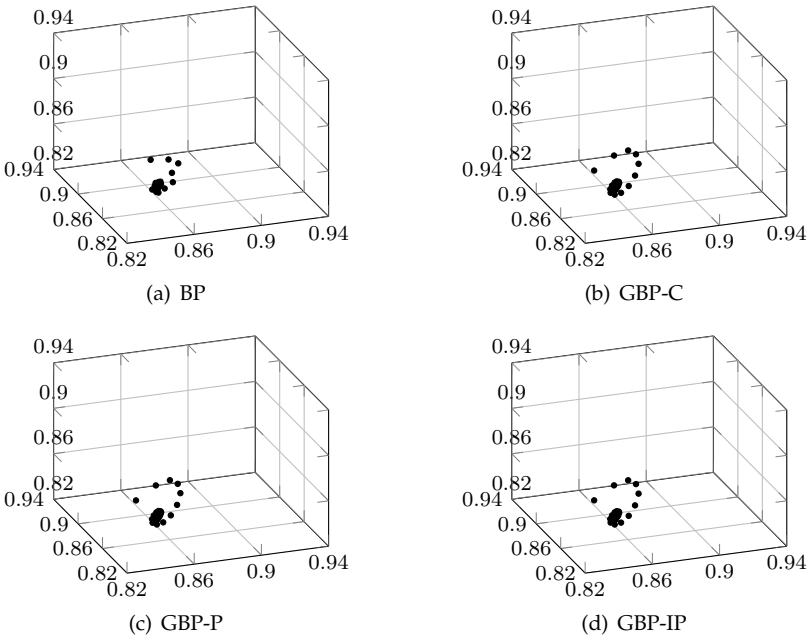
Choice of τ and m are known in the literature to condition the SSR success, research on this topic is still on progress [H. 06]. We set the embedding dimension $m = 3$, because we need a three-dimensional space. Practically, the number of points K is not very large, around a hundred, to keep realistic decoders whereas a faithful reduced trajectory needs a large number of points L.

[15]Tanner code of length $N = 155$ is considered as a small code in coding theory

We accordingly choose the delay τ as small as possible, *i.e.* $\tau = 1$. Eventually, we obtain the 3-dimensional trajectory of $K-2$ points:

$$T = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \\ \vdots \\ \mathbf{T}_{K-2} \end{bmatrix} = \begin{bmatrix} E(1) & E(2) & E(3) \\ E(2) & E(3) & E(4) \\ \vdots & \vdots & \vdots \\ E(n) & E(n+1) & E(n+2) \\ \vdots & \vdots & \vdots \\ E(K-2) & E(K-1) & E(K) \end{bmatrix}$$

We display in Fig.6.10,Fig.6.11,Fig.6.12 and Fig.6.13 reduced trajectories of BP and GBPs for SNR $\in \{2.00, 2.30, 2.90, 3.09\}$dB, respectively.



(a) BP (b) GBP-C

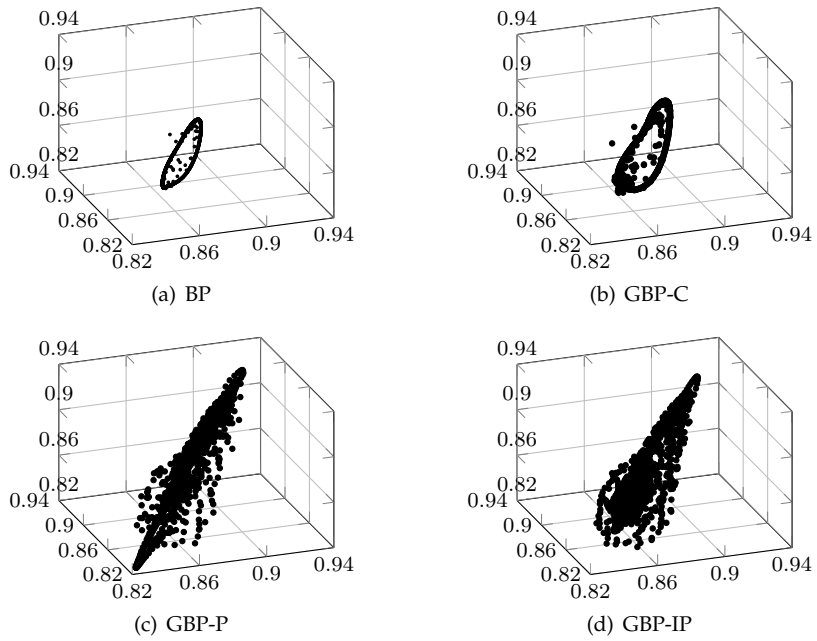
(c) GBP-P (d) GBP-IP

Figure 6.10: *Reduced trajectory with SNR* $= 2.00dB$

In the case SNR $= 2.00$dB, we observe the same convergence in spiral to the same fixed-point for any of the decoders. This fixed-point corresponds to what is called a faulty decoding in the study of the MSB.

We increase the SNR to 2.30dB meaning that we enter the second SNR interval, where decoders behaviors are distinguishable. We observe that BP and GBP-C clearly oscillates[16]. Associated limit cycles look like ellipses, which axes, either minor or major, are oriented in the same direction for both of them. The only difference is that the GBP-C limit cycle is larger than that of the BP, *i.e.* oscillation of GBP-C are more important. Considering GBP-P and GBP-IP, we recall the damping helps them converge, making observations of limit cycle impossible, both end in fixed-points.

[16]the line drawn by oscillations is called a limit cycle



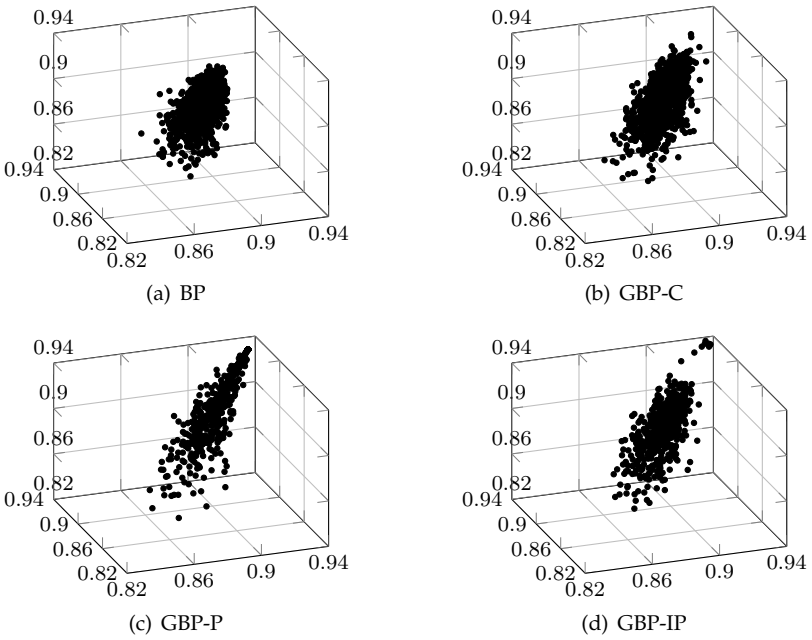
(a) BP (b) GBP-C

(c) GBP-P (d) GBP-IP

Figure 6.11: *Reduced trajectory with SNR* $= 2.30dB$

Associated trajectories do not exhibit any obvious convergence as in the previous case SNR= 2.00dB. In addition, we perceive similar continuous lines inside their clouds of points, as for GBP-C and BP. These lines stem from the fact that damping, either parabolic or inverse parabolic, makes GBP eventually leave a "transient" limit cycle after a certain number of iterations k, as we also saw in Fig.6.4. Limit cycles are oriented in the same direction, as for BP and GBP-C, towards the all-one point, *i.e.* the perfect decoding state. More precisely, ellipses of all decoders are close to the straight line, denoted by Δ, that goes from the all-zero point (completely faulty decoding) to the all-one point (perfect decoding), which is the unitary linear function in the reduced state space. This means that all decoders do not discriminate between reduced state variables, they are somehow equivalent. In other words, MSB smooths beliefs such that EE do not strongly affect the reduced trajectory.

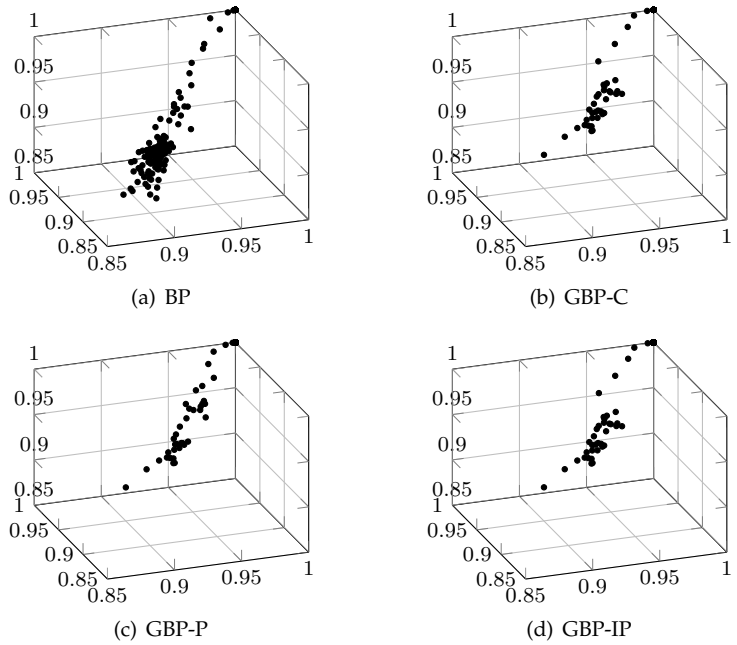
We now increase the SNR to 2.90dB. Decoders behavior appears chaotic according to the bifurcation diagram conclusions. We do not notice any particular line or known geometric form, clouds of points are themselves chaotic attractors: two close points in such an attractor are mapped by any update function to separate points that belong to this chaotic attractor. However, we observe that GBP has the advantage of stretching the attractor along Δ line, contrary to BP where the attractor shape is unchanged from any state space point of view. GBP tends to make state variables "more equivalent" than BP, *i.e.* EE is better smoothed by GBP than by BP, which is a clear advantage of the GBP algorithm. Convergence jumps of GBP-P and GBP-IP confirm that the stretched form of the chaotic attractor helps GBP correctly and quickly converge.



