UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA

Relações entre aprendizagem dentro e fora de
equilíbrio termodinâmico

Evaldo Araújo de Oliveira Filho

*Dissertação apresentada
ao Instituto de Física da
Universidade de São Paulo
para a obtenção do título de
Mestre em Ciências*

Orientador : Prof. Dr. Nestor Felipe Caticha Alfonso

Banca Examinadora:

Prof^a Dr^a Carmen Pimentel Cintra do Prado (IF-USP)

Prof. Dr. José Fernando Fontanari (IFSC-USP)

Prof. Dr. Nestor Felipe Caticha Alfonso (IF-USP)

SÃO PAULO
2000

FICHA CATALOGRÁFICA

**Preparada pelo Serviço de Biblioteca e Informação
do Instituto de Física da Universidade de São Paulo**

Oliveira Filho, Evaldo Araújo de

Relações entre Aprendizagem Dentro e Fora de Equilíbrio
Termodinâmico. São Paulo, 2000.

Dissertação (Mestrado) - Universidade de São Paulo.
Instituto de Física - Departamento de Física Geral

Orientador: Prof. Dr. Nestor Felipe Caticha Alfonso
Área de Concentração: Física da Matéria Condensada

Unitermos:

1. Redes Neurais;
2. Dinâmica de Aprendizagem;
3. Inferência Bayesiana.

USP/IF/SBI-043/2000

“Feliz o homem que acha sabedoria, e o homem que adquire conhecimento”

Pv 3:13

Aos meus pais e irmãos.

AGRADECIMENTOS

Quero aqui agradecer inicialmente à FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo) por tornar possível a realização dessa dissertação por meio de apoio financeiro, ao Prof. Salinas por sua confiança inicial que me incentivou a vir para cá e também ao Prof. Nestor Caticha, que além de sua confiança inicial me orientou nesse trabalho com tanta paciência e amizade.

Agradeço também aos meus amigos de moradia Evandro, Gustavo e Ivan que, juntamente com meus amigos cristãos, os quais se tornaram meus irmãos, fizeram dessa estadia um verdadeiro lar para mim; aos colegas da sala de trabalho e a André pelas inúmeras ajudas e momentos divertidos; lembrando também de nossas secretárias, do Departamento de Física Geral, que me ajudaram com paciência e alegria.

Como esquecer da querida amiga Chong? Não seria possível! Por isso deixo aqui os meus agradecimentos por todos os passeios, brincadeiras, incentivos e momentos alegres que representam a sua sincera amizade por mim.

Por último quero enfatizar minha dedicatória à minha família, que tanto amo, e meus agradecimentos a Deus por ser um suporte na minha vida.

RESUMO

A aplicação da Mecânica Estatística no estudo de Redes Neurais é baseada no fato que a extração de informação de dados (exemplos) pode ser modelada por um processo de minimização de uma função energia.

Técnicas originadas no estudo de sistemas desordenados, tais como o Método de Réplicas; o Método da Cavidade; Equações de TAP; bem como técnicas de Monte Carlo tem sido exaustivamente estudadas, levando a vários resultados dentro do que temos conhecido como *aprendizagem off-line*, onde o sistema é posto em equilíbrio termodinâmico. A possibilidade do tempo de relaxação ser muito grande implica alto custo computacional, o que tem estimulado a busca por algoritmos de aprendizagem fora do equilíbrio, onde surge uma interessante classe de métodos conhecidos por *aprendizagem on-line* [1], na qual cada informação (exemplo) é apresentada ao sistema apenas uma vez, trazendo um baixo custo computacional junto a um bom desempenho.

Nessa dissertação nós trabalhamos em cima do trabalho de Oppen [2], que relacionou a aprendizagem on-line ótima à aprendizagem off-line Bayesiana por meio de uma aproximação Gaussiana da distribuição posterior. Isso porém, pode ser visto como apenas o primeiro passo numa expansão generalizada de Gram-Charlier (G-Ch) da densidade posterior, a qual pode trazer novos caminhos para o entendimento da relação on-line/off-line.

A expansão também pode ser estendida à aprendizagem por potencial, onde a distribuição de probabilidades é Gibbsiana utilizando-se de todos os termos da série, ou seja, na aprendizagem off-line. E assim a G-Ch nos permite estudar extensões não Gaussianas da aprendizagem fora do equilíbrio (on-line) para uma aprendizagem em equilíbrio (off-line), dando uma interpretação do uso das famílias dos hiperparâmetros, construídos a partir dos cumulantes da distribuição posterior, como uma incorporação de informações sobre a forma geométrica do espaço de Versões.

Apresentamos duas aplicações para os algoritmos obtidos por tais caminhos: um Perceptron unidimensional e um N-dimensional. Na primeira observamos a

aprendizagem Bayesiana na presença de ruído e quando a regra a ser aprendida muda no tempo, onde construímos um diagrama de robustez para a análise da adaptabilidade do algoritmo frente a estimação incorreta do nível de ruído.

Na segunda aplicação apresentamos resultados preliminares para o Perceptron em N dimensões. Estudamos aprendizagem usando o potencial ótimo que leva à saturação do limite de Bayes para a generalização no limite termodinâmico.

ABSTRACT

The application of Statistical Mechanics to the study of learning in Neural Networks stems from the fact that the extraction of information from data (examples) can be modeled by a dynamical process of minimization of an energy function.

Techniques originated in the study of disordered systems, such as the replica and cavity methods, TAP equations, as well as Monte Carlo techniques, have been borrowed and extended, leading to several results in what has become known as Off-line learning. Since disordered systems may take too long to equilibrate, implying a high computational cost, the search for efficient nonequilibrium learning algorithms has been undertaken. An interesting class of methods - where essentially, examples are used one at a time - is collected under the name of On-line learning [1]. These bring the possibility of efficient performance and low computational cost.

Optimal on-line learning in Neural networks, which can be described as a problem in off-equilibrium Statistical Mechanics, is related to the thermal equilibrium Bayesian off-line learning scheme. We build on work of Oppor [2], who showed a general link between off-line learning and the gaussian approximation that leads to optimal on-line learning. This is however only the first step in a cummulant type expansion and we are interested in studying its extensions. A Gram-Charlier expansion (G-Ch) of the posterior density gives a systematic way to understand the on/off-line relation.

The idea can be further extended to potential based learning, where in the full off-line version, the likelihood is a Gibbs distribution. Now the G-Ch permits studying the non-gaussian extensions of on-line off-equilibrium learning to the off-line in-equilibrium case. We give an interpretation of the use of families of hyperparameters, constructed from the cummulants of the posterior, as a means of incorporating information about the shape of a generalization of the version space.

Two applications are presented. First we study a toy model, a one-legged perceptron, and look at Bayesian learning in the presence of noise. Robustness diagrams are used to discuss the adaptability of these algorithms to misidentifications of noise

levels. In a second application we present preliminary results for the Perceptron in N dimensions. We study potencial based learning with the optimal potencial that leads to saturation of the Bayes limit.

Sumário

1	Introdução	1
1.1	Aprendizado numa rede neural.	2
1.2	Um pouco sobre inferência estatística.	5
1.3	Estrutura da dissertação.	8
2	Estimação Bayesiana	9
2.1	Parametrização Gaussiana.	12
2.2	Parametrização Hermite $\times$ Gaussiana.	19
2.2.1	Encontrando as equações de atualização.	22
2.3	Usando uma distribuição Gibbsiana.	29
3	O Perceptron unidimensional	35
3.1	Professor constante no tempo.	38
3.2	Professor com variações no tempo.	41
4	O Perceptron N-dimensional	47
5	Conclusões e Perspectivas	53
	Apêndice A: Distância de Kullback-Leibler	61
	Apêndice B: Polinômios de Hermite	63
	Apêndice C: Equações de atualização	67
	Bibliografia	71

Lista de Figuras

1.1	Componentes de um neurônio.	3
1.2	(a) Neurônio Artificial (b) Rede Neural Artificial Típica.	4
1.3	Evolução da distribuição no processo de aprendizado.	7
2.1	Aproximação e atualização da distribuição paramétrica.	13
2.2	Um típico potencial V da estabilidade λ da rede aluno.	32
3.1	Um perceptron simples.	35
3.2	Diferença de desempenho para o Perceptron entre a parametrização com polinômios até a quarta ordem e com apenas o primeiro termo da série.	38
3.3	Evolução do hiperparâmetro $c(t)$ no processo de aprendizado.	39
3.4	Desempenho da rede aluno para diferentes níveis de ruído.	39
3.5	Expoente de queda q do desvio $\langle (\hat{w} - w_0)^2 \rangle$ em função dos níveis de ruído real ϵ_0 e estimado ϵ (dados em %).	40
3.6	Professor variando harmonicamente durante o treinamento.	42
3.7	Desvio médio quadrático para um professor que varia harmonicamente durante o treinamento.	42
3.8	Rede professor w_0 com mudanças aleatórias de $\delta \leq 0, 1$ a cada exemplo.	43
3.9	Desvio médio quadrático para um professor com mudanças aleatórias.	43
3.10	Desvio médio quadrático para um professor com mudanças bruscas.	44
4.1	Representação de uma seção do espaço de entrada N-dimensinoal mostrando o vetor $\vec{w}^*$ perpendicular ao hiperplano $\mathbf{D}$, onde a linha tracejada representa sua projeção no plano da figura.	49
4.2	O plano $\vec{w}-\vec{w}^*$, onde C e D são os seus respecitvos hiperplanos perpendiculares e A e B as regiões onde $\vec{w}$ e $\vec{w}^*$ dão respostas diferentes a um mesmo exemplo.	49
4.3	Restrições no espaço de versões.	51

5.1	Diagrama de robustez em função dos níveis reais ε_0 e estimados ε de ruído na presença de ruído multiplicativo.	55
5.2	Primeira e segunda derivada do potencial on-line Bayesiano para o Perceptron.	57

1

INTRODUÇÃO

Por volta de 1960, Frank Rosenblatt e colaboradores [3] publicaram um trabalho sobre um tipo específico de rede neural, chamado por eles de *Perceptron* devido ao fato de o considerarem um modelo simplificado do mecanismo biológico do processamento de informações sensoriais (*percepção*), que chamou a atenção dos cientistas da época. Esse trabalho veio a trazer uma nova luz para os chamados estudos de *Redes Neurais Artificiais* (ANN)¹, promovidos por uma publicação de Warren McCulloch e Walter Pitts [4] em 1943, que propunha uma teoria geral do processamento de informação baseada em redes de elementos de decisão binária, que seriam os análogos aos neurônios biológicos (em sua versão bastante simplificada). Nesse contexto o aprendizado por uma rede de neurônios consiste em encontrar configurações de sinapses que gerem as saídas desejadas (ou comportamento desejado). McCulloch e Pitts mostraram que tais redes poderiam realizar quaisquer tarefas imagináveis, semelhantes aos computadores digitais ou suas abstrações matemáticas (*a Máquina de Turing*). Mas nem tudo estava perfeito, a rede neural proposta ainda possuía um problema não resolvido: como escolher as sinapses da rede de forma que tarefas específicas sejam realizadas por uma máquina?

Em 1961, Caianiello [5] construiu uma teoria estatística, usando idéias da Mecânica Estatística, e incorporou o aprendizado baseado nas propostas feitas por Hebb [6] sobre o processamento de informações no cérebro. Surgindo no ano seguinte o Perceptron apresentado pelo grupo de Rosenblatt, o qual demonstrou que esse simples e rico modelo, semelhante a programas de computador, podia generalizar tarefas!

Mesmo após aproximadamente sessenta anos, a questão levantada no trabalho de McCulloch e Pitts ainda continua sendo exaustivamente estudada, tomando am-

¹Do inglês: Artificial Neural Network.

plos caminhos abertos pelo trabalho de Rosenblatt, em diversos aspectos: biológico, matemáticos, psiconeurológicos, etc ..., entrando no que hoje conhecemos por *Teoria do Aprendizado e Ciências Cognitivas*.

Nosso trabalho aqui refere-se a mesma questão: *como escolher os valores dos pesos sinápticos para obter certas saídas desejadas?* Para responder tal pergunta nos basearemos em desenvolvimentos feitos pela Teoria de Redes Neurais Artificiais em paralelo com a Mecânica Estatística, interessando-nos levantar relações gerais para tais sistemas entre os resultados obtidos no limite de equilíbrio termodinâmico e os obtidos fora do equilíbrio.

Neste capítulo de introdução exporemos na seção 1.1 alguns conceitos e definições comuns às redes neurais, dando uma idéia mais clara do sistema em estudo. Enquanto que na seção 1.2 faremos um apanhado geral sobre *inferência estatística*, com intuito de também deixar mais claro porque a usamos. Na última seção, 1.3, apresentaremos uma descrição da estrutura dessa dissertação como um todo.

1.1 Aprendizado numa rede neural.

Numa rede neural biológica, um típico neurônio coleta sinais vindo de outros neurônios, por meio de árvores de estruturas finas chamadas *dendritos* ou até mesmo pelo próprio corpo celular (ver Fig.1.1). Tal neurônio envia impulsos elétricos por um longo tubo conhecido como *axônio*, que se divide em centenas de ramificações. No fim de cada ramificação, uma estrutura chamada *sinapse*, converte os impulsos vindos do axônio em atividades de inibição ou excitação nos neurônios conectados. Quando um neurônio recebe entradas excitantes que sejam suficientemente grandes comparadas com as inibitórias e com um certo limiar de disparo, ele envia um impulso de resposta por seu axônio. Assim, o aprendizado ocorre por mudar as relevâncias das sinapses de forma que influencie outro neurônio a dar respostas diferentes.

Semelhante ao modelo biológico, o nosso neurônio artificial (ver Fig.1.2(a)) também constará de um corpo central, dendritos, um axônio e as componentes que mais nos interessarão : as sinapses. A função das sinapses é modelada por uma relevância variável, que aqui será representada pelas variáveis w_i 's, associadas a cada conexão. Cada neurônio artificial converterá o padrão dos impulsos de entrada em um único padrão de impulso de saída. Sendo tal conversão feita em duas etapas:

Figura 1.1: Componentes de um neurônio.

1. Multiplica-se cada impulso de entrada pelo peso de sua conexão (sinapse), somando e guardando-os numa quantidade chamada de *soma*.
2. Usa-se uma função de relação *entrada-saída* que transformará a soma num único impulso de saída.

O comportamento da ANN dependerá então tanto da função de transferência *entrada-saída* como também da distribuição dos pesos de suas sinapses, que são especificados para cada neurônio. De forma que, para fazer nossa rede realizar alguma tarefa específica, devemos escolher como os neurônios serão conectados uns aos outros e distribuir os pesos das sinapses adequadamente².

O tipo mais comum de uma ANN, mostrada na Fig.1.2(b), consiste de três estruturas: uma camada de entrada por onde entram as informações externas; uma camada intermediária (camadas internas) e outra de saída. A camada de entrada, por onde entra as informações externas, é conectada às camadas internas que, por sua vez, são conectadas à camada de saída, que processa a resposta final aos dados (informações) entrados³.

O processo de aprendizado em tais redes se dá então por meio de um processo iterativo sobre as informações de entrada, onde a rede busca uma configuração que solucione o problema proposto generalizando-o. De acordo com as informações de entradas temos três tipos de aprendizado: *Supervisionado*, *Não Supervisionado* e

²As conexões determinam como é possível um neurônio influenciar outro, e o pesos das sinapses dizem quão grande será tal influência.

³Este tipo simples de rede torna-se interessante por causa das camadas internas (ou escondidas) que ficam livres para construir suas próprias respostas da entrada.

Figura 1.2: (a) Neurônio Artificial (b) Rede Neural Artificial Típica.

com Reforço. No aprendizado supervisionado as informações de entrada constam com as respostas desejadas às perguntas. Assim, temos o que chamamos de cenário *professor - aluno*, diferente da não supervisionada, onde não há um *supervisor* com as respostas desejadas. Já no caso de aprendizagem por reforço o supervisor apenas estima as respostas, pois não possui as respostas corretas.

A resposta desejada para rede neural é tratada como um vetor $\vec{\sigma}$, determinado por uma função de transferência $\mathcal{F}$ desconhecida que atua no vetor com os dados de entrada $\vec{\xi}$, geralmente de dimensão bem maior que $\vec{\sigma}$;

$$\vec{\sigma} = \mathcal{F}(\vec{\xi}),$$

onde tal mapeamento pode ser determinístico ou estocástico, de forma a possibilitar a corrupção de alguns dados por ruído.

O ato de aprender é então colocado como um problema de otimização por introduzir uma função erro, a qual mede a qualidade da aproximação da rede à função $\mathcal{F}$. Assim o processo de aprendizado implica achar uma configuração das sinapses da rede que minimize a função erro.

As diferentes maneiras como as sinapses da rede são modificadas são definidas como algoritmo de aprendizado.

O processo de interação da rede com as informações se dá por duas formas distintas; a *off-line* e a *on-line*. A classificação em *off-line* ocorre quando as informações de entrada são processadas em conjunto e, a cada nova informação, a rede revê todas as outras passadas para só então atualizar o sistema, procurando a situação

mais adequada para o conjunto inteiro de treinamento⁴. O que pode ser facilmente associado à idéia de equilíbrio termodinâmico. Em oposição temos o algoritmo que é classificado como *on-line*, onde a atualização é feita a cada nova informação que chega, sem consultar o conjunto de treinamento como um todo. O que também pode ser facilmente associado a um processo em desequilíbrio termodinâmico. Com tais associações torna-se possível a utilização das ferramentas da Mecânica Estatística no estudo de tais sistemas, visto também que o problema de aprendizado pode ser visualizado como um processo de minimização de uma função erro (energia).

A necessidade de um aprendizado *on-line* surge imediatamente quando se precisa tomar decisões rápidas, onde não se tem tempo para analisar todas as informações disponíveis. Por exemplo, podemos citar a situação quando uma pessoa está sozinha na selva e vê entre as folhas de uma moita um tecido amarelo com manchas pretas. No primeiro momento a pessoa não tem a plena certeza se aquilo que está na moita é apenas um pedaço de pano ou se a pele de uma onça, mas se ela ficar ali até ter essa certeza, e se for realmente uma onça, essa informação não lhe será mais tão útil uma vez que não conseguirá escapar do ataque da onça, por outro lado, se a pessoa correr assim que vê o tecido entre as folhas, poderá escapar. Assim vemos que em determinadas situações é preferível termos rápidos processamentos com conclusões razoáveis do que termos conclusões exatas mais com longo tempo de processamento. É claro que em tais processamentos mais rápidos (processos *on-line*) perdemos uma certa quantia de informação que não foi utilizada, mas surpreendentemente recentes estudos [7] mostram que algoritmos baseados em aprendizagem *on-line* podem encontrar um desempenho semelhante aos baseados na *off-line*, quando o número de neurônios é consideravelmente grande ($N \rightarrow \infty$).

1.2 Um pouco sobre inferência estatística.

Nesta seção nós pretendemos rever brevemente alguns conceitos de inferência estatística. Ressaltando específicas características que irão aparecer no decorrer dos próximos capítulos.

Em inferência estatística o conhecimento sobre qualquer sistema é representado por um conjunto de variáveis $w = \{w_1, w_2, \dots, w_K\}$, que podem ser discretas ou contínuas, estando cada possível configuração, ou ponto, de w no espaço Ω . A

⁴Que consta de todas as informações já utilizadas mais a que chega a cada instante.

partir disso consideramos que o conhecimento de um estado qualquer do sistema é definido por meio de uma distribuição de probabilidades P sobre o espaço Ω . De forma que encontramos dois diferentes problemas:

- dado um estado (uma distribuição P) e alguma nova informação (um conjunto de restrições y sobre w), como inferir um novo estado (uma distribuição Q) baseado na nova informação;
- dado um estado (uma distribuição P) e os valores de algumas variáveis (w_i) do conjunto w como inferir os valores mais prováveis (de acordo com P) para as demais variáveis (w_j).

O primeiro problema é conhecido como *inferência dinâmica*, pois concerne em como a distribuição P , o estado, muda com a chegada da nova informação, enquanto que o segundo é conhecido por *inferência estática*, pois trata da consistência do estado num dado tempo t .

Aqui estamos interessados no primeiro tipo de problema: inferência dinâmica, pois esse tipo de problema é facilmente identificado como um modelo para o aprendizado por experiência, no sentido de ser uma maneira de evoluir um dado estado baseado na nova informação que chega. O problema de inferência estática, por sua vez, está relacionado a como retirar informações que foram armazenadas durante o aprendizado (como no caso de reconhecimento de padrões ...).

O objetivo da inferência dinâmica é então: dado um conjunto de variáveis w no espaço Ω , uma distribuição *a priori* P e alguma nova informação y , encontrar uma distribuição posterior Q que inclua o efeito de considerar y .

A forma consistente de incluir a informação contida no conjunto de dados w está baseada no teorema de Bayes. Seja w o conjunto de variáveis que queremos determinar, que no caso específico de uma rede neural será o vetor de pesos, a probabilidade conjunta dos pesos e dos dados pode ser escrita de duas formas equivalentes:

$$\begin{aligned} P(w, y) &= P(w)P(y|w) \\ &= P(y)P(w|y). \end{aligned} \tag{1.1}$$

Portanto temos o teorema de Bayes:

$$P(w|y) = \frac{P(w)P(y|w)}{\int P(w')P(y|w')dw'}.$$

É importante enfatizar os nomes que serão usados durante esta dissertação:

- $P(w)$: a distribuição prévia (*a priori*) de probabilidades. Inclui quaisquer informações que tenhamos sobre o problema antes de termos acesso aos dados ou informação contida nos exemplos...
- $P(y|w)$: Se os parâmetros ou pesos tivessem efetivamente o valor w , qual seria a probabilidade de que os dados y fossem observados. Este termo inclui por sua vez duas coisas. Por um lado a distribuição dos exemplos, e por outro a arquitetura da máquina.
- $P(w|y)$: A distribuição *posterior*. É a distribuição após ter incluído a informação contida nos dados. Note que pode ser considerada, em um instante posterior como a nova distribuição *a priori* , antes da inclusão de nova informação.

Mostramos esquematicamente na figura 1.3 como evolui a distribuição posterior após a inclusão de nova informação num caso em duas dimensões. Na falta de evidência *a priori* que dê mais importância a um valor que outro dos valores dos pesos (neste caso w_1 e w_2) usamos o princípio de Laplace de razão insuficiente, e começamos com uma distribuição *a priori* uniforme (fig. 1.3a). Suponhamos agora que um dado novo carrega informação sobre w_1 . A posterior a essa informação será algo como a figura 1.3b. Essa é a distribuição *a priori* antes do segundo dado. Suponhamos que outro traga informação sobre w_2 . Teremos a posterior mostrada na figura 1.3c. Talvez após a inclusão de vários dados, a posterior se pareça com a figura 1.3d. Estas idéias serão discutidas com muito mais detalhe nos capítulos que seguem.

Figura 1.3: Evolução da distribuição no processo de aprendizado.

1.3 Estrutura da dissertação.

O nosso objetivo nessa dissertação é estudar a relação da aprendizagem fora de equilíbrio com a aprendizagem no equilíbrio na Teoria de Redes Neurais. Para isso iremos utilizar um procedimento baseado em inferência Bayesiana que, por sua própria construção, descreve a aprendizagem dentro do equilíbrio termodinâmico, ou seja, aprendizagem off-line, interessando-nos levantar características gerais e novas informações sobre tais sistemas. Não estamos apenas em busca de novos algoritmos que sejam mais eficientes e menos custosos, mas principalmente a procura de esclarecimentos de cenários que se repetem em diferentes técnicas, tais como do método de réplicas e do método da cavidade, dentre outros.

Nas seções anteriores expomos alguns conceitos sobre a Teoria de Redes Neurais e um pouco sobre Inferência Estatística que nos serão úteis no decorrer da dissertação, ou melhor, que serão o nosso ponto de partida.

No capítulo 2 apresentamos a teoria proposta para seguir o caminho on-line/off-line. Iremos descrever uma base de funções tipo uma expansão de Gram-Charlier, semelhante à base de funções utilizadas no problemas do oscilador hârmônico em Mecânica Quântica. A idéia é expandir a distribuição posterior $P(w|D)$ nessa base e observar o efeito de truncamento em ordens cada vez maiores, ou seja, o efeito de considerar cada vez mais termos na expansão, onde a utilização de todos os termos corresponde ao aprendizado off-line. De forma que na seção 2.1 apresentamos o algoritmo obtido apenas com o primeiro termo da expansão, que equivale ao trabalho feito por Oppen [2], seguindo para a inclusão de mais termos na seção 2.2 e introduzindo uma função potencial na seção 2.3.

No capítulo 3 apresentamos uma aplicação em uma dimensão para a teoria desenvolvida na seção 2.2 e no capítulo 4 uma aplicação em N-dimensões referente ao desenvolvimento feito por introduzir uma função potencial, onde infelizmente não conseguimos aprontar a tempo a simulação contendo o acréscimo dos demais termos além da Gaussiana pura, que corresponde ao primeiro termo da expansão.

No capítulo 5 fazemos um apanhado geral dos resultados obtidos, tantos os analíticos como os computacionais, e apresentamos nossas conclusões e perspectivas com relação aos possíveis novos caminhos abertos. E por último colocamos apêndices com algumas demonstrações de procedimentos utilizados e o próprio algoritmo referido no capítulo 4.

2

ESTIMAÇÃO BAYESIANA

Numa estimação Bayesiana considera-se que o conhecimento sobre qualquer grandeza possa ser representado por uma distribuição de probabilidades dos eventos observados no sistema em estudo, de forma que aprender uma função de transferência $\mathcal{F}$ que faça o mapeamento entre os dados que entram na rede e suas respectivas saídas consiste em aprender a distribuição que gerou tais dados, de entrada e saída. Também devemos transformar nosso conhecimento inicial sobre o sistema, ou seja, qualquer tipo de informação que já conhecemos antes mesmo de começar o processo de aprendizado (condições iniciais, espaço de atuação, vínculos...), numa distribuição prévia (*a priori*) que será o nosso ponto de partida, o que chama a atenção a saber se a verdadeira distribuição que gera os dados do treinamento de aprendizado está contido na mesma família que a distribuição prévia e quais são as consequências disso no desempenho do aprendizado. Assim o processo de aprendizado pode ser descrito pela evolução de uma distribuição prévia $p(\vec{w})$, definida pelo que sabemos de antemão sobre o sistema em estudo, numa distribuição posterior (*a posteriori*) que incorpore a informação dada no conjunto de treinamento $D_t = (y_1, y_2, \dots, y_t)$, que para o caso de aprendizado supervisionado pode ser tratado como sendo pares entrada-saída $y_k = (\vec{\xi}_k, \vec{\sigma}_k)$ e para aprendizados não supervisionados ou por reforço $y_k = \vec{\xi}$. Considera-se que os dados D_t são gerados independentemente, de uma maneira aleatória, de acordo com uma distribuição¹:

$$P(D_t|\vec{w}) = \prod_{k=1}^T P(y_k|\vec{w}). \quad (2.1)$$

E, no nosso caso, os parâmetros $\vec{w}$ representarão os pesos sinápticos da rede neural, $\vec{\xi}$ as informações que entram na rede e σ a resposta dada pela rede professor às

¹Onde $P(D_t|\vec{w})$ caracteriza a estrutura e o nível de ruído do sistema.

entradas $\vec{\xi}$.

Essa distribuição posterior, construída por meio da *regra de Bayes*: o produto normalizado da distribuição prévia pela distribuição de probabilidades dos dados D_t ,

$$p(\vec{w}|D_t) = \frac{p(\vec{w})P(D_t|\vec{w})}{\int p(\vec{w}')P(D_t|\vec{w}')d^N w'}, \quad (2.2)$$

quantifica a plausibilidade de cada possível configuração dos pesos sinápticos $\vec{w}$, jogando como uma regra crucial na predição dos novos dados. Uma vez construída a distribuição posterior 2.2 a estimativa $\hat{w}_i$ das sinapses da rede é obtida por:

$$\hat{w}_i = \int w_i p(\vec{w}|D_t) d^N w. \quad (2.3)$$

Predições baseadas nesse procedimento (Bayesiano) são consideradas ótimas no sentido de produzir um erro médio de predição mínimo². Lembrando que tal otimidade considera serem corretas as informações dadas *a priori*.

A pergunta que queremos analisar concerne com: o que fazer na presença de nova informação? Isto é, como modificar a distribuição posterior em face a novos dados? A resposta para tal pergunta é simples! De acordo com a eq. 2.1, a eq. 2.2 para a distribuição $p(\vec{w}|D_{t+1})$ pode ser escrita em função da distribuição anterior $p(\vec{w}|D_t)$:

$$p(\vec{w}|D_{t+1}) = \frac{p(\vec{w})P(y_{t+1}|\vec{w})P(D_t|\vec{w})}{\int p(\vec{w}')P(y_{t+1}|\vec{w}')P(D_t|\vec{w}')d^N w'}, \quad (2.4)$$

que utilizando novamente a eq. 2.2 em D_t torna-se

$$p(\vec{w}|D_{t+1}) = \frac{p(\vec{w}|D_t)P(y_{t+1}|\vec{w})}{\int p(\vec{w}'|D_t)P(y_{t+1}|\vec{w}')d^N w'}, \quad (2.5)$$

mas, como pode ser visto, tal procedimento é intrinsecamente off-line, pois requer o conhecimento do conjunto inteiro de dados D_t para atualizar a distribuição da eq. 2.5, que pode tornar-se uma distribuição de difícil integração tornando a estimação dos parâmetros $\vec{w}$ um problema computacionalmente difícil³, e da mesma maneira a predição da saída $\vec{\sigma}$ para uma nova entrada y_{t+1} . Contudo, suponha que esses parâmetros tenham sido razoavelmente estimados para o conjunto de treinamento

²A média é feita sobre todos os possíveis pontos dentro do espaço definido pela distribuição prévia.