(a) BP (b) GBP-C

(c) GBP-P (d) GBP-IP

Figure 6.12: *Reduced trajectory with SNR* $= 2.90dB$



(a) BP (b) GBP-C

(c) GBP-P (d) GBP-IP

Figure 6.13: *Reduced trajectory with SNR* $= 3.09dB$

When SNR= 3.09dB, we observe that all decoders, whatever the damping factor for GBP, eventually converge to the perfect decoding fixed-point. Furthermore, the number of iterations needed to converge are clearly fewer for GBP than for BP, making GBP consistent with the previous observations concerning the convergence jumps.

The method of the reduced trajectory allows us to visualize and confirm nature of typical decoders behaviors. A numerical method now turns out necessary to associate to any attractor intensity and size, to relate them to decoding performance.

§ 6.7 Sensitivity to initial conditions

Reduced trajectories and bifurcation diagrams bring out that decoders fall into attractors of different kinds, depending on the SNR. In the interval [2.51 dB, 2.99 dB], related dynamical systems oddly behave as attractors, in reduced spaces, do not present any regular geometric shape. This observation is an evidence that decoders do greatly depend on initial conditions, *i.e.* on likelihoods. In other words, decoders reveal stability: a dynamical system is said *stable* if and only if it is not strongly affected by slight changes in its initialization, otherwise it is *unstable*.

6.7.1 Preliminaries

Describing sensitivity to initial conditions is usually done with Lyapunov exponent [A. 85, M.T93, Hiloo]. As explained in 3.4, this quantity evaluates the divergence rate λ between two trajectories initially as close as possible[17]. As an example, we consider two very close likelihoods vectors: $\mathbf{L}_A = [l_{A,1}(x_1, y_1) \dots l_{A,N}(x_N, y_N)]$, $\mathbf{L}_B = [l_{B,1}(x_1, y_1) \dots l_{B,N}(x_N, y_N)]$. We practically define:

$$\forall X_i \in \mathbf{X}, \forall x_i \quad l_{B,i}(x_i, y_i) = l_{A,i}(x_i, y_i) + \epsilon_i \tag{6.10}$$

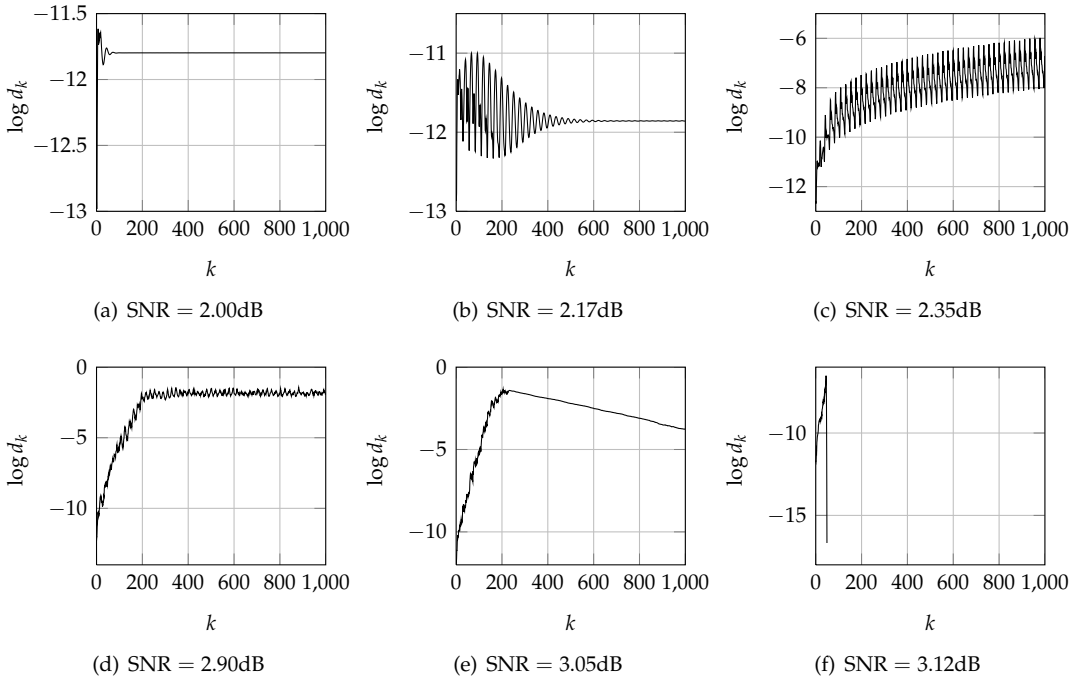
where $\{\epsilon_i\}_i$ are random disturbances between 0 and a maximum value we set to 10^{-6}[18]. For each set of N disturbances, likelihoods vectors are injected in a given decoder, either BP or GBP. Beliefs are computed at each iteration k to compute the Euclidean distance d_k between trajectories:

$$d_k = \sqrt{\sum_{i=1}^{N} \left(b_{A,i}^{(k)}(0, y_i) - b_{B,i}^{(k)}(0, y_i) \right)^2} \tag{6.11}$$

We observe the iterative evolution of $\log d_k$, known as the *log-distance*. The use of the logarithm function appears legitimate as theoretically, the largest divergence, or separation, found between two trajectories is an exponential function of k. The logarithm allows us to cancel the exponential and to extract the divergence rate. We present in Fig.6.14 evolutions of the log-distance of BP for SNR values belonging to SNR intervals previously extracted.

[17] in the true state space, not in the reduced space

[18] this is arbitrary, it depends on experimenter precision



(a) SNR = 2.00dB (b) SNR = 2.17dB (c) SNR = 2.35dB

(d) SNR = 2.90dB (e) SNR = 3.05dB (f) SNR = 3.12dB

Figure 6.14: *Evolution of the BP log-distance*

In figures Fig.6.14(a) and Fig.6.14(b), the distance converges to a very small non zero value, meaning that both trajectories have reached stationary and very close points in the state space. We evaluate d_K around 7.10^{-6} for both SNR, which is around the same magnitude as the distance between the likelihoods. Both trajectories then do not substantially get away one from the other. We note a transient phase for SNR = 2.17dB that lasts about 600 iterations where trajectories seem to pseudo-oscillate one around the other before collapsing. This behavior is an announcement of instability, given that as the SNR is increased, this transient phase lasts more and more.

When increasing the SNR to 2.35dB, see Fig.6.14(c), the log-distance is getting larger in average, *i.e.* trajectories move away from each other. Furthermore, oscillations do not stop anymore, contrary to the previous SNR values, meaning that trajectories are also turning around. As the SNR is increased theses oscillations change to strong divergence between trajectories, *e.g.* the log-distance reaches a value of 0.19 for SNR = 2.90dB, represented in Fig.6.14(d), meaning that the divergence has been multiplied by 2.10^4 between both, that is a considerable change. In addition, we observe an exponential divergence rate up to 200 iterations, *i.e.* trajectories have reached the maximum separation, that is a sign of very high sensitivity to the initial conditions. After 200 iterations, d_k reaches a stationary state, called *plateau*, of value $d_K = 0.19$, a substantial value. This plateau only appears because we observe the log-distance, as the log-function compresses values of the true distance. As an example, we display in Fig.6.15 the distance at SNR = 2.90dB for the BP.

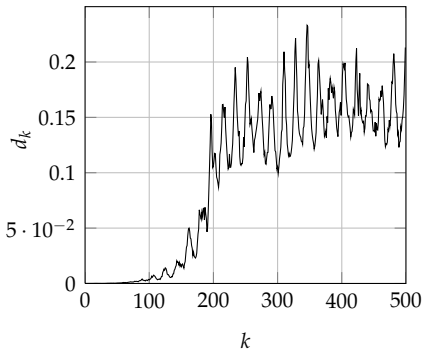


Figure 6.15: *Evolution of the BP distance at* 2.90*dB*

Values of d_k after the exponential divergence range in $[0.1; 0.23]$, with chaotic oscillation amplitudes. It is then erroneous to conclude that the apparent constant log-distance corresponds to a constant d_k. We specify that the maximum value of d_k does not exactly reflect the size of the chaotic attractor, we will tackle this point in section 6.8.

Increasing the SNR beyond the next critical value around 3dB makes trajectories getting closer and closer. We still observe, in Fig.6.14(e), an exponential divergence during first iterations $k \leq 200$, but the distance then exponentially decreases, *i.e.* trajectories that first seemed to move away eventually slowly merge. This phenomenon is confirmed, in Fig.6.14(f), with SNR= 3.12dB, where trajectories shun one another during 45 iterations and from $k = 45$, they completely fuse together.

6.7.2 Lyapunov exponent

Lyapunov exponent is a good candidate to accurately reflect these observations. According to 3.4, Lyapunov exponent λ is the divergence rate between two initially close trajectories:

$$d_k = d_0 e^{\lambda k} \tag{6.12}$$

Sign and magnitude of λ indicate the divergence or convergence nature of a given dynamical systems. Exponentially divergent trajectories are reflected by $\lambda \geq 0$, whereas merging trajectories correspond to $\lambda < 0$. We recall 3.4 that the practical method for computing λ is to extract the log-distance slope given by:

$$\log d_k = \log d_0 + \lambda k \tag{6.13}$$

Usually, computing λ is well-suited for chaotic attractor as they are responsible for exponential divergence. More precisely, the shape of $\log d_k$ help deduce whether a dynamical system is attracted into a chaotic attractor or not. As a matter of fact, the value at last iteration K strongly conditions the slope value, and then the nature of the corresponding attractor. As an example, for an SNR of 3.05dB, we see, in Fig.6.14(e), that taking $K = 200$ provides a higher λ value than taking $K = 1000$ because the shape of $\log d_k$ is changing.

Furthermore, for SNR= 3.12dB, selecting $K = 45$ involves a positive λ whereas $K \geq 200$ involves a negative λ. Accordingly, we strongly recommend to define Lyapunov exponent as the linear regression slope not on the transient part of $\log d_k$ but on the whole values of k. This way, we combine emerging divergence and possible steady state with merging evolution, we do not omit any information. We specify that the last iteration K is restricted to one hundred in order to keep consistency with previous results on bifurcation diagram, reduced trajectory and MSB. The linear regression provides the following equation of Lyapunov exponent:

$$\lambda = \frac{\sum_{k=1}^{K} k \log d_k - \frac{1}{K}\left(\sum_{k=1}^{K} k\right)\left(\sum_{k=1}^{K} \log d_k\right)}{\sum_{k=1}^{K} k^2 - \frac{1}{K}\left(\sum_{k=1}^{K} k\right)^2} \tag{6.14}$$

We show in Fig.6.16 the λ evolution[19] as a function of the SNR.