³Como no caso em que o sistema tem uma estrutura tipo *vidros de spin*.

D_t , então a incorporação da informação no novo exemplo y_{t+1} pode ser eficiente e facilmente feita ao menos de uma forma aproximada. Usar tal procedimento é fazer um algoritmo de aprendizagem *on-line*, enquanto que o uso da eq. 2.5 em sua íntegra nos retornará um algoritmo de aprendizagem *off-line*. Então a pergunta que imediatamente surge é: como fazer tal *estimação razoável?*, ou melhor, como adaptá-lo à um processo onde apenas o novo dado y_{t+1} e pouca coisa mais seja suficiente para atualizar todos os parâmetros da rede em jogo, superando a dificuldade computacional?

Responder tal pergunta é justamente o nosso objetivo, não apenas para obter um algoritmo *on-line*, mas sim como obter um algoritmo que esteja intimamente ligado com a sua fonte, o algoritmo *off-line*, de forma que tomando limite de certas quantidades possamos recuperá-lo em sua íntegra, obtendo-se de forma mais natural os parâmetros que sejam realmente relevantes hierarquicamente a medida que nos aproximemos da distribuição completa, eq. 2.5, para que com isso possamos traçar um caminho *off-line* $\rightarrow$ *on-line* mais claramente, expondo a natureza de tais sistemas e suas implicações. Dito isso, a primeira idéia que se pode pensar é expandir a distribuição $p(\vec{w}|D_{t+1})$ numa base completa e ver os efeitos de truncar a série em ordens cada vez maiores. O que também nos dá um grande leque de possibilidades: *qual base de funções enquadra-se melhor?* Uma boa dica é que comecemos por uma base que seja descrita em termos dos momentos da distribuição a ser expandida, mas que também tenha uma rápida convergência, de forma que possamos obter a ordem hierárquica de relevância que desejamos.

Com base nesse mesmo procedimento M. Oppor [8][2] mostrou para sistemas com pesos sinápticos reais que se parametrizarmos a distribuição posterior por uma Gaussiana⁴ encontraremos um algoritmo *on-line* assintoticamente ótimo. Porém mais surpreendente ainda é que com parametrização Gaussiana, mas considerando uma distribuição Gibbsiana, pode-se mostrar [9] que para o caso particular do potencial do Perceptron de Rosenblatt encontra-se a mesma forma funcional ótima obtida pelo método variacional por Kinouchi e Caticha [10][11]. O que sugere fortemente que os termos de nossa base tenham um comportamento assintótico Gaussiano ao menos para sistemas onde os pesos sinápticos $\vec{w}$ estejam distribuídos nos eixos reais positivos e negativos. Sendo também interessante observar a atuação do *Teorema*

⁴A escolha de uma Gaussiana está intimamente ligada com o fato dos pesos sinápticos tomarem valores reais negativos e positivos, se por exemplo os pesos fossem binários ± 1 escolheríamos outra função.

Central do Limite em determinados sistemas. Assim, resolvemos utilizar uma base onde os termos são descritos por produtos de polinômios de Hermite com Gaussianas. O que traz junto consigo uma rica álgebra, que se encontra no problema do oscilador harmônico, a ser explorada.

Na próxima seção 2.1 descreveremos o resultado de parametrizarmos a função posterior por apenas uma Gaussianas, semelhantemente ao trabalho do Oppen [2] e que pode ser visto como uma primeira aproximação considerando apenas o primeiro termo da expansão de 2.5. Seguindo pela expansão formal na seção 2.2, onde definiremos nossa base e iremos a termos de ordem maiores, descrevendo por último na seção 2.3 o resultado de considerar-se a distribuição eq. 2.1 como sendo Gibbsiana.

2.1 Parametrização Gaussianas.

Como dito anteriormente, nossa idéia é aproximar $p(\vec{w}|D_t)$ por uma simples distribuição $p(par)$, onde par será um pequeno conjunto de parâmetros capaz de recolher grande parte da informação sobre os dados anteriores e que serão atualizados a cada passo. Assim, podemos resumir o procedimento em dois passos (Fig.2.1):

1. Adicionar um exemplo: a atual distribuição posterior é atualizada exatamente como a regra de Bayes (eq.2.5):

$$p(\vec{w}|par_t, y_{t+1}) = \frac{p(\vec{w}|par_t)P(y_{t+1}|\vec{w})}{\int p(\vec{w}'|par_t)P(y_{t+1}|\vec{w}')d^N w'}. \quad (2.6)$$

2. Aproximar: a distribuição posterior atualizada é parametrizada:

$$p(\vec{w}|par_t, y_{t+1}) \rightarrow p(\vec{w}|par_{t+1}). \quad (2.7)$$

A escolha de par deve ser feita de forma a diminuir ao máximo a perda de informação na parametrização. Perda essa que pode ser medida por meio da *distância de Kullback-Leibler* (KL):

$$D_{KL}(p(\vec{w}|par_t, y_{t+1})||p(\vec{w}|par_{t+1})) = \int p(\vec{w}|par_t, y_{t+1}) \ln \left(\frac{p(\vec{w}|par_t, y_{t+1})}{p(\vec{w}|par_{t+1})} \right).$$

Ao considerarmos uma parametrização por uma Gaussianas já garantimos a miniza-

Figura 2.1: Aproximação e atualização da distribuição paramétrica.

ção da distância de KL entre $p(\vec{w}|par_t, y_{t+1})$ e $p(\vec{w}|par_{t+1})$, impondo que elas tenham os mesmos dois primeiros momentos⁵ (ver apêndice A).

Além disso, para que tenhamos consistência, é necessário que a distribuição inicial $p(\vec{w}|par_0)$ não exclua nenhum valor de $\vec{w}$ para qual $p(\vec{w})$ seja diferente de zero. A escolha natural $p(\vec{w}|par_0) = p(\vec{w})$ preenche tal condição.

Então o segundo passo (eq.2.7) resume-se em fazer a projeção:

$$\begin{aligned} p(\vec{w}|par_t, y_{t+1}) &\Rightarrow p(\vec{w}|par_{t+1}) \\ &= cte \cdot \exp\left(-\frac{1}{2}(\vec{w} - \hat{\vec{w}}(t+1))\mathbf{C}^{-1}(t+1)(\vec{w} - \hat{\vec{w}}(t+1))\right), \end{aligned}$$

a qual consistirá em estimar os novos valores para os pesos sinápticos e covariâncias $\vec{w}$ e $\mathbf{C}^{-1}$ respectivamente:

$$\hat{w}_i(t+1) = \int w_i p(\vec{w}|par_t, y_{t+1}) d^N w, \quad (2.8)$$

$$C_{ij}(t+1) = \int (w_i - \hat{w}_i(t+1))(w_j - \hat{w}_j(t+1)) p(\vec{w}|par_t, y_{t+1}) d^N w, \quad (2.9)$$

que tomando $\vec{u}$ como as flutuações em torno de $\hat{\vec{w}}$: $\vec{u} = \vec{w} - \hat{\vec{w}}(t)$, e usando a eq.2.6 tornará a primeira delas em:

⁵Que para o primeiro momento significa: $\langle w \rangle_{p(par_{t+1})} = \langle w \rangle_{p(par_t, y_{t+1})}$

$$\begin{aligned}
\hat{w}_i(t+1) &= \frac{\int w_i e^{-\frac{1}{2}(\vec{w}-\hat{\vec{w}}(t))\mathbf{C}^{-1}(t)(\vec{w}-\hat{\vec{w}}(t))} P(y_{t+1}|\vec{w}) d^N w}{\int e^{-\frac{1}{2}(\vec{w}-\hat{\vec{w}}(t))\mathbf{C}^{-1}(t)(\vec{w}-\hat{\vec{w}}(t))} P(y_{t+1}|\vec{w}) d^N w}, \\
&= \hat{w}_i(t) + \frac{\int u_i e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}(t)\vec{u}} P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) d^N u}{\int e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}(t)\vec{u}} P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) d^N u}. \quad (2.10)
\end{aligned}$$

Agora, observe que

$$\begin{cases} u_i = \sum_l u_l \delta_{i,l} = \sum_{l,j} C_{i,j} C_{j,l}^{-1} u_l \\ \sum_l C_{i,l}^{-1} u_l e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}} = -\partial_{u_i} e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}}, \end{cases}$$

ou seja,

$$u_i e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}} = -\sum_l C_{i,l} \partial_{u_l} e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}}.$$

Usando essas equivalências na eq. 2.10 e integrando-a por partes, uma vez que a distribuição $P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t))$ seja diferenciável, encontraremos:

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \sum_l C_{il}(t) \frac{\int e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}(t)\vec{u}} \partial_{u_l} P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) d^N u}{\int e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}(t)\vec{u}} P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) d^N u},$$

resultando, ao usarmos o fato que $\partial_{u_l} f(\vec{w} + \vec{u}) = \partial_{\hat{w}_l} f(\vec{w} + \vec{u})$, no algoritmo on-line:

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \sum_l C_{il}(t) \partial_{\hat{w}_l(t)} \ln \langle P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) \rangle. \quad (2.11)$$

Com $\langle \dots \rangle$ significando a média com respeito à uma distribuição gaussiana com média nula e covariância $\mathbf{C}(t)$.

Para terminarmos o nosso algoritmo precisamos apenas achar uma equação de atualização para a matriz de covariância $\mathbf{C}$. Para isso, seguindo o mesmo caminho anterior, escrevamos a eq. 2.9 como:

$$C_{ij}(t+1) = \int (u_i - \Delta \hat{w}_i(t+1))(u_j - \Delta \hat{w}_j(t+1)) p(\vec{u} + \hat{\vec{w}}|par_t, y_{t+1}) d^N u,$$

sendo $\Delta \hat{w}_i(t+1) = \hat{w}_i(t+1) - \hat{w}_i(t)$ e dado pela eq. 2.11, que usando a identidade:

$$u_i u_j e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}} = C_{ij} e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}} + \sum_{k,l} C_{ik} C_{jl} \partial_{u_k} \partial_{u_l} e^{-\frac{1}{2}\vec{u}\mathbf{C}^{-1}\vec{u}},$$

e integrando duas vezes por partes resultará em:

$$C_{ij}(t+1) = C_{ij}(t) + \sum_{k,l} C_{ik}(t) C_{jl}(t) \partial_{\hat{w}_k} \partial_{\hat{w}_l} \ln \langle P(y_{t+1}|\vec{u} + \hat{\vec{w}}(t)) \rangle. \quad (2.12)$$

As eqs. 2.11 e 2.12 formam então o nosso algoritmo de aprendizado on-line Bayesiano, que por outro lado pode ser visto como uma primeira aproximação a um algoritmo Bayesiano off-line, como indicaremos na próxima seção. Mas antes é interessante notar que tal algoritmo de aprendizado se assemelha ao algoritmo de descida do gradiente⁶, onde a atualização dos pesos sinápticos é dada por [12]

$$\hat{w}_i(t+1) = \hat{w}_i(t) - \eta(t) \partial_{\hat{w}_i} \sum_{k=1}^t E_t(y_k | \hat{\vec{w}}(t)), \quad (2.13)$$

sendo a energia $E_t(y_k | \hat{\vec{w}}(t))$ associada a uma função erro e ajustada por $\eta(t)$, que controla o tamanho de cada nova correção nas estimativas dos w_i 's e que pode ser constante ou decrescente com t de forma a ajudar a convergência assintótica do algoritmo. Por comparar a eq. 2.13 com a eq. 2.11 assimilamos a energia $E_t(y_k | \hat{\vec{w}}(t))$ ao $\ln \langle P(y_{t+1} | \vec{u} + \hat{\vec{w}}(t)) \rangle$, salientando a média sobre a distribuição posterior da nova informação e também que no algoritmo Bayesiano o ajuste agora é tensorial e possui uma dinâmica própria, eq. 2.12, que surge naturalmente no desenvolvimento do algoritmo ao contrário do ajuste η que é posto a mão no algoritmo de descida do gradiente.

A necessidade de um ajuste decrescente na correção dos pesos w 's com o aprendizado já foi tratado em varios trabalhos: [1], [13] ... Tal necessidade vem do fato de uma vez que uma estimação dos w 's nos leva próximo a região do seu espaço de configurações que soluciona o problema, a correção Δw deve diminuir para que não sejamos jogados longe de tal região, o que pode ser causado por informações y_{t+1} corrompidas por algum tipo de ruído.

Façamos agora um rápido estudo sobre o comportamento do algoritmo descrito pelas eqs. 2.11 e 2.12 para longo tempo. Para começar vejamos a forma assintótica da matriz de covariância C .

Definindo $V_{kl} = \partial_{\hat{w}_k} \partial_{\hat{w}_l} \ln \langle P(y_{t+1} | \vec{u} + \hat{\vec{w}}(t)) \rangle$ e considerando que para grandes tempos ' t ' as mudanças temporais da matriz C são pequenas, poderemos introduzir uma evolução temporal contínua para C pela equação:

$$\frac{dC}{dt} = CVC, \quad (2.14)$$

que pode ser reescrita como

$$\frac{dC^{-1}}{dt} = -V.$$

⁶Gradient descent algorithm.

Integrando no tempo essa última equação encontraremos:

$$C^{-1}(t) - C^{-1}(t_0) = - \int_{t_0}^t V(t') dt'. \quad (2.15)$$

Para prosseguir consideraremos que a distribuição $Q(y)$ que gerou o conjunto D_t não pertence necessariamente à mesma família de $P(y|\vec{w})$, ou seja a rede aluno não compartilha da mesma família de distribuição que a rede professor.

Devido a ausência de métodos gerais para o cálculo explícito de uma convergência global para quaisquer algoritmos on-line iremos nos limitar à situação na qual o sistema (a rede neural) está próximo a um ponto fixo atrativo $\vec{w}^*$, o que nos possibilitará obter uma expressão exata para a convergência assintótica do algoritmo. O ponto fixo $\vec{w}^*$ corresponde a um mínimo local da distância entre $Q(y)$ e $P(y|\vec{w})$ dada por

$$D_{KL}(Q(y)||P(y|\hat{\vec{w}})) = \int Q(y) \ln \left(\frac{Q(y)}{P(y|\hat{\vec{w}})} \right) dy,$$

ou seja, satisfaz a igualdade:

$$\int Q(y) \partial_{\hat{w}_i} \ln P(y|\hat{\vec{w}}) dy = 0. \quad (2.16)$$

Levando em consideração que para longo tempo 't' a distribuição posterior $P(\hat{\vec{w}}|D_t)$ estará fortemente concentrada em torno de seu máximo em $\vec{w}^*$, podemos negligenciar a sua largura de forma a termos $\langle P(y|\hat{\vec{w}}) \rangle$ equivalente à $P(y|\vec{w}^*)$. Assim, dividindo a eq. 2.15 por t e tomando o limite $t \rightarrow \infty$ teremos:

$$\begin{aligned} \lim_{t \rightarrow \infty} \frac{C_{ij}^{-1}(t)}{t} &= \lim_{t \rightarrow \infty} -\frac{1}{t} \int_{t_0}^t \partial_{\hat{w}_i} \partial_{\hat{w}_j} \ln P(y|\hat{\vec{w}}) \Big|_{\vec{w}^*} dt \\ &= - \int Q(y) \partial_{\hat{w}_i} \partial_{\hat{w}_j} \ln P(y|\hat{\vec{w}}) \Big|_{\vec{w}^*} dy. \end{aligned} \quad (2.17)$$

desde que o sistema seja *ergódico* para para que possamos passar de uma média em t para uma média em y na eq. 2.17.

A eq. 2.17 nos mostra que para tempos suficientemente longos as flutuações da matriz C podem ser negligenciadas e encontramos que $C \simeq [f(\cdot)]^{-1}/t$, ou seja, uma vez que estivermos numa região que satisfaça as informações do conjunto D_t o nosso termo de correção Δw será cada vez menos significativo, decaindo a zero e garantindo assim que não escaparemos da região de solução do espaço de configuração para os

pesos sinápticos. Sendo também interessante notar que o decaimento por $1/t$ é comumente usado no algoritmo de gradiente decrescente [13] e que quando $Q(y) = P(y|\vec{w}^*)$ teremos

$$\lim_{t \rightarrow \infty} \frac{C_{ij}^{-1}(t)}{t} = \int dy P(y|\vec{w}^*) \partial_i \ln P(y|\vec{w}^*) \partial_j \ln P(y|\vec{w}^*). \quad (2.18)$$

Onde estamos usando uma notação mais simplificada para representar as derivadas com respeito à $\hat{w}_k$ e calculadas em $\vec{w}^*$, e que pela eq. 2.16 temos

$$\begin{aligned} - \int P(y|\vec{w}^*) \partial_i \partial_j \ln P(y|\vec{w}^*) dy &= \int \partial_i P(y|\vec{w}^*) \partial_j \ln P(y|\vec{w}^*) dy \\ &= \int \frac{1}{P(y|\vec{w}^*)} \partial_i P(y|\vec{w}^*) \partial_j P(y|\vec{w}^*) dy, \end{aligned}$$

que por sua vez é idêntico ao termo do lado direito da eq. 2.18 que será representado por $J_{ij}(\vec{w}^*)$.

A matriz $J_{ij}(\vec{w}^*)$ é justamente a *matriz de informação de Fisher*, de forma que assintoticamente o ajuste tensorial em 2.11 obtido pelo algoritmo Bayesiano on-line torna-se proporcional ao *gradiente natural* [1] se o modelo probabilístico $P(y|\vec{w})$ é corretamente escolhido e se o ponto fixo $\vec{w}^*$ é realmente o parâmetro usado pela rede professor.

Vejam agora o comportamento assintótico para o erro de estimação dos w 's, definido como a diferença entre $\hat{\vec{w}}$ e $\vec{w}^*$. Para isso consideraremos novamente que o parâmetro estimado $\hat{w}$ está próximo ao $\vec{w}^*$ e que a distribuição posterior está centrada em $\hat{\vec{w}}$ estreitamente. Com isso, vamos expandir $\partial_l \ln P(y|\hat{\vec{w}})$ em torno de $\vec{w}^*$, considerando os termos de até primeira ordem em $\epsilon_i(t) = \hat{w}_i(t) - w_i^*$:

$$\partial_l \ln P(y|\hat{\vec{w}}) = \partial_l \ln P(y|\hat{\vec{w}}) \Big|_{\vec{w}^*} + \sum_k \partial_k \partial_l \ln P(y|\hat{\vec{w}}) \Big|_{\vec{w}^*} \epsilon_k(t) + \dots$$

o que nos levará a

$$\Delta \hat{w}_i(t) = \Delta \epsilon_i(t) = \sum_l C_{il} \partial_l \ln P(y|\vec{w}^*) + \sum_{lk} C_{il} \partial_k \partial_l \ln P(y|\vec{w}^*) \epsilon_k(t). \quad (2.19)$$

Agora é só tirarmos a média da eq. 2.19 com relação ao conjunto D_{t+1} , que será representado por uma barra superior, e usarmos do mesmo argumento utilizado na eq. 2.14 para passarmos de tempos discretos para contínuos ($\Delta \rightarrow d/dt$).

$$\begin{aligned}
\frac{d}{dt} \langle \epsilon_i(t) \rangle_{D_{t+1}} &= \frac{d\bar{\epsilon}_i}{dt} \\
&= \sum_l C_{il} \overline{\partial_l \ln P(y|\vec{w}^*)} + \sum_{lk} C_{il} \langle \epsilon_k(t) \rangle_{D_t} \langle \partial_k \partial_l \ln P(y|\vec{w}^*) \rangle_{y_{t+1}} \\
&= \sum_l \frac{\mathcal{J}_{il}^{-1}}{t} \overline{\partial_l \ln P(y|\vec{w}^*)} - \frac{\bar{\epsilon}_i}{t},
\end{aligned} \tag{2.20}$$

que é válido no limite $t \rightarrow \infty$ e onde definimos a matriz

$$\mathcal{J}_{il} = - \int Q(y) \partial_i \partial_l \ln P(y|\vec{w}^*) dy. \tag{2.21}$$

Devido a eq. 2.16 vemos que o primeiro termo do lado direito da eq. 2.20 desaparece, de forma que o erro linear médio $\bar{\epsilon}_i$ decai com $1/t$, ou seja, o valor médio da estimativa dos pesos sinápticos pela rede aluno $\bar{w}_i$ tende com o passar do tempo ao ponto fixo w_i^* que satisfaz a eq. 2.16.

Por último, vejamos agora quão eficiente é tal algoritmo descrito pelas eqs. 2.11 e 2.12, segundo o critério da *desigualdade de Rao-Cramér* que limita a velocidade de aproximação do parâmetro estimado $\hat{w}_i$ a w_i^* :

$$\langle (\hat{w}_i - w_i^*)(\hat{w}_j - w_j^*) \rangle \geq \frac{1}{t} J_{ij}^{-1}. \tag{2.22}$$

Estimadores que satisfazem a eq. 2.22 com uma igualdade são considerados eficientes, de forma que um requerimento razoável para um bom algoritmo é que cumpra tal igualdade quando $t \rightarrow \infty$, ou seja, que seja assintoticamente eficiente. De forma que multiplicando a eq. 2.19 por $\epsilon_j(t)$ e tirando em seguida a média sobre D_t , chegaremos à:

$$\begin{aligned}
\frac{d}{dt} \mathcal{O} &= C\mathcal{K}C - C\mathcal{J}\mathcal{O} - \mathcal{O}\mathcal{J}C \\
&= \frac{1}{t^2} \mathcal{J}^{-1} \mathcal{K} \mathcal{J}^{-1} - \frac{2}{t} \mathcal{O},
\end{aligned} \tag{2.23}$$

onde definimos

$$\mathcal{O}_{ij} = \langle \epsilon_i(t) \epsilon_j(t) \rangle_{D_t}; \tag{2.24}$$

$$\mathcal{K}_{ij} = \int Q(y) \partial_i \ln P(y|\vec{w}^*) \partial_j \ln P(y|\vec{w}^*) dy. \tag{2.25}$$

A eq. 2.23 é resolvida por

$$\mathcal{O}_{ij} = \frac{1}{t} (\mathcal{J}^{-1} \mathcal{K} \mathcal{J}^{-1})_{ij}, \tag{2.26}$$

para $t \rightarrow \infty$, ou seja, para um mínimo local $\vec{w}^*$ de $-\int Q(y)P(y|\vec{w})dy$ sempre teremos o decaimento ótimo de $\propto 1/t$ para o erro! E quando o modelo é bem especificado, $Q(y) = P(y|\vec{w})$, a eq. 2.26 se transformará em

$$\mathcal{O}_{ij} = \frac{1}{t} J_{ij}^{-1}(\vec{w}^*),$$

ou seja, teremos uma eficiência assintótica segundo o critério de Rao-Cramér, eq. 2.22. Sendo também interessante citar que a eq. 2.26 possui o mesmo fator de decaimento para algoritmos Bayesianos off-line obtido por Amari e Murata [14].

2.2 Parametrização Hermite $\times$ Gaussiana.

Como mencionamos no início desse capítulo a nossa intenção agora é ir além da aproximação Gaussiana, por meio de uma expansão direta na distribuição posterior, mas escolhendo uma base que tenha um comportamento assintótico Gaussiano. Em [15] Yang e Amari usaram uma expansão unidimensional truncada numa série de Gram-Charlier para estudar algoritmos para uma separação *cega*, descrevendo os termos da série pelos momentos da distribuição expandida. A nossa proposta é seguir a mesma idéia mas para o caso de dimensões maiores que um, onde usaremos um tratamento para espaços multidimensionais utilizando polinômios de Hermite estudado por Hagedorn [16] para análises em mecânica quântica semi-clássica.

Aqui consideraremos que letras maiúsculas ($M, L, \dots$) denotarão multi-índices, $M = \{m_1, m_2, \dots, m_N\} \in \mathcal{Z}_+^N$, $\vec{a}$ e $\vec{x} \in \mathfrak{R}^N$ e A uma matriz real simétrica e regular $N \times N$. Também definimos:

$$\begin{cases} |M| = \sum_{n=1}^N m_n, \\ M! = \prod_{n=1}^N m_n!, \\ D_x^M = \partial^{|M|} / \prod_{n=1}^N (\partial_{x_n})^{m_n}, \end{cases} \quad \begin{cases} x^M = \prod_{n=1}^N (x_n)^{m_n}, \\ \delta_{ML} = \prod_{n=1}^N \delta_{m_n l_n}, \\ 1_k = (0, \dots, 1, \dots, 0), \text{ na } k\text{-ésima posição.} \end{cases}$$

E operadores de criação e aniquilação:

$$\begin{aligned} \mathcal{A}_j^\dagger(A, a) &= \frac{1}{\sqrt{2}} \sum_n \left(A_{jn}^{-1} (x_n - a_n) - A_{jn} \frac{\partial}{\partial x_n} \right) \\ \mathcal{A}_j(A, a) &= \frac{1}{\sqrt{2}} \sum_n \left(A_{jn}^{-1} (x_n - a_n) + A_{jn} \frac{\partial}{\partial x_n} \right), \end{aligned} \tag{2.27}$$

que são as componentes do operator $\mathcal{A}^{(\dagger, \cdot)} = (\mathcal{A}_1^{(\dagger, \cdot)}, \mathcal{A}_2^{(\dagger, \cdot)}, \dots, \mathcal{A}_N^{(\dagger, \cdot)})$.

Na semelhança do oscilador harmônico construímos o estado fundamental do sistema pela solução da equação diferencial⁷:

$$\mathcal{A}(A, \vec{a})\varphi_0(A, \vec{a}, \vec{x}) = 0, \quad (2.28)$$

que resulta em:

$$\varphi_0(A, \vec{a}, \vec{x}) = \frac{\pi^{-\frac{N}{4}}}{\sqrt{\det A}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{a})A^{-2}(\vec{x} - \vec{a})\right) \quad (2.29)$$

E então construímos $\varphi_k(A, \vec{a}, x)$ recursivamente por⁸:

$$\varphi_{k+1}(A, \vec{a}, \vec{x}) = \frac{1}{\sqrt{k+1}} \mathcal{A}^\dagger(A, \vec{a})\varphi_k(A, \vec{a}, \vec{x}), \quad (2.30)$$

de forma que, definindo os operadores de criação e aniquilação com multi-índices,

$$\mathcal{A}_M^{(\dagger, \cdot)} = \prod_{n=1}^N (\mathcal{A}_n^{(\dagger, \cdot)})^{m_n},$$

encontramos um conjunto completo de funções ortonormais $\{\varphi_M\} \in L^2$;

$$\varphi_M(A, \vec{a}, \vec{x}) = \frac{1}{\sqrt{M!}} \mathcal{A}_M^\dagger \varphi_0(A, \vec{a}, \vec{x}), \quad (2.31)$$

com,

$$\int \varphi_M(A, \vec{a}, \vec{x}) \varphi_L(A, \vec{a}, \vec{x}) d^N x = \delta_{ML}. \quad (2.32)$$

Sendo reescrito em termos dos polinômios de Hermite:

$$\varphi_M(A, \vec{a}, \vec{x}) = \frac{1}{\sqrt{2^{|M|} M!}} H_M(A, \vec{a}, \vec{x}) \varphi_0(A, \vec{a}, \vec{x}). \quad (2.33)$$

Cada polinômio de Hermite $H_M(A, \vec{a}, \vec{x})$, por sua própria construção, é gerado por:

$$H_M(A, \vec{a}, \vec{x}) = \prod_{n=1}^N H_{m_n}(z_n) = \prod_{n=1}^N (-1)^{m_n} e^{\vec{z} \cdot \vec{z}} \frac{\partial^{m_n}}{\partial z_n^{m_n}} e^{-\vec{z} \cdot \vec{z}}, \quad (2.34)$$

onde $\vec{z} = A^{-1}(\vec{x} - \vec{a})$.

⁷Para uma breve revisão em algumas propriedades das funções que formam uma base de soluções para o oscilador Harmônico veja o apêndice B.

⁸Observe que a eq. 2.28 define a solução 2.29 a menos de uma fase global.