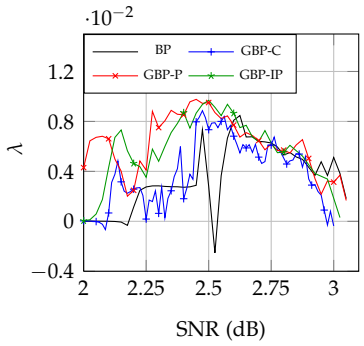


Figure 6.16: *Lyapunov exponent on the Tanner code*

We observe that the SNR intervals indicated by the bifurcation diagrams are still valid, especially for BP:

- first interval: no change in distance, $d_k \approx d_0$,
- second interval: distance increases to a stationary value, trajectories turn around inside a limit cycle,
- third interval: distance increases meaning that trajectories follow very different evolutions, trajectories are trapped into a chaotic attractor,
- fourth and fifth intervals: λ reaches infinite negative value, trajectories merge to a fixed-point.

However, non-trivial behaviors of GBPs appear earlier than BP. The values of λ significantly increase near 2.12dB. Furthermore, while Lyapunov exponent of BP is stabilized to $\lambda = 3.10^{-3}$ in the second interval, GBP-P and GBP-IP exhibit a stronger sensitivity to initial conditions as $\lambda \approx 0.8$.

[19] averaged over a thousand simulations

In contrast, GBP-C presents values of λ comparable to those of BP, that indicates that their sensitivity are quite equivalent. We then observe a common decreasing behavior in the third interval for all decoders, we can then conclude that all decoders are subject to a similar chaos intensity. We specify that it does not imply that chaotic attractors are identical, but that their sensitivity to a slight change in the likelihoods are equivalent.

By the use of the Lyapunov exponent, we are able to quantify the behaviors nature and their intensity, that helps confront decoders. We now deal with the size of the attractors so as to relate them to decoders performance.

§ 6.8 Hyperspheres and attractors width

6.8.1 Static hypersphere

The nature of an attractor, *e.g.* fixed point, limit cycle or chaos, is not enough to describe decoders and to relate with their decoding performance. It appears that widths of chaotic attractors may be very unpredictable and that they largely influence algorithms performance: as the size of a chaotic attractor is increased, beliefs exhibit more and more fluctuations making decoding more and more troublesome.

The size of an attractor is here defined as a measurement of its hypervolume in the state space. As we previously saw, nothing ensures that shapes of chaotic attractors are usual geometric forms (spheres, ellipsoids, hypercubes, etc.), therefore computing the value of the hypervolume of a given chaotic attractor is unfortunately hopeless because it is completely dependent on its shape. To circumvent this problem, we establish a procedure, that we called the *hyperspheres method*: for a given attractor, it consists in approximating its hypervolume by the hypersphere circumscribed to any of its trajectories. Practically, we only compute the radius R of the hypersphere given that the corresponding hypervolume is proportional to R^N. R is then a measure of attractors width.

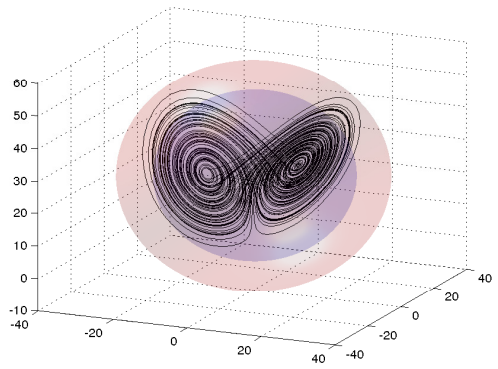


Figure 6.17: *Lorenz chaotic attractor: hypersphere from d_K in blue, from our own method in red*

Practically, we compute R first by extracting the mean point of the attractor, then by searching for the attractor point that is the farthest one. The Euclidean distance between them is the radius of the hypersphere. One would point out here that any attractor width should be provided by the maximum of d_k, the distance between initially nearby trajectories. This is quite realistic as the radius R resulting from the hyperspheres method is not far from the maximum of d_k. As an example, we consider the Lorenz map in its chaotic behavior, see chapter 3.2.2 for details. We recall the profile of the chaotic attractor in Fig.6.17. The red hypersphere stems from our own computation method whereas the blue one is computed by means of distance d_k. We observe that the attractor is enclosed inside the red one without any ambiguity, but not in the blue one, *i.e.* it is less accurate than our method. To reach same results for both methods, it would be required to compute an average radius over a numerous number of trajectories couples, but it represents a too large computation time, compared with the hyperspheres method.

We display experimental results on BP and GBP in Fig.6.18, where are exhibited radii of associated circumscribed hyperspheres as functions of the SNR. We observe that BP falls into larger and larger attractors as the SNR is increased, from 2.18dB to 2.80dB, where the width reaches its maximum value $R_{BP} = 3.30$. As a matter of fact, results show that chaotic attractors of BP, and of GBPs too, are the largest ones.

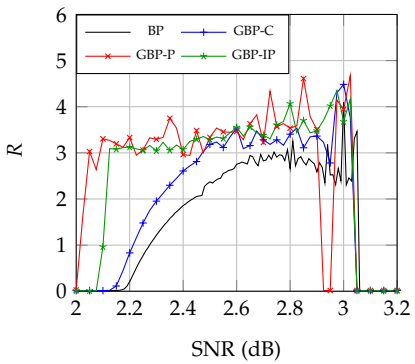


Figure 6.18: *Radii evolution of BP and GBP*

In the last SNR values, R_{BP} decreases slowly up to an abrupt peak at 3.00dB. This may be due to the fact that chaos remains only a transient behavior at this SNR value, the steady state being the right decoding state where $R_{BP} \approx 0$. In other words, convergence jumps, see Fig.6.13, make inaccurate the method when attractors are not centered around the computed mean point but farther. We also observe this phenomenon with GBPs around the same SNR value.

Contrary to GBP-C which static radius follows the same evolution as for BP, we see that GBP-P and GBP-IP present a significant and early increase in the attractors width, at 2.05dB and 2.10dB, respectively. While BP and GBP-C are still enclosed in small-sized attractor, attractors of GBP-P and GBP-IP get substantially larger up to their average steady value $R = 3.25$. By increasing the SNR, both decoders fall into chaotic attractors which widths are not stable, contrary to BP where fluctuations of R_{BP} are less important.

Finally, the static radius reveals that damped GBPs have their trajectory enclosed in quite constant-sized attractors, their widths are a little larger than the maximum width of BP attractors.

6.8.2 Local hypersphere

The hyperspheres method allows us to find an estimate of any attractor radius R. We observe when SNR reaches value of convergence jumps appearance that R was not very reliable, because transient disturbs the measurement. On one hand, transient phase, especially for chaotic attractors, may last about 200 iterations as we can see in Fig.6.14(d), which is a non negligible value. On the other hand, we previously mentioned that the number of iterations should not exceed a hundred to preserve a consistency with realistic situations. Therefore, it would be very useful to determine how to modify the hyperspheres method to fit with realistic situations that may not reach steady states.

The idea is to transform static radius R to a dynamical radius R_k: rather than considering the whole trajectory, we consider a temporal window, of a finite length $W < K$, sliding step by step along the trajectory, each step k giving the radius R_k of the hypersphere circumscribed to the partial trajectory. The hyperspheres method is then transformed such that it results in a sequence of local radii $\{R_{1+\frac{W}{2}}, \ldots, R_{K-\frac{W}{2}}\}$.

Practically, for each point T_k of the trajectory, $k \in \{1 + \frac{W}{2}, \ldots, K - \frac{W}{2}\}$, we compute the trajectory mean point B_k in the temporal window $W_k =]k - \frac{W}{2}, k + \frac{W}{2}]$. After that, we search for the farthest point to B_k inside W_k. R_k is then computed as the Euclidean distance between both points, see representation in Fig.6.19.

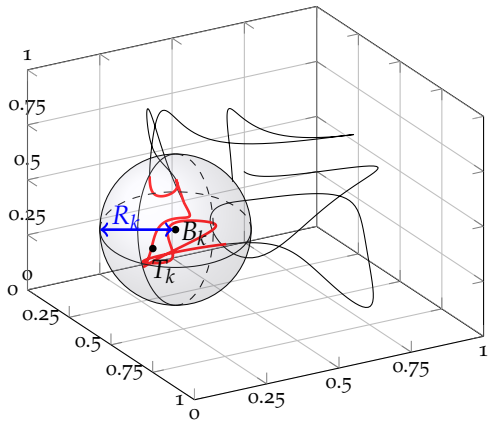
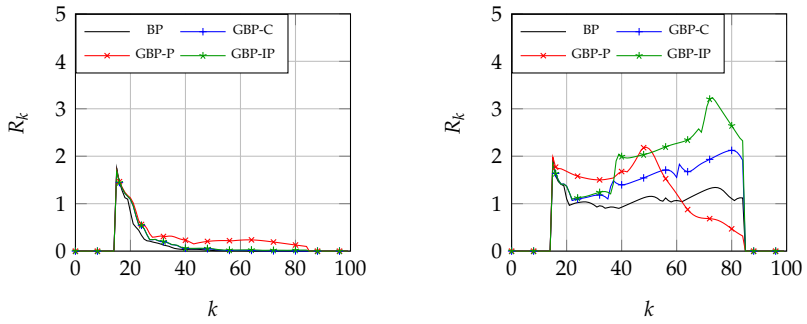


Figure 6.19: *Hypersphere of radius R_k centered on mean point B_k of the partial trajectory $\{T_{k-\frac{W}{2}}, \ldots, T_{k-\frac{W}{2}}\}$*

The dynamical radius R_k allows us to observe iterative evolutions of attractors using temporal windows. The term attractor has slightly changed as it refers here to the set of points of a given trajectory enclosed in W_k, *i.e.* it is only a partial attractor. The dynamical hyperspheres method then use local average behaviors to observe iterative evolutions of dynamical systems.

For any iteration k, a large value of R_k means that the given decoder has significant changes, whereas a small value of R_k indicates that the decoder is slightly changing. In other words, local, or dynamical, hyperspheres provide the decoder scattering.

The window length W wields influence on results of the method. On one hand, selecting a too large value of W could make the method find a non varying radius along iterations as it would be too close to the static radius, *i.e.* the dynamical local hyperspheres would be similar to the static hypersphere circumscribed to the whole trajectory. On the other hand, opting for a too small length W implies that local hyperspheres are completely determined by trajectory points themselves, *i.e.* no information can be enlightened on the attractor width. In our experiments, we chose a window of length $W = 30$. The reason for this choice is that this value is around the minimum practical number of iterations used to run any decoder[20]. We present in Fig.6.20 the R_k evolution for BP and GBPs when trapped in attractors previously mentioned[21].



(a) SNR = 2.00dB (b) SNR = 2.30dB

(c) SNR = 2.90dB (d) SNR = 3.09dB

Figure 6.20: *Local radii R_k evolution for BP and GBP*

When the SNR equals 2.00dB, decoders are attracted by fixed-points, as depicted in Fig.6.10. We previously saw that corresponding reduced trajectories drew spirals in reduced spaces, justifying R_k as hyperbolic functions of k.

[20] no reference asserts it, but it is a commonly accepted computation detail

[21] values of R_k are not available for $k < \frac{W}{2}$ and $k > K - \frac{W}{2}$, that is why figures exhibit null values outside the interval $[\frac{W}{2}, K - \frac{W}{2}]$

We now consider that SNR $=$ 2.30dB. We note that BP radius is quite constant compared with GBPs ones. At this SNR value, BP is trapped into a limit cycle, that explains the relative stability of R_k. Concerning the GBP algorithms, we observe that the R_k shape of GBP-C behave in the same way that BP one with an increasing dilatation as k increases, meaning that the associated limit cycle is getting larger and larger. A noteworthy point is that the GBP-P radius diminishes for $k \geq 48$, *i.e.* the limit cycle attractor retracts. In contrast, the GBP-IP radius does not present any decreasing values before $k = 72$, meaning that the damping factor does not help focus on a single point of the state space.

As the SNR is set to 2.90dB, decoders are trapped into chaotic attractors. These attractors are larger than the previous limit cycles, as R_k values are increased about a unit. We observe that the BP attractor slowly collapses as k is increased, that could be approximated by an affine function, which makes emerge that inside this chaotic attractor, fluctuations are linearly decreasing. While $k \leq 60$, we can distinguish that GBP-C and GBP-IP follow a BP-like evolution, *i.e.* R_k slowly decreases in average too. In contrast, GBP-P evolves pseudo-oscillating in its chaotic attractor with R_k values close to GBP-C ones. As $k > 60$, each GBP changes its behavior, due to damping factors. Radii of GBP-C and GBP-IP grow up to $R_k > 3$ whereas GBP-P radius decreases to $R_k \leq 0.5$, which is even lower than BP radius. Finally, for a SNR of 3.09dB, all GBPs converge to smaller attractors than BP one. More precisely, GBP-C and GBP-IP both converge to a fixed-point ($R_k = 0$) and GBP-P slowly converges to a smaller and smaller attractor, whereas BP is still stuck to an attractor of radius $R_k \approx 2$. This piece of information could not be revealed by the static radius, that makes the use of the local radius relevant.

These observations tend to conclude that damping factors wield substantial influence on GBPs attractors, especially when these attractors are non-trivial. We see that to select either a damping rule or another one deeply conditions GBP performance. However, when approaching the major bound of the chaotic interval, the GBP algorithms tend to converge to fixed-points contrary to BP that is still stuck in an attractor of a non-negligible size. In other words, GBP diverge more but shorter than BP.

Thanks to the hyperspheres method, we are able to offer measurements of attractors hypervolumes for all decoders, which particularly helps contextualize chaos in decoding performance. The remaining question now is where these attractors take place in their associated state space, or more simply, where trajectories evolve in the state space.

§ 6.9 Lengths ratio to determine the relative position in the state space

By means of previous descriptors, we now have information about attractors: we know their nature, their size and the SNR values for which they appear. The last description we need is about relative positions of these attractors to the perfect decoding state in the state space, that will be offered by the last quantifier, called the *complementary length*.

6.9.1 Complementary length

The length of any beliefs vector, in the state space, is defined as its Euclidean distance to all-zero point:

$$L(k) = \sqrt{\sum_{i=1}^{N} \left(b_i^{(k)}(0) \right)^2} \tag{6.15}$$

which is very close to MSB. To keep connections with concept of length and positions, we prefer to keep using $L(k)$ instead of $E(k)$. We now define the complementary length as the Euclidean distance to all-one point:

$$\bar{L}(k) = \sqrt{\sum_{i=1}^{N} \left(1 - b_i^{(k)}(0) \right)^2} \tag{6.16}$$

Experimentally, we always consider that the emitted codeword is the zero-codeword, therefore, all-one point in the state space corresponds to perfect decoding[22]. The vector length is then the distance to completely faulty decoding, or wrong decoding. As a matter of fact, only one of these two quantities does not provide decoding performance of a given decoder. We display a toy example in Fig.6.21 where we consider a three-dimensional space which axes are denoted by b_1, b_2, b_3.

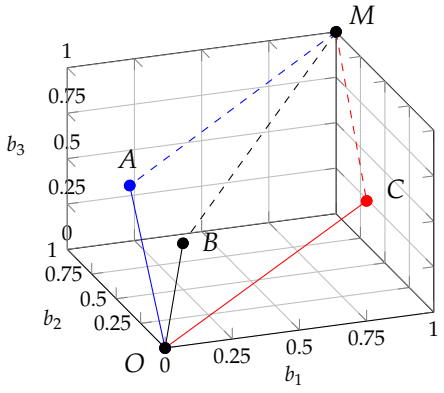


Figure 6.21: *Length and complementary length*

Coordinates of points A,B and C are given in table 6.3. We include in this table lengths and complementary lengths of vectors $\overrightarrow{OA}, \overrightarrow{OB}, \overrightarrow{OC}$. Points A, B, C are equidistant to all-zero point O, as $L_A = L_B = L_C$. However, they are not equidistant to all-one point M as $\bar{L}_A < \bar{L}_C < \bar{L}_B$, *i.e.* point B is the closest to point M. This kind of information is not given by the only knowledge of length L, as it does not take into account any other point that M. Using also $\bar{L}$ helps faithfully represent the relative position of any decoder in its state space.

[22] for any variable node X_i, $b_i(0) = 1$

	A	B	C
b_1	0.2	0.4	0.75
b_2	0.911	0.9165	0.0
b_3	0.3607	0.0	0.6614
L	1.0	1.0	1.0
$\bar{L}$	1.0279	1.1692	1.0850

Table 6.3: *Coordinates, lengths and complementary lengths*

6.9.2 Lengths ratio

We now apply concept of complementary length to extract positions of decoders inside their own state space and more accurately, their vicinity with the optimal point[23]. Instead of using length and complementary length, we introduce a combination thereof to obtain a unique function that reflects the relative decoders positions to the correct decoding state[24]. This combination is called the *lengths ratio* and is given by:

$$\alpha_k = \frac{\bar{L}(k)}{L(k)} \tag{6.17}$$

This allows us to judge about the accuracy of BP and GBP. For any couple of beliefs vectors of identical lengths, α_k brings out the closest one to the right decoding state which is also the farthest one to the wrong decoding state. In the previous example, lengths ratios associated to points A, B, C are: $\alpha_A = 1.0279$, $\alpha_B = 1.1692$ and $\alpha_C = 1.0850$. As their lengths are equal, we consistently find that $\alpha_A < \alpha_C < \alpha_B$. Two points of identical lengths ratios are comparable in the state space as their decoding performance are similar. We display in Fig.6.22 the surface covered by α_k in the plane generated by all possible values of $L(k)$ and $\bar{L}(k)$.

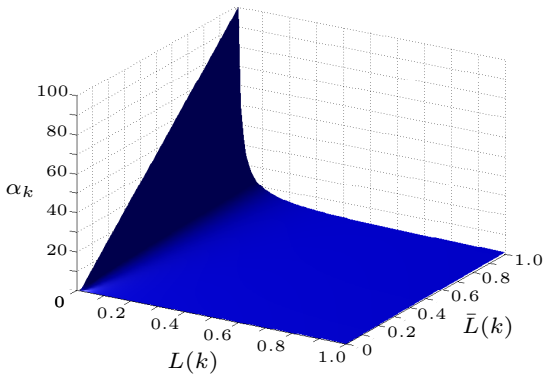


Figure 6.22: *Lengths ratio*

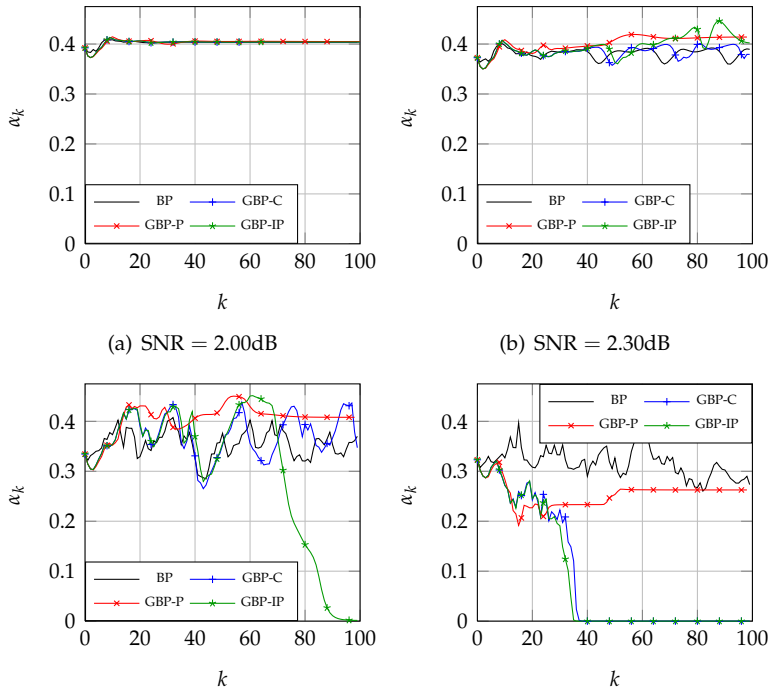
[23] all-one point

[24] both quantities are required as they play dual roles in state spaces

We link typical values of α_k with decoding situations:

- $\alpha_k = 0$: perfect decoding, $\bar{L}(k) = 0$,
- $\alpha_k = +\infty$: completely faulty decoding, $L(k) = 0$,
- $\alpha_k = 1$: indecisive decoding, entropy is maximum on the beliefs.

Thus, the smaller α_k is, the more relevant the decoding is. In Fig.6.23 are displayed iterative evolutions of α_k for BP and GBP.