Agora estamos prontos para expandir a distribuição posterior nesse conjunto completo de funções $\{\varphi_M\} \in L^2$, que nada mais é que uma expansão generalizada de Hermite:

$$P(\vec{w}|D) = \sum_{M \in \mathbb{Z}_+^N} \frac{B_M}{2^{|M|} M!} H_M(A, \vec{a}, \vec{w}) \varphi_0^2(A, \vec{a}, \vec{w}), \quad (2.35)$$

com a equação para os coeficientes B 's:

$$B_M = \int H_M(A, \vec{a}, \vec{w}) P(\vec{w}|D) d^N w. \quad (2.36)$$

Veja que temos uma certa liberdade na escolha de A e $\vec{a}$. O primeiro termo $M = (0, \dots, 0)$, ($B_{(0, \dots, 0)} = 1$), será exatamente a mesma função Gaussiana utilizada na seção anterior se fizermos $\vec{a} = \hat{\vec{w}}$ e $A^2 = 2C$. Com isso também teremos que:

$$\begin{aligned} B_{1_k} &= \int H_1(z_k) P(\vec{w}|D) d^N w = \int 2 \sum_i A_{ik}^{-1} (w_i - \hat{w}_i) P(\vec{w}|D) d^N w = 0, \\ B_{2_k} &= \int H_2(z_k) P(\vec{w}|D) d^N w = \int \left(4 (\sum_i A_{ik}^{-1} (w_i - \hat{w}_i))^2 - 2 \right) P(\vec{w}|D) d^N w \\ &= 4 \sum_{ij} A_{ik}^{-1} A_{jk}^{-1} C_{ij} - 2 = 2 \sum_{ijl} A_{ik}^{-1} A_{jk}^{-1} A_{il} A_{lj} - 2 = 2 (\sum_l \delta_{kl}^2 - 1) = 0. \end{aligned}$$

Quando a função a ser expandida nessa base for separável em suas variáveis, ou seja, poder ser escrita como um produto de N funções, uma em cada direção sem depender das demais variáveis que representam as outras direções⁹, teremos que qualquer coeficiente B_M que tenha em uma ou mais de suas direções $m_k = 1$ ou $m_k = 2$ será nulo, de forma a diminuir consideravelmente o número de hiperparâmetro¹⁰ a ser calculado. Outra observação interessante é que a utilização de tal base é mais indicada quando os parâmetros da rede percorrem tanto o eixo negativo de seu espaço quanto o positivo. Quando houver outra informação a priori que leve a limitações no espaço de configuração sugere-se a utilização de outro tipo de base, por exemplo, quando os pesos sinápticos da rede só tomarem valores positivos podemos construir uma base com os polinômios de Laguerre.

Uma vez definida a base a ser utilizada, nosso próximo passo será encontrar as equações de atualização dos parâmetros e hiperparâmetros da rede frente a chegada de novas informações, construindo assim o algoritmo de aprendizado.

⁹Por exemplo: $F(x_1, x_2) = F_1(x_1)F_2(x_2)$

¹⁰ C_{ij} 's e B_M 's

2.2.1 Encontrando as equações de atualização.

Aqui seguiremos o mesmo caminho da seção 2.1: encontrar uma equação para os pesos estimados $\hat{w}_i$'s; para as covariâncias C_{ij} e para os coeficientes B_M .

Atualizando os pesos sinápticos.

Partindo da eq. 2.8 e usando $w_i = \sum_j A_{ij}(t)z_j + \hat{w}_i(t)$ encontraremos:

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \sum_j A_{ij}(t) \int z_j p(\vec{w}(\vec{z})|par_t, y_{t+1}) d^N w(z),$$

ou seja:

$$\Delta \hat{w}_i(t) = \sum_j A_{ij}(t) \int z_j p(\vec{w}(\vec{z})|par_t, y_{t+1}) d^N w(z),$$

que usando a eq. 2.6 e em seguida expandindo a distribuição $p(\vec{w}(\vec{z})|par_t)$ como na eq. 2.35 ficará:

$$\Delta \hat{w}_i(t) = \sum_{M,j} \frac{A_{ij}(t) B_M(t)}{2^{|M|} M! \Xi} \int z_j H_M(z) \varphi_0^2(z) P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t)) d^N z \quad (2.37)$$

onde definimos a função de partição

$$\begin{aligned} \Xi &= \sum_M \frac{B_M(t)}{2^{|M|} M!} \int H_M(z) \varphi_0^2(z) P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t)) d^N z \\ &= \sum_M \frac{B_M(t)}{2^{|M|} M!} \int \varphi_0^2(z) D_z^M P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t)) d^N z \\ &= \sum_M \frac{B_M(t)}{2^{|M|} M!} D_{A^{-1}(t)\hat{\vec{w}}(t)}^M \int \varphi_0^2(z) P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t)) d^N z. \end{aligned} \quad (2.38)$$

Observe que na obtenção da eq. 2.38 usamos novamente o truque da mudança de derivada e integração por partes, como na seção anterior, e que o jacobiano da mudança $d^N w$ para $d^N z$ na eq. 2.37 se cancela com o mesmo termo contido no denominador, eq. 2.38. Além disso, ainda na eq. 2.37, também podemos escrever z_j como $\frac{1}{2} H_1(z_j)$ que se transformará numa derivada e, após uma integração por partes, atuará em $H_M(z) P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t))$. A necessidade de transformarmos os polinômios em

derivadas, tirando-as em seguida de dentro da integral, vem da possibilidade de simplificarmos as integrais e consequentemente os seus cálculos. Normalmente o cálculo de integrais N -dimensionais não é uma tarefa fácil, principalmente quando o número de dimensões N cresce, podendo se complicar mais ainda dependendo da função a ser integrada, de forma que transformar a integral da eq. 2.37 numa derivada da média da distribuição posterior $P(y_{t+1}|A(t)\vec{z} + \hat{w}(t))$ com relação a gaussiana $\varphi_0^2(z)$ pode nos dar um grande ganho em seu cálculo.

Transformando z_j em D_z^{1j} na eq. 2.37 encontramos

$$\Delta \hat{w}_i(t) = \sum_{M,j} \frac{A_{ij}(t)B_M(t)}{2^{|M|}M!\Xi} \int \varphi_0^2(z) \frac{1}{2} D_z^{1j} (H_M(z)P(y_{t+1}|A(t)\vec{z} + \hat{w}(t))) d^N z, \quad (2.39)$$

que fazendo a derivada do produto $H_M P$ e usando a identidade¹¹ (ver apêndice B):

$$D_z^I H_M(z) = \frac{2^{|I|}M!}{(M-I)!} H_{M-I}, \quad (2.40)$$

resulta em:

$$\begin{aligned} \Delta \hat{w}_i(t) = & \sum_{M,j} \frac{A_{ij}(t)B_M(t)}{2^{|M|}M!\Xi} \int \varphi_0^2(z) \left(\frac{1}{2} H_M(z) \partial_{z_j} + \frac{M!}{(M-1_j)!} H_{M-1_j}(z) \right) \\ & \cdot P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) d^N z. \end{aligned} \quad (2.41)$$

Onde usaremos novamente a mudança da derivada D_z^M :

$$\frac{\partial}{\partial z_j} F(A\vec{z} + \hat{w}(t)) = \sum_k A_{kj} \frac{\partial}{\partial \hat{w}_k} F(A\vec{z} + \hat{w}(t)).$$

Da eq. 2.41 obtemos a correção $\Delta \hat{w}_i(t)$ nos pesos sinápticos $\hat{w}_i(t)$ frente a chegada de mais uma informação y_{t+1} :

$$\begin{aligned} \Delta \hat{w}_i(t) = & \sum_{jk} \frac{1}{2} A_{ij}(t) A_{jk}(t) \partial_{\hat{w}_k(t)} \ln \left(\sum_M \frac{B_M(t)}{2^{|M|}M!} D_{A^{-1}(t)\hat{w}(t)}^M \left\langle P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) \right\rangle \right) \\ & + \sum_j A_{ij}(t) \frac{\sum_M \frac{B_M(t)}{2^{|M|}(M-1_j)!} D_{A^{-1}(t)\hat{w}(t)}^{M-1_j} \left\langle P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) \right\rangle}{\sum_M \frac{B_M(t)}{2^{|M|}M!} D_{A^{-1}(t)\hat{w}(t)}^M \left\langle P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) \right\rangle}. \end{aligned} \quad (2.42)$$

Encontrando uma equação de atualização para os pesos $\hat{w}_i$'s ¹²:

¹¹Que retornará valor nulo quando $I > M$.

¹²Relembrando que $A^2 = 2C$.

$$\begin{aligned}
\hat{w}_i(t+1) &= \hat{w}_i(t) \\
&+ \sum_j C_{ij}(t) \partial_{\hat{w}_j(t)} \ln \left(\sum_M \frac{B_M(t)}{2^{|M|} M!} D_{A^{-1}(t)\hat{\vec{w}}(t)}^M \left\langle P(y_{t+1} | A(t)\vec{z} + \hat{\vec{w}}(t)) \right\rangle \right) \\
&+ \frac{\sum_{M,j} \frac{A_{ij}(t) B_M(t)}{2^{|M|} (M-1_j)!} D_{A^{-1}(t)\hat{\vec{w}}(t)}^{M-1_j} \left\langle P(y_{t+1} | A(t)\vec{z} + \hat{\vec{w}}(t)) \right\rangle}{\sum_M \frac{B_M(t)}{2^{|M|} M!} D_{A^{-1}(t)\hat{\vec{w}}(t)}^M \left\langle P(y_{t+1} | A(t)\vec{z} + \hat{\vec{w}}(t)) \right\rangle}. \tag{2.43}
\end{aligned}$$

Atualizando os hiperparâmetros: C.

Uma vez que já encontramos uma forma para atualizar os pesos sinápticos, vamos agora partir em busca da atualização das covariâncias.

Fazendo a mudança $w_i = \sum_j A_{ij}(t) z_j + \hat{w}_i(t)$ na eq. 2.9:

$$\begin{aligned}
C_{ij}(t+1) &= \left\langle \left[\sum_k A_{ik}(t) z_k - \Delta \hat{w}_i(t) \right] \left[\sum_l A_{jl}(t) z_l - \Delta \hat{w}_j(t) \right] \right\rangle \\
&= \sum_{kl} A_{ik}(t) A_{jl}(t) \langle z_k z_l \rangle + \Delta \hat{w}_i(t) \Delta \hat{w}_j(t) \\
&\quad - \Delta \hat{w}_i(t) \sum_l A_{jl}(t) \langle z_l \rangle - \Delta \hat{w}_j(t) \sum_k A_{ik}(t) \langle z_k \rangle;
\end{aligned}$$

portanto,

$$C_{ij}(t+1) = \sum_{kl} A_{ik}(t) A_{jl}(t) \langle z_k z_l \rangle - \Delta \hat{w}_j(t) \Delta \hat{w}_i(t),$$

ou seja,

$$\begin{aligned}
C_{ij}(t+1) &= \sum_{M,k,l} \frac{A_{ik}(t) A_{jl}(t) B_M(t)}{2^{|M|} M! \Xi} \int \varphi_0^2(z) D_z^M \left(z_k z_l P(y_{t+1} | A(t)\vec{z} + \hat{\vec{w}}(t)) \right) d^N z \\
&\quad - \Delta \hat{w}_j(t) \Delta \hat{w}_i(t) \tag{2.44}
\end{aligned}$$

De forma que só precisamos encontrar o valor do primeiro termo, uma vez que já conhecemos $\Delta \hat{w}(t)$. Para isso, escrevamos $z_k z_l$ na forma:

$$z_k z_l = \delta_{kl} z_k^2 + (1 - \delta_{kl}) z_k z_l,$$

e calculemos a derivada D_z^M atuando no produto $z_k z_l f(Az + \hat{w})$:

$$\begin{aligned}
D_z^M(z_k z_l f) &= \delta_{kl} D_z^M(z_k^2 f) + (1 - \delta_{kl}) D_z^M(z_k z_l f) \\
&= \delta_{kl} (z_k^2 D_z^M f + 2m_k z_k D^{M-1}_k f + m_k(m_k - 1) D^{M-2}_k f) \\
&\quad + (1 - \delta_{kl}) (z_k z_l D_z^M f + m_k z_l D^{M-1}_k f + m_l z_k D^{M-1}_l f \\
&\quad + m_k m_l D^{M-1}_{kl} f) \\
&= z_k z_l D_z^M f + m_k z_l D^{M-1}_k f + m_l z_k D^{M-1}_l f \\
&\quad + m_k(m_l - \delta_{kl}) D^{M-1}_{kl} f,
\end{aligned}$$

que ainda pode ser escrito em termos dos polinômios de Hermite:

$$z_k z_l = \frac{1}{4} \delta_{kl} H_2(z_k) + \frac{1}{2} \delta_{kl} + \frac{1}{4} (1 - \delta_{kl}) H_1(z_k) H_1(z_l),$$

logo,

$$\begin{aligned}
\langle D_z^M(z_k z_l f) \rangle &= D_{A^{-1}\hat{w}}^M \left\langle \left(\frac{1}{4} \delta_{kl} H_2(z_k) + \frac{1}{2} \delta_{kl} + \frac{1}{4} (1 - \delta_{kl}) H_1(z_k) H_1(z_l) \right) f \right\rangle \\
&\quad + m_k D_{A^{-1}\hat{w}}^{M-1_k} \langle z_l f \rangle + m_l D_{A^{-1}\hat{w}}^{M-1_l} \langle z_k f \rangle \\
&\quad + m_k(m_l - \delta_{kl}) D_{A^{-1}\hat{w}}^{M-1_{kl}} \langle f \rangle \\
&= \left(\frac{1}{4} \delta_{kl} D_{A^{-1}\hat{w}}^{M+2_l} + \frac{1}{2} \delta_{kl} D_{A^{-1}\hat{w}}^M + (1 - \delta_{kl}) \frac{1}{4} D_{A^{-1}\hat{w}}^{M+1_k+1_l} + m_k D_{A^{-1}\hat{w}}^{M-1_k+1_l} \right. \\
&\quad \left. + m_l D_{A^{-1}\hat{w}}^{M-1_l+1_k} + m_k(m_l - \delta_{kl}) D_{A^{-1}\hat{w}}^{M-1_{kl}} \right) \langle f \rangle,
\end{aligned}$$

que substituindo na eq. 2.44 resultará em:

$$\begin{aligned}
C_{ij}(t+1) &= \sum_{M,k,l} \frac{A_{ik}(t) A_{jl}(t) B_M(t)}{2^{|M|} M! \Xi} \left(\frac{1}{2} \delta_{kl} D_{A^{-1}\hat{w}}^M + \frac{1}{4} D_{A^{-1}\hat{w}}^{M+1_k+1_l} \right. \\
&\quad \left. + m_k D_{A^{-1}\hat{w}}^{M-1_k+1_l} + m_l D_{A^{-1}\hat{w}}^{M-1_l+1_k} + m_k(m_l - \delta_{kl}) D_{A^{-1}\hat{w}}^{M-1_{kl}} \right) \langle P \rangle \\
&\quad - \Delta \hat{w}_j(t) \Delta \hat{w}_i(t), \tag{2.45}
\end{aligned}$$