(a) SNR = 2.00dB (b) SNR = 2.30dB

(c) SNR = 2.90dB (d) SNR = 3.09dB

Figure 6.23: *Iterative evolution of the lengths ratio*

We observe for SNR= 2.00dB that all decoders are exactly in the same relative position as $\alpha_k = 0.4$. This does not mean that beliefs are common to all of them, but that trajectory points are equidistant to all-zero and all-one points. Once the SNR reaches 2.30dB, we know that decoders oscillate, especially BP and GBP-C[25]. Only small fluctuations are visible around the average value 0.4, meaning that oscillations are not very large compared with the size of the state space[26].

We previously mentioned, by means of static radius of hyperspheres, that attractors are getting larger as the SNR is increased. This is why at 2.90dB, significant differences between decoders behaviors are visible. In addition, attractors are chaotic and of non-negligible widths.

[25] as GBP-P and GBP-IP present pseudo-oscillations

[26] state space made with beliefs is a hypercube which largest diagonal length is $\sqrt{N}$, *i.e.* around 12.4 for Tanner code

Lengths ratio of any decoder is then subject to important variations. GBP-IP lengths ratio has the great advantage of converging to the perfect decoding state, contrary to other ones. However, by increasing again the SNR, *e.g.* 3.09dB, we observe that GBP-C also presents such a behavior. Surprisingly, GBP-P still does not converge to the right state even though its lengths ratio has been diminished compared with that of the previous SNR. In addition, this lengths ratio is lower than the BP one, indicating that BP is even more remote to the right state than GBP-P. Not displayed experiments have showed that increasing again the SNR makes lengths ratios of all algorithms converge to zero.

The noteworthy point is that the GBP algorithm, especially damped with the inverse parabolic law, presents a better proximity to the perfect decoding state than other decoders. This fact is not particular to the error event EE we selected. Other experiments demonstrated that the lengths ratio of GBP is generally smaller than the BP one. In other words, for any initial condition, GBP approaches the right state more accurately than BP.

Practically, performance of any decoder are measured once either messages have converged, or the estimate word is valid according to the LDPC code, or the last iteration has been reached. We then picked out the value of the steady lengths ratio α_K for different SNR values, K being the iteration where the given decoder stopped[27]. We average values of α_K over a thousand error events close to the original error event EE, at the Euclidean sense. Results are represented in Fig.6.24.

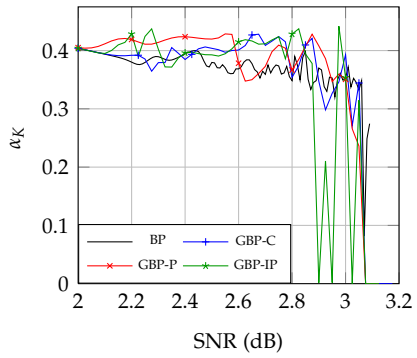


Figure 6.24: *Steady lengths ratio along the SNR for BP and GBP*

For SNR values lower than 2.00dB, lengths ratios of all decoders merge in a single line, as well as when SNR values greater than 3.1dB. As previously explained, the α_K behavior may fluctuate more and more as the SNR is increased up to 2.30dB. As a summary of previous observations, we observe that from this SNR value, GBP-IP tends to approach the right decoding state significantly earlier than other decoders, that is a manifestation of the convergence jumps. In addition, we see that just after 3.00dB, GBP-P and GBP-IP converge to all-one point, whereas GBP-C and BP has to wait a little bit longer. Furthermore, BP exhibits a rebound once all other algorithms have converged, which is a sign of worse decoding.

[27] less than one hundred to maintain realistic situations

§ 6.10 Conclusion

The study of dynamical behaviors of iterative decoders brought out properties cannot be deduced from the only knowledge of the average BER. We saw that BP and GBP were comparable in terms of BER when the SNR ranges in $[2.00; 3.10]$dB. But when focusing on non-trivial error events, we observed different evolutions. Bifurcation diagram provided critical SNR values for which message-passing algorithms encountered particular attractors that prevent them from easily converging. We observed that for all decoders, critical SNR values and even SNR intervals are similar, which makes them very interesting challengers. Lyapunov exponents and reduced trajectories indicated that algorithms are trapped by chaotic attractors in the same SNR interval, and it was clear that their sensitivity to initial conditions in this interval was similar, even though it is not the case for lower SNR values, partly due to damping factors.

We introduced few original estimators to analyze widths and relative positions of attractors and trajectories in state spaces. We motivated this by trying to relate dynamical behaviors to decoding performance, that is not an easy task. In addition, the usual estimators from the literature on dynamical systems theory cannot reflect the whole influence of dynamical properties on BP and GBP efficiency. By means of the hyperspheres methods, we first extracted attractors size: it appears that damping rules make GBP evolve in attractors of a non negligible but stationary size, contrary to BP that is enclosed in larger and larger attractors as the SNR is increased, up to the critical SNR values where all algorithms become chaotic. Related to reduced trajectories, chaotic attractors of GBP are more stretched in the direction of the right state whereas BP has a quite uniform shape in its state space. By the use of the lengths ratio, we exhibited the vicinity of decoders to the right state, more accurately than what has been done with MSB. We noticed once more that the chaos disappear earlier for GBP than BP, thanks to decreasing damping rules.

Conclusion

We demonstrated in first chapter that fixed-points of BP are stationary point of Bethe approximation. This property allowed us to extend connections between decoding algorithms and statistical physics. Both methods are subject to a significant suboptimality due to the non-convexity of variational free energies, in deep correlation with the loop-like topology of most Tanner graphs.

We demonstrated in second chapter that Bethe approximation could extend to Region-Based Approximation (RBA), which provided the GBP algorithm, a message-passing algorithm running on edges of a region-graph. Goal of the region-graph construction is to absorb harmful topological structures of any Tanner graph, such that resulting RBA is less damaged than Bethe approximation. We made use of this principle to introduce an original region-graph construction based on trapping sets of Tanner code, that are responsible for the degradation of BP performance, especially for high SNR values. By means of an original method, we brought out relevant results of the GBP algorithm for error events based on trapping sets. We observed that this decoder could surpass BP in these situations, making our own construction well-suited.

In third chapter, we introduced a toolbox to examine dynamics of any iterative decoder. This helps understand non-trivial behaviors of algorithms, *e.g.* the lack of convergence, oscillations or even chaos. By the use of bifurcation diagrams and Lyapunov exponent, we extracted typical attractors that trapped decoders, and SNR values they appear at. We introduced original descriptors of dynamics, that we called hyperspheres and lengths ratio, to discuss size and position of these attractors, and they allowed us to determine their relations with decoding performance of BP and GBP. It appeared that the triplets clustering presented very relevant properties as associated attractors are closer to the right decoding state than BP ones, though Lyapunov exponent demonstrated that BP was a little more stable. In addition, damping rules, that we introduced to overcome the lack of convergence of the pure GBP algorithm, provided that attractors always shrink as the iteration increases, contrary to BP.

Statistically, BP and GBP algorithms are quite comparable. Without any specific construction of the region-graph, results of the GBP algorithm are strictly equal to the results of the BP. This is generally observed as the systematic construction for LDPC codes provides a region-graph which topology is identical to the Tanner graph one. With our dedicated construction, we show that it is possible to make a region-graph such that GBP surpasses BP.

Conclusion

Conclusion

In this thesis, we addressed inference problems on loopy factor graphs. We presented the Belief Propagation algorithm (BP) used as a message-passing algorithm to almost solve inference. We exhibited its deep relations with statistical physics, especially with Bethe approximation, a method that is aimed at minimizing the variational free energy to obtain the log-partition function. By the use of examples and experiments, we showed that loop-like topology of most factor graphs, and more precisely Tanner graphs for LDPC codes, prevents BP from being optimal.

Introduction of the region-based approximation, and the associated Generalized Belief Propagation algorithm (GBP), helped circumvent this problem by mapping the factor graph to the region-graph. Provided that the region-graph is constructed according to the factor graph topology, the GBP algorithm results in more efficient performance than BP. In the case of LDPC codes, we used trapping sets to construct a specific region-graph dedicated to the Tanner code. By means of damping factors and new principles, we developed a method for high SNR values that demonstrated that the GBP algorithm was able to outperform the BP algorithm.

The comparison of both decoders for middle SNR values turned out non conclusive as they exhibit equivalent performance. From the dynamical systems theory point of view, we introduced a relevant investigation that brought out decoders typical behaviors. Bifurcation diagrams offered that the SNR values for which they do not trivially evolve were common. Furthermore, both are subject to chaos for same SNR values. The use of Lyapunov exponent and reduced trajectories revealed that damping factors of the GBP algorithm wield much influence on the decoder stability at a low memory cost. In addition, these functions allowed us to observe that the chaos intensity was similar to both, but that GBP always converged earlier than BP to the right decoding state, especially with a decreasing damping factor. We introduced the static hyperspheres method to approximate the attractors width, especially for chaotic attractors. We observed that the widths of the GBP attractors were quite stationary compared with the BP ones that grew as the SNR is increased. Local hyperspheres helped us conclude that along the iterations, the GBP algorithm is more convergent than BP, due to the damping factor particularly. The introduction of the complementary length and the lengths ratio provided more relevant information than the mean square beliefs as it faithfully reflects the relative position of any decoder to the right decoding state.

We saw that the GBP algorithm presented better length ratios when approaching the major bound of the chaotic interval.

Finally, the specific region-graph construction with a triplets clustering turned out powerful according to the number of errors decoded by the GBP algorithm compared with the BP's. Furthermore, the investigation on dynamical aspects revealed that the GBP behavior provided with damping factors offered a better convergence than the BP algorithm.

Future works

From this work have emerged a certain number of issues that constitute an interesting basis for future works. Points we are about to tell belong to the different research areas that were tackled in this thesis.

- First of all, the clustering in triplets has to be adapted to other LDPC codes of the same family as the Tanner code's, to acquire a larger overview of GBP decoding performance. In addition, it will be necessary to introduce constructions based on other kinds of trapping sets. The investigation here remains on TS$(5, 3)$ but nothing prevents from considering other structures, provided that they are small enough to preserve a reasonable computation complexity.

- The computation complexity of the GBP algorithm is its weak point. Therefore, we began to study the decimation of the region-graph. The method that we started to investigate consists in switching off a fixed percentage of edges – to some extent, messages – in the region-graph that we randomly select without any discrimination. After a given number of iterations, we switch them on and we switch off other edges similarly selected. The two parameters of this method are the percentage of switched off edges and the number of iteration before changing in the edges. Preliminary simulations showed that with 30% of switched off edges during about 10 iterations, the GBP algorithm does not present significant difference in its performance in terms of the BER. By increasing the percentage, we induce a too large modification that damage GBP results. Therefore, it would be relevant to consider a more specific rule of edges selection instead of a uniform one.

- Recently was introduced an alternative to the BP algorithm that consists in weighting messages [H. 12],[H. 11],[T.G08]. It reminds us of the damping factor of the GBP algorithm, even though the mathematical details are different and the goal is not to ensure convergence but to improve results. Furthermore, experiments show that regular BP, *i.e.* BP provided with a unitary weight, is outperformed by non unitary weighted BP. It appeared that this weighting should apply to other iterative decoders, such as GBP, FAID, etc.

- The lack of convergence and the suboptimality of the decoding algorithms are due to the loop-like topology of the Tanner graphs, as we try to minimize non-convex variational free energies. The CCCP proposed in [A.L03] is a powerful tool to treat this problem, even though the double-loops imply a significant increase in the computation time. As a matter of fact, CCCP has been derived for Bethe and region-based variational free energies. Experimenting CCCP on LDPC codes, especially with well-suited region-graphs, *e.g.* the Tanner code provided with the triplets construction, to compare with the original GBP algorithm would be an interesting work.

- In [T. 03a], authors deal with convexity of the Bethe variational free energy. Studying in the same way the region-based free energy to extract sufficient conditions for its convexity, or even to convexify it, would solve many problems.

- One of the drawback of decoders dynamics is that we do not know basins of attraction of attractors. It would be very useful to know where chaotic attractors are in the state space, that would help to control the BP evolution. Independently from the work presented in this thesis, we have demonstrated that it was possible to write a "reverse BP", *i.e.* a message-passing algorithm that goes back in time from a given point. It would help associate to any state its initial conditions. "Downdate" equations unfortunately do not result in a unique previous state but in numerous states. In addition, what is true for regular BP is also true for its reverse version, *i.e.* the sensitivity to initial conditions. Given a point in the state space or a very close neighbor, we obtain by reverse BP very different point when dealing with chaotic attractors. Therefore, we did not go further in this method but it needs a deep investigation to reveal relevant results that would help control BP dynamics, and even GBP dynamics.

List of Figures

1.1 Channel . . . 12
1.2 Two channel models . . . 12
1.3 Bit-error rate of an uncoded transmission . . . 13
1.4 Model with an error-correcting code . . . 14
1.5 Maximum likelihood decoding on the Hamming code . . . 16
1.6 2-level Tanner graph of the Hamming code . . . 17
1.7 Tanner code ($N = 155$) hard decoding on a BSC of crossover probability p . . . 17
1.8 Message m_{ij} in a Bayesian network . . . 19
1.9 Messages in a factor graph . . . 20
1.10 Tanner code decoding ($N = 155$) . . . 21
1.11 CCCP decoding of the Tanner code . . . 23
1.12 Inner loop of the CCCP decoding of the Tanner code . . . 23
1.13 Tanner code ($N = 155$) decoding by BP on AWGNC . . . 24

2.1 Spin glass: the dashed lines are the anti-ferromagnetic interactions and the solid lines are the ferromagnetic interactions, the arrows are the spins (up or down) . . . 28
2.2 Spin glass: dashed lines are anti-ferromagnetic interactions and solid lines are ferromagnetic interactions; arrows are spins (up or down). The red line is an interaction modified by an external noise and red spins are spins modified by interactions. . . . 30
2.3 Ising chain with $N = 3$ spins embedded in an external magnetic field . . . 33
2.4 $N = 1$ . . . 34
2.5 $N = 2$ . . . 34
2.6 $N = 3$ . . . 35
2.7 p-range model . . . 36

3.1 Logistic map $x_{k+1} = 4x_k(1 - x_k)$ . . . 41
3.2 Logistic map . . . 41
3.3 Lorenz attractor . . . 42
3.4 Lorenz map . . . 43
3.5 Logistic map - Period doubling . . . 44
3.6 Bifurcation diagram of the logistic map . . . 45
3.7 Poincaré section $\mathcal{P}$ . . . 46

3.8 Tinkerbell attractor with $a = 0.5$ 46
3.9 Tinkerbell attractor with $a = 0.1$ 47
3.10 Tinkerbell attractor with $a = 0.9$ 48
3.11 Lyapunov exponent estimates 50
3.12 Time evolution of the separation 51
3.13 Fake chaos, $K = 300$ 52
3.14 Solid line is the alternative Lyapunov exponent, dashed line is the Lyapunov exponent from the differential method 53
3.15 Hénon map state space 54
3.16 Lorenz map state space 55

4.1 Tanner graph – $\mathbf{X}_{\mathcal{N}_7} = \{X_1, \ldots, X_6\}, \mathcal{N}_7 = \{c_a, c_b, c_c\}$ 65
4.2 Tanner tree 67
4.3 Square Tanner lattice 67
4.4 Square lattice of length $N = 9$ 68
4.5 Hamming code 68
4.6 Two formulations of the normalization constraint. (a) We only scan $\mathcal{X}$ to extract point A, the argument of the minimum. (b) Two points that live on the same line (B, B'), (C, C') or (D, D') will have the same mapping by $f\left(\frac{\cdot}{||\cdot||}\right)$ then the argument of the minimum if not a unique point but a line. 75
4.7 Wrong formulation of the normalization constraint. The projection of M on $\mathcal{X}$ is different from A. 76
4.8 Tree-like Tanner graph 81
4.9 Non tree-like Tanner graph 82
4.10 Local Tanner graph for messages computations 82
4.11 Tanner graph of a loop 83
4.12 Trapping-set of $a = 5$ variable nodes and $b = 3$ unverified check nodes 85
4.13 BP error floor when decoding the Tanner code covered by TS(5, 3). 85

5.1 Square lattice of length $N = 9$ 88
5.2 A Bayesian network as a graphical representation of (5.2) 89
5.3 Another Bayesian network as a graphical representation of (5.3) 89
5.4 Average errors of MF, Bethe and region-based approximations on square lattice $N = 9$ 92
5.5 Two clusterings for two factorizations 93
5.6 Two region-graphs for two factorizations 94
5.7 Explicit square lattice of length $N = 9$ 100
5.8 Region-graph of the explicit square lattice 101
5.9 Small region-graph 102
5.10 Reduced region-graph 103
5.11 Uniformly reweighted BP ($\rho = 1.00$ is BP) 104
5.12 BER of the GBP with changing w_0 104
5.13 Number of iterations of the GBP with changing w_0 105
5.14 Performance of the GBP for three weights 105
5.15 BER of the GBP with changing w_0 106
5.16 Number of iterations of the GBP with changing w_0 106
5.17 Damping factor with initial value $w_0 = 0.85$ 107

5.18 BER of the GBP algorithm on Tanner code with decreasing damping factors . . . 107
5.19 BER of the GBP algorithm on the Hamming code . . . 108
5.20 Parallel version performance . . . 109
5.21 A systematic region-graph for the Hamming code . . . 110
5.22 Suited region-graph for the square lattice code $N = 9$ . . . 111
5.23 Bit-error rate of the BP and the GBP algorithms on the square lattice code $N = 36$ . . . 112
5.24 Trapping set TS(5, 3) . . . 112
5.25 Split of the TS(5, 3) – Quadruplets construction . . . 113
5.26 Split of the TS(5, 3) – Triplets construction . . . 114
5.27 Region-graph resulting from the split of a TS(5, 3) in two quadruplets . . . 114
5.28 Third quadruplet as a cluster . . . 115
5.29 Region-graph resulting from the split of a TS(5, 3) in quadruplets 115
5.30 Region-graph resulting from the split of a TS(5, 3) in triplets . . 115
5.31 A TS(5, 3) with all variable nodes connected to its check nodes 116
5.32 BER on the Tanner code for EE completely dependent on TS(5, 3) 118
5.33 Modification of moments – Dashed lines are original SNR and mean, solid lines are their practical estimates according to EE . 119
5.34 BER of the BP and the GBP algorithms on a EE with an AWGNC 120
5.35 Average BER of the BP and the GBP-P algorithms . . . 121

6.1 Iterative BER of BP with two different sets of likelihoods at the same SNR . . . 124
6.2 Likelihoods space at SNR = 2.3dB on the Tanner code for quenched likelihoods except on variable nodes $X_i = 100$, $X_j = 102$ – BER of the BP . . . 125
6.3 BP MSB of EE: four SNR values, to each one is found a typical behavior . . . 128
6.4 GBP MSB on EE . . . 129
6.5 Bifurcation diagram of BP . . . 130
6.6 Bifurcation diagram of the BP on the Mac Kay code . . . 131
6.7 Influence of K on the bifurcation diagram of the BP . . . 132
6.8 Bifurcation diagram of GBPs . . . 132
6.9 Chaos in the logistic map $x_{k+1} = ax_k(1 - x_k)$, for $a \geq 3.44$ . . . 133
6.10 Reduced trajectory with SNR = 2.00dB . . . 135
6.11 Reduced trajectory with SNR = 2.30dB . . . 136
6.12 Reduced trajectory with SNR = 2.90dB . . . 137
6.13 Reduced trajectory with SNR = 3.09dB . . . 137
6.14 Evolution of the BP log-distance . . . 139
6.15 Evolution of the BP distance at 2.90dB . . . 140
6.16 Lyapunov exponent on the Tanner code . . . 141
6.17 Lorenz chaotic attractor: hypersphere from d_K in blue, from our own method in red . . . 142
6.18 Radii evolution of BP and GBP . . . 143
6.19 Hypersphere of radius R_k centered on mean point B_k of the partial trajectory $\{T_{k-\frac{W}{2}}, \ldots, T_{k-\frac{W}{2}}\}$ . . . 144
6.20 Local radii R_k evolution for BP and GBP . . . 145

6.21 Length and complementary length 147
6.22 Lengths ratio 148
6.23 Iterative evolution of the lengths ratio 149
6.24 Steady lengths ratio along the SNR for BP and GBP 150

List of Tables

6.1 BP behaviors in terms of SNR values 129
6.2 BP behaviors in terms of SNR intervals 130
6.3 Coordinates, lengths and complementary lengths 148

Bibliography

[A. 59] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres*, 2, 1959. 16

[A. 67] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13, 1967. 18

[A. 85] A. Wolf, J.B. Swift, H.L. Swinney and J.A. Vastano. Determining Lyapunov Exponents from a Time Series. *Physica*, pages 285–317, 1985. 138

[A. 03] A. Shokrollahi. LDPC codes: An introduction. *Digital Fountain, Inc., Tech. Rep.*, 2003.