onde definimos $\langle P \rangle = \int \varphi_0^2(z) P(y_{t+1} | A(t) \vec{z} + \hat{\vec{w}}(t)) d^N z$. Da eq. 2.42 temos que:

$$\begin{aligned}
\Delta \hat{w}_j(t) \Delta \hat{w}_i(t) &= \left(\sum_{kl} \frac{1}{2} A_{jl} A_{lk} \partial_{\hat{w}_k} \ln \left[\sum_M \frac{B_M(t)}{2^{|M|} M!} D_{A^{-1}\hat{w}}^M \langle P \rangle \right] \right. \\
&\quad \left. + \sum_l A_{jl} \sum_M \frac{B_M(t)}{2^{|M|} (M-1_l)! \Xi} D_{A^{-1}\hat{w}}^{M-1_l} \langle P \rangle \right)
\end{aligned}$$

$$\cdot \left(\sum_{kl} \frac{1}{2} A_{il} A_{lk} \partial_{\hat{w}_k} \ln \left[\sum_M \frac{B_M(t)}{2^{|M|} M!} D_{A^{-1} \hat{w}}^M \langle P \rangle \right] \right. \\ \left. + \sum_l A_{il} \sum_M \frac{B_M(t)}{2^{|M|} (M-1_l)! \Xi} D_{A^{-1} \hat{w}}^{M-1_l} \langle P \rangle \right),$$

e sabendo que $\partial_x \partial_y \ln f(x, y) = \frac{\partial_x \partial_y f(x, y)}{f(x, y)} - \frac{\partial_x f(x, y)}{f(x, y)} \frac{\partial_y f(x, y)}{f(x, y)}$, encontraremos:

$$\begin{aligned} C_{ij}(t+1) &= C_{ij}(t) + \sum_{r,s} C_{ir}(t) C_{js}(t) \partial_{\hat{w}_r} \partial_{\hat{w}_s} \ln \left(\sum_M \frac{B_M}{2^{|M|} M!} D_{A^{-1} \hat{w}}^M \langle P \rangle \right) \\ &+ \sum_{M,k,l} \frac{A_{ik} A_{jl} B_M}{2^{|M|} M! \Xi} \left(m_k D_{A^{-1} \hat{w}}^{M-1_k+1_l} + m_l D_{A^{-1} \hat{w}}^{M-1_l+1_k} \right) \langle P \rangle \\ &+ \sum_{M,k,l} \frac{A_{ik} A_{jl} B_M}{2^{|M|} M! \Xi} m_k (m_l - \delta_{kl}) D_{A^{-1} \hat{w}}^{M-1_k-1_l} \langle P \rangle \\ &- \sum_{l,s} \sum_{M,I} \frac{B_M B_I A_{jl} A_{is}}{2^{|M+I|} (M-1_l)! (I-1_s)! \Xi^2} D_{A^{-1} \hat{w}}^{M-1_l} \langle P \rangle D_{A^{-1} \hat{w}}^{I-1_s} \langle P \rangle \\ &- \frac{1}{2} \left(\sum_l A_{jl} \sum_M \frac{B_M}{2^{|M|} M! \Xi} D_{A^{-1} \hat{w}}^{M+1_l} \langle P \rangle \right) \\ &\cdot \left(\sum_s A_{is} \sum_M \frac{B_M(t)}{2^{|M|} (M-1_s)! \Xi} D_{A^{-1} \hat{w}}^{M-1_s} \langle P \rangle \right) \\ &- \frac{1}{2} \left(\sum_s A_{is} \sum_M \frac{B_M}{2^{|M|} M! \Xi} D_{A^{-1} \hat{w}}^{M+1_s} \langle P \rangle \right) \\ &\cdot \left(\sum_l A_{jl} \sum_M \frac{B_M}{2^{|M|} (M-1_l)! \Xi} D_{A^{-1} \hat{w}}^{M-1_l} \langle P \rangle \right) \end{aligned} \quad (2.46)$$

Onde estamos usando a notação de não especificarmos o tempo das variáveis quando esse é igual a "t" (com exceção dos C_{ij} 's) e a atualização dos A 's, $\Delta A = A(t+1) - A(t)$, pode ser feita em seguida em termos de ΔC por meio da *Teoria de Perturbação*, até segunda ordem:

$$\Delta A = A^{-1} \Delta C - \frac{1}{2} A^{-1} (A^{-1} \Delta C)^2. \quad (2.47)$$

A necessidade de se obter os elementos A_{ij} para as eqs. de atualização dos demais parâmetros e hiperparâmetros da rede nos revela uma outra dificuldade quando a matriz de covariância C não é diagonal: o cálculo da raiz quadrada de uma matriz ($A = \sqrt{2C}$). Isso faz com que lancemos mão de aproximações que possam diminuir a complexidade do nosso algoritmo, como na eq. 2.47.

Atualizando os hiperparâmetros: B_M .

Por último, vamos agora encontrar uma equação de atualização para os coeficientes B_M .

Da eq. 2.36 temos que

$$B_M(t+1) = \int H_M(A(t+1), \hat{w}(t+1), \vec{w}) P(\vec{w}|D_{t+1}) d^N w,$$

com,

$$H_M(A(t+1), \hat{w}(t+1), \vec{w}) = H_M(A^{-1}(t+1)(\vec{w} - \hat{w}(t+1))) = H_M(\Lambda \vec{z} - \Delta' \hat{w}(t)).$$

Sendo $\Lambda = A^{-1}(t+1)A(t)$; $\Delta' \hat{w}(t) = A^{-1}(t+1)\Delta \hat{w}(t)$ e $\vec{z} = A^{-1}(t)(\vec{w} - \hat{w}(t))$, o que traz o jacobiano $\mathcal{J} = \det |A(t)|$ que se cancela novamente com o denominador eq. 2.38, e por meio da regra de Bayes 2.5 e da expansão 2.35 chegaremos à:

$$B_M(t+1) = \sum_I \frac{B_I(t)}{2^{|I|} I! \Xi} \int H_M(\Lambda z - \Delta' \hat{w}(t)) H_I(z) \varphi_0^2(z) P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) d^N z. \quad (2.48)$$

Com isso, o nosso próximo passo será expandir os polinômios $H_M(\Lambda z - \Delta' \hat{w}(t))$ em termos do conjunto de polinômios $H_L(z)$, de forma que possamos novamente transformá-los em derivadas em $\hat{w}$ tirando-os em seguida da integral.

$$H_M(\Lambda z - \Delta' \hat{w}(t)) = \sum_L \frac{\Gamma_{ML} H_L(z)}{2^{|L|} L!}. \quad (2.49)$$

$$\Gamma_{ML} = \int \varphi_0^2(z) H_M(\Lambda z - \Delta' \hat{w}(t)) H_L(z) d^N z. \quad (2.50)$$

Com a introdução do coeficiente Γ podemos passar de uma integral de dois polinômios de Hermite por uma Gaussiana e pela distribuição posterior da nova informação y_{t+1} para uma integral com apenas polinômios e Gaussiana, que novamente apresenta dificuldades quando a matriz A não for diagonal e que normalmente é vista como uma probabilidade de transição do estado “ M ” para o “ L ” no problema do Oscilador Harmônico Quântico, e outra integral que nada mais é do que a média da distribuição posterior da nova informação com relação a distribuição φ^2 . De forma que a equação de atualização 2.48 ficará:

$$B_M(t+1) = \sum_{I, L \leq M} \frac{B_I(t) \Gamma_{ML}}{2^{|I+L|} I! L! \Xi} \int H_L(z) H_I(z) \varphi_0^2(z) P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) d^N z$$

$$= \sum_{I, L \leq M} \frac{B_I(t) \Gamma_{ML}}{2^{|I+L|} I! L! \Xi} \int \varphi_0^2(z) D_z^I (H_L(z) P(y_{t+1} | A(t) \vec{z} + \hat{\vec{w}}(t))) d^N z,$$

a qual por meio da regra da derivada do produto:

$$\begin{aligned} D^I (H_L P) &= \sum_{S \leq I} \frac{I!}{S! (I-S)!} D^{I-S} H_L D^S P \\ &= \sum_{S \leq I} \frac{2^{|I-S|} I! L!}{S! (I-S)! (L-I+S)!} H_{L-I+S} D^S P, \end{aligned}$$

pode ser escrita como:

$$B_M(t+1) = \sum_{I, L \leq M, S \leq I} \frac{B_I(t) 2^{-|S+L|} \Gamma_{ML}}{S! (I-S)! (L-I+S)! \Xi} D_{A^{-1}(t) \hat{\vec{w}}(t)}^{L-I+2S} \left\langle P(y_{t+1} | A(t) \vec{z} + \hat{\vec{w}}(t)) \right\rangle. \quad (2.51)$$

Quando consideramos todos os elementos das equações de atualização (eq.: 2.43; 2.46; 2.51), e uma vez que suas respectivas séries convirjam, temos então uma atualização off-line da distribuição posterior. Mas sua implementação em novos algoritmos de aprendizagem fica bastante limitada devido a sua complexidade, revelada pelo número de hiperparâmetros. Então o que temos pode ser visto como apenas uma descrição formal de processos off-line, de onde podemos tirar alguma nova informação que indique caminhos que venham a melhorar o desempenho de outros algoritmos. Uma amostra disso será visto na próxima seção, onde encontraremos cenários semelhantes à outras técnicas, tipo o método da cavidade, que a princípio não estão correlacionados. Uma outra observação é que nesse algoritmo off-line temos uma aparente quebra de simetria entre os padrões do conjunto de treinamento, devido à explícita atenção dada ao padrão mais recente y_{t+1} , ao contrário dos demais algoritmos off-line, em sua maioria.

Por meio da expansão eq. 2.35 podemos ainda definir diferentes aproximações analisando sistematicamente o efeito de ir adiante na expansão, por considerar um número cada vez maior de hiperparâmetros, com o intuito de extrair e usar mais informações relevantes ao aprendizado em relação aos usuais algoritmos on-line restritos aos pesos sinápticos e covariâncias.

Também deve ser notado que ao truncarmos a expansão possibilitamos que a distribuição posterior admita valores negativos. Mas salientamos que não estamos interessados na evolução da distribuição posterior e sim na aproximação de algoritmos off-line Bayesianos eficientes por algoritmos on-line, justificando-se por seu

desempenho e custo computacional, sendo ainda o nosso principal foco de interesse teórico a passagem para o equilíbrio termodinâmico.

2.3 Usando uma distribuição Gibbsiana.

O processo de aprendizado de uma rede neural por um conjunto de exemplos também pode ser tratado por uma aproximação Gibbsiana, onde introduzimos uma função *custo-energia* $E_t = \sum_{k=1}^t V(\vec{w}, y_k)$ em termos de um potencial $V(\vec{w}, y_k) = V(\sigma, \mathcal{F}_w, \xi)$, que de alguma forma quantifique a distância entre a saída (resposta) da rede aluno, definida pelo mapeamento $\mathcal{F}_w$, e a rede professor ($\mathcal{F}$) no espaço descrito pelo conjunto de treinamento D_t . Com isso, temos agora dois possíveis caminhos para a rede *aprender*: ou encontramos uma configuração dos parâmetros da rede que maximize a distribuição $P(D_t|\vec{w})$ (eq.2.2); ou minimizamos a função custo-energia, diminuindo assim o erro cometido por $\mathcal{F}_w$. Mas, mais razoável ainda é que os dois caminhos sejam equivalentes, ou seja, quando encontrarmos uma configuração que maximize a probabilidade de se gerar os pares y_t , esta também minimize a função custo-energia. Para tornar esses dois critérios equivalentes é necessário que sejam diretamente relacionadas por uma função diferenciável e monotônica ϕ ,

$$\prod_{k=1}^t P(y_k|\vec{w}) = \phi(E_t), \quad (2.52)$$

considerando que suas derivadas com respeito a w se anulem nos pontos extremos de $P(y_k|\vec{w})$ e E_t .

Para um potencial diferenciável a eq. 2.52 resultará em [17]

$$P(y_k|\vec{w}) = \frac{e^{-\beta V(\vec{w}, y_k)}}{\int e^{-\beta V(\vec{w}, y_k)} d^N \vec{w}}, \quad (2.53)$$

que de acordo com a regra de Bayes eq. 2.5 nos dará a equação:

$$P_V(\vec{w}|D_{t+1}) = \frac{P_V(\vec{w}|D_t) e^{-\beta V(\vec{w}, y_{t+1})}}{\int P_V(\vec{w}'|D_t) e^{-\beta V(\vec{w}', y_{t+1})} d^N \vec{w}'}. \quad (2.54)$$

A partir da eq. 2.54 podemos seguir o mesmo procedimento da seção anterior encontrando as mesmas equações 2.43, 2.46 e 2.51, substituindo $\langle P(y_{t+1}|A(t)\vec{z} + \hat{\vec{w}}(t)) \rangle$ por $\langle e^{-\beta V(\vec{w}, y_{t+1})} \rangle$. Mas agora, consideraremos apenas o caso em que a tarefa dada à

rede seja separar as informações de entrada em duas classes $\sigma = \pm 1$, sendo o potencial uma função diferenciável da estabilidade $\lambda = \sigma \vec{w} \cdot \vec{\xi} / \sqrt{N}$, onde N corresponde a dimensão de $\vec{\xi}$ e $\vec{w}$. O que nos dá dois novos parâmetros:

$$\begin{cases} \tau = \sigma \hat{\vec{w}} \cdot \vec{\xi} / \sqrt{N} \rightarrow \text{estabilidade do exemplo } (\vec{\xi}, \sigma) \text{ na atual rede,} \\ \lambda = \sigma \vec{w} \cdot \vec{\xi} / \sqrt{N} = \tau + \sigma \vec{z} A \cdot \vec{\xi} / \sqrt{N} \rightarrow \text{estabilidade para qualquer rede } \vec{w}. \end{cases} \quad (2.55)$$

Também faremos as mudanças: $\tilde{C}_{ij}(t) = \beta C_{ij}(t)$ e $\mathbf{x} = \sum_{ij} \frac{\xi_i \tilde{C}_{ij} \xi_j}{2N}$, de maneira que passaremos de uma integral em N dimensões para uma integral unidimensional na estabilidade λ trazendo um grande ganho de cálculo a medida que N cresce, compensando algumas dificuldades de cálculo já encontradas na seção anterior.