[Akr11] N. Akroune. Sur un aspect numérique de la dimension fractale d'un attracteur chaotique. *Matematicki Vesnik*, 63, 2011.

[A.L03] A.L. Yuille and A. Rangarajan. The Concave-Convex Procedure (CCCP). *Neural Computation*, 15:915–936, 2003. 22, 79, 159

[An88] G. An. A Note on the Cluster Variation Method. *Journal of Statistical Physics*, 52, 1988.

[Asa00] H. Asada. A New Statistical Aspect of the Cluster Variation Method for Lattice Systems. *Journal of Statistical Physics*, 98, 2000.

[B. 00] B. Caputo. A spin glass model of a Markov random field. In *Proc. Neural Information Processing Systems*, 2000. 28

[B. 09] B. Vasic, S.K. Chilappagari, D.V. Nguyen and S.K. Planjery. Trapping set ontology. In *Proc. 47th Allerton Conference on Communication, Control and Computing*, 2009. 84

[B. 11] B. Cseke and T. Heskes. Properties of Bethe Free Energies and Message-Passing in Gaussian Models. *Journal of Artificial Intelligence*, 41, 2011.

[Bas02] M. Bask. A positive Lyapunov exponent in Swedish exchange rate ? *Chaos, Solitons & Fractals*, 14, 2002.

[Bou09] J. Le Bourlot. *Introduction aux systèmes dynamiques disspatifs*. Observatoire de Paris, 2009.

[B.S10a] B.S. Ruffer, P. Dower, C.M. Kellet, S.R. Weller,. On robust stability of the belief propagation algorithm for LDPC decoding. In *Proc. 19th International Symposium Mathematical Theory of Networks and Systems*, 2010. 25

[B.S10b] B.S. Rüffer, C.M. Kellet, P.M. Dower and S.R. Weller. Belief Propagation as a Dynamical System: The Linear Case and Open Problems. *IET Control Theory and its Applications*, 4, 2010. 25

[C. 93] C. Berrou, A. Glavieux and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: turbo-codes. In *Proc. Int. Conf. on Communications*, 1993. 15

[C. 95] C. Ngô and H. Ngô. *Physique statistique à l'équilibre et hors équilibre*. Masson, 1995.

[CVM96] Theory and Applications of the Cluster Variation and Path Probability Methods. Plenum Press, 1996.

[D. 75] D. Sherrington and S. Kirkpatrick. Solvable model of a spin-glass. *Physics Review Letters*, 35, 1975. 29, 30

[D. 95] D. J. C. MacKay and R. M. Neal. Good codes based on very sparse matrices. In *Proc. 5th IMA Conference*, 1995. 18, 63

[D. 00] D. Agrawal. The turbo decoding algorithm and its phase trajectories. In *Proc. International Symposium on Information Theory*, 2000. 24

[D. 11] D. V. Nguyen, B. Vasic and M. W. Marcellin. Two-bit bit flipping decoding of LDPC codes. In *Proc. International Symposium on Information Theory*, 2011. 17

[D. 12] D. Declercq, E. Li, S.K. Planjery and B. Vasic. Approaching maximum likelihood decoding of finite length LDPC codes via FAID diversity. In *Proc. Information Theory Workshop*, 2012. 21, 85

[D.J01] D.J.C. MacKay, J.S. Yedidia, W.T. Freeman and Y. Weiss. Discussion document: A conversation about the Bethe free energy and sum-product, 2001.

[D.P01] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2001. 18, 74, 78, 79, 102

[D.V12] D.V. Nguyen, S.H. Chilappagari, M.W. Marcellin and B. Vasic. On the Construction of Structured LDPC Codes Free of Small Trapping Sets. *IEEE Transactions on Information Theory*, 58, 2012. 85

[E. 09] E. Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55, 2009. 16

[E. 10] E. Riegler, G.E. Kirkelund, C.N. Manchón and B.H. Fleury. Merging Belief Propagation and the Mean Field Approximation: A Free Energy Approach. In *Proc. 6th International Symposium on Turbo Codes*, 2010.

[E.B07] E.B. Sudderth and M.J. Wainwright. Loop Series and Bethe Variational Bounds in Attractive Graphical Models. In *Proc. Neural Information Processing Systems*, 2007.

[F. 06] F. Guerra. An introduction to mean field spin glass theory: methods and results. *Mathematical Statistical Physics*, 47, 2006. 30

[F. 11] F. Krzakala. Belief Propagation for the (physicist) layman, 2011. 33, 37

[F. 12] F. Zamponi. Mean field theory of spin glasses, 2012. 29

[F.R98] F.R. Kschischang and B.J. Frey. Iterative decoding of compound codes by probability propagation in graphical models. *IEEE Journal of Selected Areas in Communications*, 16, 1998. 18

[F.R01] F.R. Kschischang, B. J. Frey eand H.A. Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47, 2001. NO. 2. 16, 18, 19, 66

[fra89] A.M. fraser. Information and Entropy in Strange Attractors. *IEEE Transactions on Information Theory*, 35, 1989.

[G. 10] G. Duclos-Cianci and D. Poulin. A renormalization group decoding algorithm for topological quantum codes. In *Proc. Inf. Th. Workshop*, 2010. 38

[Gon04] E. Goncalvès. *Introduction aux systèmes dynamiques et chaos*. Institution National Polytechnique de Grenoble, 2004.

[H. 06] H. Ma and C. Han. Selection of Embedding Dimension and Delay Time in Phase Space Reconstruction. *Frontiers of Electrical and Electronic Engineering in China*, 1, 2006. 54, 134

[H. 11] H. Wymeersch, F. Penna and V. Savic. Uniformly Reweighted Belief Propagation: A Factor Graph Approach. In *Proc. International Symposium on Information Theory*, 2011. 104, 158

[H. 12] H. Wymeersch, F. Penna and V. Savic. Uniformly Reweighted Belief Propagation for Estimation and Detection in Wireless Networks. *IEEE Transactions on Wireless Communications*, 11, 2012. 104, 158

[H.A35] H.A. Bethe. Statistical Theory of Superlattices. In *Proc. Royal Society of London*, 1935. 22, 77

[Hil00] R. Hilborn. *Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers*. Oxford University Press, 2000. 24, 43, 51, 128, 131, 133, 138

[I.S60] I.S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 2, 1960. 16

[J. 96] J. Hagenauer, E. Offer and L. Papke. Iterative Decoding of Binary Block and Convolutional Codes. *IEEE Transactions on Information Theory*, 42, 1996. 21

[J. 02] J. Zinn-Justin. *Quantum Field Theory and Critical Phenomena*. Oxford University Press, 2002.

[J. 03a] J. Harel, R.J. McEliece and R. Palanki. Poset Belief Propagation: Experimental Results. In *Proc. International Symposium Information Theory*, 2003.

[J. 03b] J. S. Yedidia and J.-P. Bouchaud. Renormalization Group Approach to Error-Correcting Codes. *Journal of Physics A: Mathematical and General*, 36, 2003. 38

[J.-12a] J.-C. Sibel, S. Reynal and D. Declercq. A novel region graph construction based on trapping sets for the Generalized Belief Propagation. In *Proc. Int. Conf. on Communication Systems*, 2012. 87

[J.-12b] J.-C. Sibel, S. Reynal and D. Declercq. Evidence of chaos in the Belief Propagation for LDPC codes. In *Proc. 6th Chaotic Modeling and Simulations International Conference*, 2012. 123

[J.-12c] J.-C. Sibel, S. Reynal and D. Declercq. Experimental results about the dynamics of the Generalized Belief Propagation used on LDPC codes. In *Proc. Int. Conf. on Computational Physics*, 2012. 87, 123

[J.-12d] J.-C. Sibel, S. Reynal and D. Declercq. On the Belief Propagation and its Dynamics. In *Proc. European Conference on Complex Systems*, 2012. 123

[J.-12e] J.-C. Sibel, S. Reynal and D. Declercq. On the Generalized Belief Propagation and its Dynamics. In *Proc. Int. Conf. on Control, Automation and Information Sciences*, 2012. 123

[J. 12f] J. Ren, X. Yang, L-U. Yang, Y. Xu and F. Yang. A delayed computer virus propagation model and its dynamics. *Chaos, Solitons & Fractals*, 45, 2012.

[J.-13a] J.-C. Sibel, S. Reynal and D. Declercq. Décodage des codes LDPC par le CCCP. In *Proc. GRETSI*, 2013. 79

[J.-13b] J.-C. Sibel, S. Reynal and D. Declercq. Evidence of chaos in the Belief Propagation for LDPC codes. *Chaotic Modeling and Simulations Journal*, 6, 2013. 123

[J.M71] J.M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. 1971. 65

[J.S02] J.S. Yedidia. Generalized Belief Propagation and Free Energy Minimization. In *Proc. Neural Information Processing System*, 2002. 21, 38

[J.S03a] J.S. Yedidia. *Exploring Artificial Intelligence in the New Millenium. Chapter 3: Understanding Belief Propagation and its Generalizations*. Morgan Kaufmann, 2003. 21

[J.S03b] J.S. Yedidia and J.-P. Bouchaud. Renormalization Group Approach To Error-Correcting Codes. *Journal of Physics A: Mathematical and General*, 36, 2003.

[J.S03c] J.S. Yedidia and M. Fossorier. Representing Codes for Belief Propagation Decoding. In *Proc. International Symposium on Information Theory*, 2003.

[J.S05] J.S. Yedidia, W.T. Freeman and Y. Weiss. Constructing free energy approximations and Generalized Belief Propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282–2312, 2005. 19, 22, 38, 73, 91, 92, 94, 98, 103, 110

[Kop10] Introduction to the Functional Renormalization Group, Lectures Notes in Physics - Mean Field Theory and the Gaussian Approximation. Springer, 2010. 30

[K.P99] K.P. Murphy, Y. Weiss and M.I. Jordan. Loopy belief propagation for approximate inference: an empirical study. In *Proc. Uncertainty in Artificial Intelligence*, 1999.

[K.T96] K.T. Alligood, T.D. Sauer, J.A. Yorke. *Chaos: An Introduction to Dynamical Systems*. Springer-Verlag New York, Inc., 1996. 40, 130

[Kug96] D. Kugiumtzis. State space reconstruction parameters in the analysis of chaotic time series – the role of the time window length. *Physica D*, 95, 1996. 53

[Kul68] S. Kullback. *Information Theory and Statistics*. Dover, 1968. 72

[L. 64] L. Boltzmann. *Lectures on Gas Theory*. University of California Press, 1964. 70

[L. 02] L. Kocarev, Z. Tasev and A. Vardy. Improving turbo codes by control of transient chaos in turbo-decoding algorithms. *Electronics Letters*, 38, 2002. 24

[L. 06] L. Kocarev, F. Lehmann, G.M. Maggio, B. Scanavino, Z. Tasev and A. Vardy. Nonlinear Dynamics of Iterative Decoding Systems: Analysis and Applications. *IEEE Transactions on Information Theory*, 52, 2006. 24

[Les96] A. Lesne. *Méthodes de renormalisation*. Eyrolles Sciences, 1996.

[M. 87] M. Mezard, G. Parisi and M. Virasoro. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and its Applications*. World Scientific, 1987. 28, 30

[M. 01] M. Welling and Y. W. Teh. Belief Optimization for Binary Networks: A Stable Alternative to Loopy Belief Propagation. In *Proc. 17th Conference on Uncertainty in Artificial Intelligence*, 2001.

[M. 02] M. R. Naphade, I. V. Kozintsev and T. S. Huang. A factor graph framework for semantic video indexing. *IEEE Transactions on Circuits and Systems for Video Technology*, 12, 2002. 63

[M. 04] M. Welling. On the Choice of Regions for Generalized Belief Propagation. In *Proc. 20th Conference on Uncertainty in Artificial Intelligence*, 2004. 116

[M. 05] M. Welling, T.P. Minka and Y.W. Teh. Structured Region Graphs: Morphing EP into GBP. In *Proc. 21st Conference on Uncertainty in Artificial Intelligence*, 2005.

[M. 06] M. Mezard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, 2006. 29, 33, 63

[Mano5] P. Manneville. *Dynamique non linéaire et chaos*. Summer School on Physics, 2005.

[Meao1] Advanced Mean Field Methods. Theory and Practice. MIT Press, 2001. 32, 33

[M.Jo5] M.J. Wainwright, T.S. Jaakkola and A.S. Willsky. A New Class of Upper Bounds on the Log Partition Function. *IEEE Transactions on Information Theory*, 51, 2005.

[M.Mo8] M.M. Mazilu, S. Azou and A. Serbanescu. On the reconstruction of the phase space of a dynamical system. *ROMAI Journal*, 4, 2008.

[M.T93] M.T. Rosenstein, J.J. Collins and C.J. De Luca. A practical method for calculating largest Lyapunov exponents from small data sets. *Physica D*, 65:117–134, 1993. 138

[Niso1] H. Nishimori. *Statistical physics of spin glasses and information processing an introduction*. Clarendon Press - Oxford, 2001. 21, 30, 33

[Olm11] E. Olmedo. Is there chaos in the Spanish labour market ? *Chaos, Solitons & Fractals*, 44, 2011.

[Orzo8] W. Orzeszko. The new method of measuring the effects of noise reduction in chaotic data. *Chaos, Solitons & Fractals*, 38, 2008.

[P. 90] P. Clifford. *Disorder in Physical Systems: a volume in honour of John M. Hammersley on the occasion of his 70th birthday, from G. Grimmett and D. J. A. Welsh. Markov Random Fields in Statistical Physics*. Clarendon Press, 1990. 65

[P. 03] P. Pakzad et V. Anantharam. Estimation and marginalization using Kikuchi approximation methods. *Neural Computation*, 17:1836–1873, 2003. 22, 101, 103

[P. 06] P. Pakzad and V. Anantharam. Kikuchi Approximation Method for Joint Decoding of LDPC Codes and Partial-Response Channels. *IEEE Transactions on Communications*, 54, 2006.

[P. 08] P. Castiglione, M. Falcioni, A. Lesne and A. Vulpiani. *Physique statistique. Chaos et approches multiéchelles*. Belin, 2008.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988. 18, 63

[Preo5] M. Pretti. On the Convergence of Kikuchi's Natural Iteration Method. *Journal of Statistical Physics*, 119, 2005.

[R. 51] R. Kikuchi. A Theory of Cooperative Phenomena. *Phys. Rev*, 81:988–1003, 1951. 22, 91

[R. 05] R. Santana, P. Larrañaga and J. A. Lozano. Technical Report: Properties of Kikuchi Approximations Constructed From Clique Based Decompositions. Technical report, Department of Computer Science and Artificial Intelligence, University of the Basque Country, 2005.

[R. 07a] R. Höns, R. Santana, P. Larrañaga and J.A. Lozano. Technical Report: Optimization by Max-Propagation Using Kikuchi Approximations. Technical report, Department of Computer Science and Artificial Intelligence, University of the Basque Country, 2007.

[R. 07b] R. Höns, R. Santana, P. Larrañaga and J.A. Lozano. Technical Report: Optimization by Max-Propagation Using Kikuchi Approximations. Technical report, Department of Computer Science and Artificial Intelligence, University of the Basque Country, 2007.

[R.C60] R.C. Bose and D.K. Ray-Chaudhuri. On a class of error correcting binary codes. *Information and Control*, 3, 1960. 16

[R.G63] R.G. Gallager. *Low-Density Parity-Check Codes*. PhD thesis, MIT, 1963. 15, 17, 20, 63

[R.J98] R.J. McEliece, D.J.C. MacKay, and J.-F. Cheng. Turbo decoding as an instance of Pearl's 'belief propagation' algorithm. *IEEE Journal of Selected Areas in Communications*, 16, 1998. 18

[R.Jo2] R.J. McEliece and M. Yildrim. Belief propagation on partially ordered sets. In *Proc. 15th International Symposium on Mathematical Theory of Networks and Systems*, 2002. 98

[R.Mo1] R.M. Tanner, R. Michael, D. Sridhara and T. Fuja. A Class of Group-Structured LDPC Codes, 2001.

[Rod] Rodney Gaines and Anthony Zittle. Calculating the Lyapunov Exponent Time Series Analysis of Human Gait Data.

[R.W50] R.W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29, 1950. 14, 68

[S. 06] S. Sankaranarayanan, S. K. Chilappagari, R. Radhakrishnan and B. Vasic. Failures of the Gallager B Decoder: Analysis and Applications. In *Proc. Information Theory and Applications Workshop UCSD*, 2006. 17

[Sano5] R. Santana. Estimation of Distribution Algorithms with Kikuchi Approximations. *Evolutionary Computation*, 13, 2005.

[S.E78] S.E. Newhouse, D. Ruelle and R. Takens. Occurrence of Strange Axiom A Attractors near Quasi-Periodic Flows on $T_m (m = 3$ or more). *Communications in Mathematical Physics*, 64, 1978. 48

[S.K10] S.K. Planjery, D. Declercq, S.K. Chilappagari and B. Vasic. Multilevel decoders surpassing belief propagation on the binary symmetric channel. In *Proc. International Symposium on Information Theory*, 2010. 21

[S.K11] S.K. Planjery, D. Declercq, L. Danjean and B. Vasic. Finite alphabet iterative decoders for LDPC codes surpassing floating-point iterative decoders. *Electronics Letters*, 47, 2011. 21, 85

[Ste04] Decoherence and Entropy in Complex Systems, Selected Lectures from DICE 2002 - Spin glasses: Still Complex After All These Years by D.L. Stein. Springer, 2004. 28

[S.Y01] S.Y Chung, T.J. Richardson and R.L. Urbanke. Analysis of Sum-Product Decoding of Low-Density Parity-Check Codes Using a Gaussian Approximation. *IEEE Transactions on Information Theory*, 47, 2001. NO. 2.

[T. 03a] T. Heskes. Stable fixed points of loopy belief propagation are minima of the Bethe free energy. In Becker, S. and Thrun, S. and Obermayer, K., editor, *Advances in Neural Information Processing Systems 15*, pages 359–366. MIT Press, 2003. 22, 159

[T. 03b] T. Heskes and O. Zoeter. Generalized belief propagation for approximate inference in hybrid Bayesian networks. In *Proc. 9th International Workshop on Artificial Intelligence and Statistics*, 2003.

[T. 04] T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16:2379–2413, 2004. 22

[T. 06a] T. Heskes. Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *Journal of Artificial Intelligence Research*, 26:153–190, 2006. 22, 77, 78, 79

[T. 06b] T. Ott and R. Stoop. The Neurodynamics of Belief Propagation on Binary Markov Random Fields. In *Proc. Neural Information Processing Systems*, 2006. 63

[Tan02] T. Tanaka. *Methods of Statistical Physics*. Cambridge University Press, 2002.

[T.G08] T.G. Roosta, M.J. Wainwright and S.S. Sastry. Convergence Analysis of Reweighted Sum-Product Algorithms. *IEEE Transactions on Signal Processing*, 56, 2008. 104, 158

[T.J01] T.J. Richardson and R.L. Urbanke. The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding. *IEEE Transactions on Information Theory*, 47, 2001. No. 2.

[T.J03] T.J. Richardson. Error floors of LDPC codes. In *Proc. 41st Annual Allerton Conference on Communication, Control and Computing*, 2003. 84

[T.M91] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley Interscience, 1991.

[W. 03] W. Wiegerinck and T. Heskes. Fractional Belief Propagation. In S. Becker, S. Thrun and K. Obermayer, editor, *Proc. 15th Advances in Neural Information Processing Systems*, pages 438–445. MIT Press, 2003.

[Wie05] W. Wiegerinck. Approximations with Reweighted Generalized Belief Propagation. In *Proc. 10th International Conference on Artificial Intelligence and Statistics*, 2005.

[X. 05] X. Zheng, F.C.M. Lau, C.K. Tse and S.C. Wong. Study of bifurcation behavior of LDPC decoders. *Int. Journal of Bifurcation and Chaos*, 16:3435–3449, 2005. 24, 127

[X. 06] X. Zheng, F.C.M. Lau, C.K. Tse and S.C. Wong. Techniques for Improving Block Error Rate of LDPC Decoders. In *Proc. International Symposium on Circuits and Systems*, 2006.

[Y. 08] Y. Nishiyama and S. Watanabe. Generalization of Concave and Convex Decomposition in Kikuchi Free Energy. In *Proc. 18th International Conference on Artificial Neural Networks*, 2008.