$$\begin{aligned} \langle e^{-\beta V(\vec{w}, y_{t+1})} \rangle &= \int \varphi_0^2(z) e^{-\beta V(\lambda)} \delta(\lambda - (\tau + \sigma \vec{z} A \cdot \vec{\xi} / \sqrt{N})) d\lambda d^N z \\ &= \int \varphi_0^2(z) e^{-\beta V(\lambda)} e^{i\theta(\lambda - \tau - \sigma \vec{z} A \cdot \vec{\xi} / \sqrt{N})} d\theta d\lambda d^N z \\ &= cte \int \exp[-(\vec{z} + \frac{i\theta\sigma}{2\sqrt{N}} \vec{\xi} A)^2] e^{-\beta V(\lambda) + i\theta(\lambda - \tau) - \frac{\theta^2}{4N} \vec{\xi} A^2 \vec{\xi}} d\theta d\lambda \\ &= cte \int \exp(-\beta V(\lambda) + i\theta(\lambda - \tau) - \frac{\theta^2}{4N} \vec{\xi} A^2 \vec{\xi}) d\theta d\lambda \\ &= cte \int \exp\left(-\beta V(\lambda) - \beta \frac{(\lambda - \tau)^2}{2\mathbf{x}}\right) d\lambda. \end{aligned} \quad (2.56)$$

Com isso, vamos agora estudar o limite¹³ $\beta \rightarrow \infty$. A integral eq. 2.56 pode então ser calculada pelo método do ponto de sela, que consiste em expandir o argumento da exponencial em torno de seus pontos extremos,

$$\begin{aligned} V(\lambda) + \frac{(\lambda - \tau)^2}{2\mathbf{x}} &= \sum_n \frac{\partial^n}{\partial \lambda^n} \left[V(\lambda) + \frac{(\lambda - \tau)^2}{2\mathbf{x}} \right] \Big|_{\lambda=\lambda_0} (\lambda - \lambda_0)^n / n! \\ &\simeq V(\lambda_0) + \frac{(\lambda_0 - \tau)^2}{2\mathbf{x}} + \dots \end{aligned}$$

Onde λ_0 representa um extremo, ou seja, $[\frac{\partial V(\lambda)}{\partial \lambda} + \frac{\lambda - \tau}{\mathbf{x}}]_{\lambda=\lambda_0} = 0$. Tal resultado nos leva a definir um *potencial efetivo* dado por:

¹³O parâmetro de temperatura β foi introduzido por meio da eq. 2.53 para que, ao iniciarmos o processo de procura da configuração que minimiza a energia, o sistema não fique preso num mínimo local. Assim, β está diretamente ligado a uma pequena energia adicionada na equação de *Langevin* [18] do sistema, por meio de uma temperatura T ($\beta \propto 1/T$). De forma que devemos tomar novamente o limite de $T \rightarrow 0$ ($\beta \rightarrow \infty$) referente a situação em que o sistema já se encontra no mínimo global, retirando assim aquela energia que acrescentamos apenas para encontrá-lo.

$$\mathcal{E}_x(\tau) \equiv V(\lambda_0) + \frac{(\lambda_0 - \tau)^2}{2x}. \quad (2.57)$$

Assim, podemos reescrever as equações de atualização (2.43; 2.46; 2.51) em função desse potencial efetivo $\mathcal{E}_x(\tau)$ substituindo a média $\langle P(y_{t+1}|A(t)\vec{z} + \hat{w}(t)) \rangle$ pela exponencial $e^{-\mathcal{E}_x(\tau)}$. Para isso, observe que as derivadas $D_{A^{-1}\hat{w}}^M$ se transformam em:

$$D_{A^{-1}\hat{w}}^M = \prod_{i=1}^N \left(\sum_j A_{ij} \frac{\partial}{\partial \hat{w}_j} \right)^{m_i} = \prod_{i=1}^N \left(\sum_j A_{ij} \frac{\sigma \xi_j}{\sqrt{N}} \right)^{m_i} \partial_\tau^{|M|}. \quad (2.58)$$

Que para o caso em que $B_M = 0$, com $M \neq 0$, nos levará às eqs. de atualização:

$$\hat{w}_i(t+1) = \hat{w}_i(t) - \sum_j \frac{1}{\sqrt{N}} \tilde{C}_{ij}(t) \xi_j \sigma(t) \frac{\partial \mathcal{E}_x(\tau)}{\partial \tau}, \quad (2.59)$$

$$\tilde{C}_{ij}(t+1) = \tilde{C}_{ij}(t) - \sum_{k,l} \frac{1}{N} \tilde{C}_{ik}(t) \tilde{C}_{jl}(t) \xi_k \xi_l \frac{\partial^2 \mathcal{E}_x(\tau)}{\partial \tau^2}. \quad (2.60)$$

A eq. 2.59 pode ser vista como algoritmo *Hebbiano* de aprendizado com um ajuste tensorial modulado por $\frac{\partial}{\partial \tau} \mathcal{E}_x(\tau)$, o gradiente do potencial off-line calculado na estabilidade λ_0 ¹⁴ transmutado a um potencial on-line $\mathcal{E}_x(\tau)$, calculado agora no ponto τ : a estabilidade da rede após ser treinada com t exemplos.

O efeito de transmutação do potencial também foi observado em [19][11], onde os autores usaram o método de otimização variacional para determinar um potencial que maximizasse o desempenho por extremar medidas de desempenho, tais como o erro de generalização com respeito ao potencial. Aplicando para algumas arquiteturas de rede em aprendizado off-line e on-line, no limite termodinâmico, foi encontrado que, para um perceptron de uma camada, a eq. 2.57 fornece precisamente a relação entre o potencial ótimo on-line e o off-line. E a mesma relação também se encontra num cenário de aprendizagem não supervisionada [20].

Analizemos rapidamente agora a eq. 2.60 por meio de um potencial diferenciável V que seja plano para grandes magnitudes da estabilidade, ver figura 2.2. Temos que para valores negativos da estabilidade o potencial V satura num valor positivo enquanto que para valores positivos da estabilidade o potencial vai a zero, decaindo na região de transição monotonicamente, de forma que a segunda derivada que entra na eq. 2.60 será positiva se a nova informação for classificada corretamente e negativa se não for. Isso significa que a rede muda seu comportamento de acordo com uma

¹⁴ $\frac{\partial V}{\partial \lambda} \Big|_{\lambda=\lambda_0} = \frac{\partial \mathcal{E}_x}{\partial \tau}$

estimação de seu próprio desempenho: se comete um erro (classifica incorretamente a informação) ela reage aumentando sua estimativa da covariância da distribuição posterior de maneira que possa ser feita maiores correções na estimativa dos pesos sinápticos w_i 's (eq. 2.59). E quando a nova informação é classificada corretamente a rede fará menores correções em suas estimativas dos w_i 's.

Figura 2.2: Um típico potencial V da estabilidade λ da rede aluno.

Seguindo o mesmo desenvolvimento feito no final da seção 2.1, podemos mostrar que a matriz de covariância C é governada assintoticamente por

$$\lim_{t \rightarrow \infty} \frac{C^{-1}}{t} = \mathcal{J}_V(\vec{w}^*), \quad (2.61)$$

onde $[\mathcal{J}_V(\vec{w}^*)]_{ij} = \overline{\partial_i \partial_j \mathcal{E}_x(t)}$ e a barra superior representa a média com relação a distribuição $Q(y)$. E assim como na seção 2.1, podemos provar a eficiencia assintótica para o algoritmo descrito pelas eqs. 2.59 e 2.60 segundo o critério de Rao-Cramér.

Para entendermos melhor tais algoritmos, descritos nesse capítulo, estudaremos algumas aplicações onde poderemos ver o quão robusto serão os algoritmos de aprendizado gerados por tal desenvolvimento (tanto para o caso de usarmos uma função potencial ou de não usarmos, como na seção 2.1). Tal robustez será analisada para o caso em que o conjunto de treinamento é corrompido por algum tipo de ruído e para o caso em que a rede professor muda no tempo, ou seja, a regra de processamento para os dados de entrada varia durante o processo de aprendizado. Para

tais aplicações decidimos usar o *Perceptron*, tomando um caso unidimensional para implementar o algoritmo da seção 2.1 e n-dimensional para implementar o algoritmo com uma distribuição Gibbsiana, seção 2.2.

3

O PERCEPTRON UNIDIMENSIONAL

O Perceptron foi introduzido pela primeira vez no meio acadêmico pelo grupo de Rosenblatt num trabalho publicado em 1962 [3], tendo grande repercussão por construir um algoritmo para a modificação dos pesos sinápticos w de forma que as informações de saída do perceptron sempre¹ convergiam para os resultados desejados.

A rede Perceptron em sua forma mais simples pode ser vista não como uma rede mais como uma unidade de processamento, assim como os neurônios biológicos, com apenas uma camada com um único neurônio (ver fig. 3.1). As informações ξ_i entram no perceptron, são ponderadas por meio dos pesos sinápticos w_i e processadas no núcleo por uma função de transferência $f(\xi, w)$ resultando numa saída σ .

Figura 3.1: Um perceptron simples.

¹Embora em 1969 Minsky e Papert mostraram um problema que não podia ser resolvido pelo perceptron simples, foi mostrado mais adiante que adicionando mais camadas de neurônios no perceptron tal problema era resolvido.

De acordo com a forma da função de transferência podemos classificá-lo em:

- **Linear:** $f(h) = h$;
- **Analógico:** $f(h)$ é uma função sigmodal tipo $\tanh(h)$ ou $(1 + e^{-h})^{-1}$;
- **Booleano:** $f(h) = \text{sign}(h)$ ou $f(h) = \Theta(h) = \frac{1}{2}[1 + \text{sign}(h)]$.

Onde h é normalmente tratado como o campo local $h(\vec{\xi}) = \vec{\xi} \cdot \vec{w} / \sqrt{N}$, sendo N o número de dimensões de $\vec{\xi}$ e $\vec{w}$.

Neste capítulo e nos que seguirão iremos nos limitar ao caso de aprendizado supervisionado, onde o conjunto de treinamento é formado por pares $y = (\xi, \sigma)$, para um perceptron simples booleano com uma função de transferência dada por $f(h) = \text{sign}(h)$.

Aqui estudaremos o problema de treinarmos um perceptron simples com uma única entrada com os pares (ξ, σ) , sendo a entrada ξ um número real e o campo² h com uma pequena modificação dada por $h = \xi - w_0$, onde w_0 será o parametro a ser aprendido, representado simplesmente por w para a rede aluno e por w_0 para a rede professor, e pertencente ao conjunto dos números reais contidos no intervalo $[0, 1]$. Para tal sistema a distribuição $P(y_{t+1}|w)$ será dada como:

$$P(y_{t+1}|w) = \begin{cases} 1 - \epsilon_0 & \text{se } \begin{cases} \sigma = 1 \text{ e } \xi \geq w \\ \sigma = -1 \text{ e } \xi < w \end{cases} \\ \epsilon_0 & \text{se } \begin{cases} \sigma = 1 \text{ e } \xi < w \\ \sigma = -1 \text{ e } \xi \geq w \end{cases} \end{cases} \quad (3.1)$$

onde ϵ_0 representa a quantidade de ruído presente no conjunto de treinamento. E também teremos que

$$P(w|D) = \sum_{m \in \mathcal{N}_+} \frac{b_m}{2^m m!} H_m(a, \hat{w}, w) \varphi_0^2(a, \hat{w}, w), \quad (3.2)$$

onde $a^2 = 2c = 2 < (w - \hat{w})^2 > e^3$

$$b_m = \int H_m(a, \hat{w}, w) P(w|D) dw, \quad (3.3)$$

e os primeiros polinômios de Hermite dados por:

²Também conhecido como campo pós-sináptico.

³Lembrando que $b_1 = b_2 = 0$.

$$\begin{cases} H_0(a(t), \hat{w}(t), w) = 1; \\ H_1(a(t), \hat{w}(t), w) = 2(w - \hat{w}(t))/a(t); \\ H_2(a(t), \hat{w}(t), w) = 4(w - \hat{w}(t))^2/a^2(t) - 2; \\ H_3(a(t), \hat{w}(t), w) = 8(w - \hat{w}(t))^3/a^3(t) - 12(w - \hat{w}(t))/a(t); \\ H_4(a(t), \hat{w}(t), w) = 16(w - \hat{w}(t))^4/a^4(t) - 48(w - \hat{w}(t))^2/a^2(t) + 12. \end{cases}$$

Por último, as equações de atualização para o parâmetro e hiperparâmetro da rede:

$$\hat{w}(t+1) = \hat{w}(t) + \sum_m \frac{a(t)b_m(t)}{2^m m! \Xi} \int z H_m(z) \varphi_0^2(z) P(y_{t+1}|az + \hat{w}(t)) dz, \quad (3.4)$$

$$c(t+1) = \sum_m \frac{a(t)b_m(t)}{2^m m! \Xi} \int (a(t)z - \Delta \hat{w}(t))^2 H_m(z) \varphi_0^2(z) P(y_{t+1}|az + \hat{w}(t)) dz. \quad (3.5)$$

Observe que não utilizaremos as equações 2.43 e 2.46 pois a distribuição $P(y_{t+1}|w)$ não é diferenciável, o que nos impossibilita usar alguns truques tipo a integração por partes, e que essa indiferenciabilidade pode ser contornada se trocamos de ruído multiplicativo para aditivo, o que põe incerteza na superfície de decisão fazendo com que tenhamos uma função degradal para $P(y_{t+1}|w)$ parecida com uma $\tanh$, ou seja, tornando-a diferenciável.

Aqui resolvemos usar um problema unidimensional apenas por facilitar a implementação do algoritmo de aprendizado, com relação aos cálculos das integrais contidas nas equações de atualização 3.3, 3.4 e 3.5. Servindo também para uma melhor compreensão dos significados práticos dos parâmetros e hiperparâmetros do sistema, como uma preparação para o ingresso em sistemas com dimensões maiores aonde a inferência de significados nem sempre é fácil⁴.

A seguir vamos examinar tal sistema monitorando o erro quadrático médio : $\langle (\hat{w} - w_0)^2 \rangle$, sendo a média feita sobre os conjuntos de treinamento e professores w_0 , para indicar o desempenho da rede aluno com relação à rede professor em três cenários distintos: com w_0 fixo durante todo o aprendizado; variando suavemente e bruscamente.

Como condições iniciais tomaremos $\hat{w}(0) = 0$ e $c(0) = 1$, que como será visto adiante não se trata apenas de uma escolha qualquer.

⁴Isso obviamente só faz sentido com relação a grandezas que constam nos dois casos: para $N = 1$ e $N > 1$.

3.1 Professor constante no tempo.

A utilização de polinômios de ordem maiores que um não trouxe grandes mudanças, como pode ser observado na fig. 3.2 onde usamos polinômios de até quarta ordem. Isso devido à uma rápida convergência da rede aluno à rede professor que acompanha a convergência do hiperparâmetro $c(t)$ à zero (ver fig. 3.3), o que causa uma contribuição cada vez menor dos polinômios nas integrais predominando a Gaussiana. De maneira que nas próximas simulações dessa seção usaremos apenas

Figura 3.2: Diferença de desempenho para o Perceptron entre a parametrização com polinômios até a quarta ordem e com apenas o primeiro termo da série.

o primeiro termo da expansão 3.2, poupando assim grande tempo de simulação.

Como observa-se na fig. 3.4 a rede aluno obtém um ótimo desempenho em seu aprendizado mesmo para altos valores do nível de ruído, mas tal desempenho foi alcançado dando ao aluno os valores corretos dos respectivos níveis de ruído, sendo tal informação inerente ao referente algoritmo de aprendizado mas que nem sempre

Figura 3.3: Evolução do hiperparâmetro $c(t)$ no processo de aprendizado.

Figura 3.4: Desempenho da rede aluno para diferentes níveis de ruído.

está disponível (ao menos na maioria dos casos) ao aluno. O que torna necessário uma análise sobre a robustez de tal algoritmo de aprendizado sobre níveis de ruídos não corretos, ou seja, o comprometimento da estimação do nível de ruído pelo aluno em seu aprendizado. Nesse intuito foram feitas várias simulações onde variou-se tanto o nível de ruído presente no conjunto de treinamento quanto o estimado, construindo com tal informação um gráfico do expoente de queda da curva do desvio médio quadrático em função do nível de ruído presente ϵ_0 e do estimado ϵ pelo aluno (fig. 3.5).

Figura 3.5: Expoente de queda q do desvio $\langle (\hat{w} - w_0)^2 \rangle$ em função dos níveis de ruído real ϵ_0 e estimado ϵ (dados em %).

Tal expoente é medido por meio de um enquadramento das curvas 3.4 por uma lei de potência $\langle (\hat{w} - w_0)^2 \rangle \propto (n. \text{ exemplos})^{-q}$ variando tanto ϵ_0 quanto ϵ . Como resultado tirado pela observação da figura 3.5 temos que uma subestimação do professor pelo aluno sempre leva a um melhor desempenho que uma superestimação, ou até mesmo melhor que o valor correto do nível de ruído. O que leva a estimarmos sempre um nível de ruído alto como medida de segurança, obtendo-se assim um

desempenho médio melhor.

3.2 Professor com variações no tempo.

Diferentemente da última seção iremos agora observar um pouco o cenário onde o professor não permanece constante durante todo o processo de aprendizado, com mudanças contínuas ou bruscas. Longe de querermos fazer uma análise completa sobre tal cenário⁵, onde levanta-se todas as relações sobre os parâmetros envolvidos (tais como amplitude ou frequência de oscilação para mudanças contínuas e não estacionárias) com o desempenho da rede, aqui estamos mais interessados em apenas responder a pergunta: *Será o algoritmo estudado restrito apenas a situações em que a rede professor permanece constante durante o treinamento?*

Uma observação interessante dentre tal cenário é que: uma mudança no professor feita de forma aleatória e dentro de algum intervalo é equivalente a termos um ruído aleatório aditivo⁶ no professor w_0 , o que tem maior efeito para as entradas que estão perto da superfície de decisão⁷ discriminadamente. De forma que mesmo sem termos nos aplicados em simularmos o perceptron com ruído aditivo, podemos ter alguma idéia disso.

Como no caso anterior, seção 3.1, não se obtém grandes melhoras ao adicionar-se na eq. 3.2 os termos que contém polinômios com ordem maiores que um. De forma que para as nossas simulações usaremos apenas uma gaussiana sem correções.

Para começarmos vejamos o caso em que o professor muda harmonicamente durante o treinamento (fig: 3.6).

Como resultado observamos que para pequenas amplitudes de oscilação (menores que 0,1) a rede aluno consegue acompanhar a rede professor mesmo que esse possua um alto nível de ruído (ver fig: 3.7), caíndo mais lentamente que no caso anterior (seção 3.1) e com uma amplitude de oscilação que vai diminuindo com o processo de aprendizado. Já para professores que variam de forma aleatória (figs. 3.8 e 3.9) observamos que a rede aluno não consegue acompanhar o professor tão bem se apegando aos primeiros conceitos aprendido, sendo observado que quando o professor

⁵Para um estudo nessa direção sugerimos uma leitura do trabalho [21].

⁶Que para o nosso caso consta em fazermos $w_0 \rightarrow w_0 + \delta \leq d_{máx}$ e $\tilde{w}_0 \cdot w_0 = \delta$ para um perceptron n-dimensional.

⁷A superfície que separa as entradas classificadas em +1 das classificadas em -1.

Figura 3.6: Professor variando harmonicamente durante o treinamento.

Figura 3.7: Desvio médio quadrático para um professor que varia harmonicamente durante o treinamento.

Figura 3.8: Rede professor w_0 com mudanças aleatórias de $\delta \leq 0,1$ a cada exemplo.

Figura 3.9: Desvio médio quadrático para um professor com mudanças aleatórias.

se aproxima da região inicial do processo o aluno tem uma melhora e o seu desvio quadrático com relação ao professor volta a cair. Valendo salientar que umas das diferenças fundamentais aqui com relação ao caso harmônico é a amplitude de oscilação. Como no caso harmônico o professor se afasta do seu valor médio por pequenas distâncias o aluno consegue acompanhá-lo com mais sucesso. Já no caso de variações aleatórias, mesmo que cada passo em si seja pequeno, o professor poderá se afastar por longas distâncias de seu valor inicial e sem um período de retorno definido, o que faz com que não se encontre uma configuração que se adeque a tal regime. Assim, como já esperado, encontramos que: uma vez que o aluno aprende bem uma determinada regra, ou professor, ele já não consegue acompanhar tão bem uma mudança abrupta no professor. Isso porque quando o aluno aprende bem uma regra sua aceitabilidade a novas mudanças já se encontra muito próxima de zero, ou seja, a variância c , chegando a ser tão pequena quanto melhor tiver aprendido a regra. Portanto quando a regra é mudada abruptamente temos que se $c \ll 1$ então o novo aprendizado é muito mais difícil. Por isso que temos na figura 3.10 uma queda mais acentuada para grandes níveis de ruído, devido ao fato que com grandes níveis de ruído o aluno não chega a aprender tão bem a regra, apesar de se aproximar bastante do professor, e sua variância não chega a ficar pequena.

Figura 3.10: Desvio médio quadrático para um professor com mudanças bruscas.

Isso também pode ser considerado razoável após uma análise das equações de atualização 2.43 ou 2.59, ou seja, uma vez que $\hat{w}_i$ encontra o seu valor adequado não devemos mais fazer correções por meio de Δw , o que é satisfeito zerando o termo de ajuste C_{ij} , ou equivalentemente A_{ij} .

4

O PERCEPTRON N-DIMENSIONAL

Neste capítulo veremos uma aplicação da teoria desenvolvida na seção 2.3 com o intuito de irmos além duma aproximação Gaussiana. Como já mencionado, iremos utilizar um perceptron booleano em N-dimensões. O campo pós-sináptico será dado por $h(\vec{\xi}) = \vec{\xi} \cdot \vec{w} / \sqrt{N}$.

Seria natural usar o potencial de Rosenblatt $V(\lambda) = -\lambda\Theta(-\lambda)$, pois esse apresenta a maior diferença entre a estratégia on-line ($e_g \propto \alpha^{-1/3}$) e a off-line ($e_g \propto \alpha^{-1}$) entre os potenciais já estudados. Porém o estudo do seu comportamento fica comprometido se usarmos as equações que obtivemos no capítulo anterior, devido a sua não diferenciabilidade em $\lambda = 0$. Uma maneira tentadora de contornar tal não-diferenciabilidade é colocar ruído aditivo no perceptron. Porém uma consequência desagradável surge quando tomamos o limite de ruído nulo nas equações de atualização 2.43; 2.46 e 2.51. Isto pode ser melhor visto nas eqs. 2.59 e 2.60, mas o que aconteceria se tomássemos tal limite, não nas equações finais, mas sim no cálculo do valor esperado da exponencial do potencial? Veja que o problema anterior está em utilizarmos o potencial mencionado diretamente na eq. 2.57, do potencial efetivo. De forma que a sugestão tentadora é que seja tomado tal limite no passo anterior, no cálculo da integral eq. 2.56, que surpreendentemente resulta num potencial efetivo

$$\mathcal{E}_{\tilde{x}} = -\ln(\text{erfc}(-\tau/\sqrt{2\tilde{x}})), \quad (4.1)$$

onde $\tilde{x} = \sum_{ij} \xi_i C_{ij} \xi_j / N$, possuindo exatamente a mesma forma funcional do potencial on-line obtido pelo método variacional ótimo [19]!

É importante observar que a obtenção do potencial on-line ótimo surge apenas por causa da não-diferenciabilidade do potencial off-line de Rosenblatt, pois caso contrário iríamos direto para a eq. 2.57, que a princípio, não está claramente relacionada com o potencial on-line obtido pelo método variacional ótimo. Lembrando

que, embora seja tal transmutação ótima um resultado agradável, ainda temos o inconveniente de ter de colocar um ruído aditivo e tomarmos o limite de ruído nulo não nas equações de atualização do sistema, mas num ponto específico e particular para o dado potencial.

Estudaremos o potencial 4.1 para o Perceptron em N -dimensões porque esse é o que satura no caso off-line ($e_g \propto 0,44/\alpha$) o limite de Bayes e é ótimo no caso on-line ($e_g \propto 0,88/\alpha$) no limite Termodinâmico. O problema que surge é que o efeito procurado no desempenho da rede ao se adicionar os hiperparâmetros B_M 's será muito menor, pois agora o expoente é o mesmo e a diferença se restringe ao coeficiente de e_g .

Para diminuir a complexidade das eqs. de atualização 2.43; 2.46 e 2.51, consideraremos o caso em que a matriz de covariância C é diagonal, ou seja:

$$C_{ij} = C_{ii}\delta_{i,j} \implies A_{ij} = A_{ii}\delta_{i,j} = \sqrt{2C_{ii}}\delta_{i,j}, \quad (4.2)$$

que é razoável no limite $N \rightarrow \infty$.

Com essa aproximação e usando o potencial efetivo 4.1 encontraremos as equações de atualização para o perceptron N -dimensional (que estão escritas no apêndice C). Mas antes de prosseguir vejamos mais algumas propriedades e definições sobre o Perceptron não citadas no capítulo anterior.

Como no capítulo passado, nós iremos estudar o Perceptron com os seus pesos sinápticos tomando valores reais ($w_i \in \mathbb{R}$), o que é conhecido como perceptron esférico, com uma função de transferência Booleana e com os vetores de entrada também tomando valores reais ($\xi_i \in \mathbb{R}$). Observemos também que agora o campo pós-sináptico tem a sua estrutura usual do produto escalar entre o peso $\vec{w}$ e o exemplo $\vec{\xi}$, sendo $\sigma = \text{sign}(\vec{w}^* \cdot \vec{\xi})$ a resposta da rede professor $\vec{w}^*$ associada ao exemplo $\vec{\xi}$, de forma que σ é invariante sobre a transformação $\vec{\xi} \rightarrow \mu\vec{\xi}$. Assim vamos impor a normalização $\vec{\xi} \cdot \vec{\xi} = 1$ e $\vec{w}^* \cdot \vec{w}^* = 1$, o que define o espaço dos pesos sinápticos $\vec{w}^*$ como uma hipersfera N -dimensional de raio unitário.

Na figura 4.1 temos o vetor $\vec{w}^*$ em seu próprio espaço e perpendicular ao hiperplano $(N-1)$ -dimensional $\mathbf{D}$, de forma que apenas os exemplos $\vec{\xi}$ que estiverem no mesmo lado de $\vec{w}^*$ com relação a $\mathbf{D}$ é que serão classificados como $\sigma = +1$. Assim a rede aluno será bem sucedida quando o exemplo $\vec{\xi}$ cair na área de intersecção entre o seu lado positivo (negativo) e o lado positivo (negativo) da rede professor (ver fig. 4.2).

Figura 4.1: Representação de uma seção do espaço de entrada N -dimensional mostrando o vetor $\vec{w}^*$ perpendicular ao hiperplano D , onde a linha tracejada representa sua projeção no plano da figura.

Figura 4.2: O plano $\vec{w}$ - $\vec{w}^*$, onde C e D são os seus respectivos hiperplanos perpendiculares e A e B as regiões onde $\vec{w}$ e $\vec{w}^*$ dão respostas diferentes a um mesmo exemplo.

O aprendizado no Perceptron pode ser visto por meio de uma interpretação geométrica, que está representada na fig. 4.2, onde temos um sub-espaço do espaço de entrada com os vetores $\vec{w}$ e $\vec{w}^*$, os quais estão sobre uma hipersuperfície e são respectivamente perpendiculares aos $(N - 1)$ -dimensional hiperplanos C e D , estando num ângulo θ , sendo as projeções de C e D no plano $\vec{w}$ - $\vec{w}^*$ representadas por linhas tracejadas. Então uma classificação incorreta de um exemplo $\vec{\xi}$ pela rede aluno $\vec{w}$ ocorrerá quando o exemplo $\vec{\xi}$ cair na região A ou B , a qual tem seu ângulo diminuído a medida que $\vec{w}$ se aproxima de $\vec{w}^*$, de forma que o processo de aprendizagem pode ser visto como várias mudanças em $\vec{w}$ na direção de $\vec{w}^*$, mudanças essas ajustadas de

acordo com os exemplos $\vec{\xi}$ do conjunto de treinamento. Frente a tal representação é fácil ver que o erro de generalização¹ da rede aluno $\vec{w}$ será proporcional à soma dos ângulos A e B, ou seja,

$$e_g = \frac{2\theta}{2\pi} = \frac{1}{\pi} \cos^{-1}(\vec{w} \cdot \vec{w}^* / |\vec{w}|) = \frac{1}{\pi} \cos^{-1}(\rho). \quad (4.3)$$

Onde definimos uma nova variável $\rho = \vec{w} \cdot \vec{w}^* / |\vec{w}|$ que corresponde a sobreposição de $\vec{w}$ em $\vec{w}^*$. Surpreendentemente essa equação para e_g também vale para $N \rightarrow \infty$.

Dentro da teoria apresentada no capítulo 2 vimos que uma forma de encontrar uma estimativa do vetor da rede professor $\vec{w}^*$ é a encontrar a distribuição $P(\vec{w})$ que descreve o espaço dos possíveis candidatos a terem gerado o conjunto de treinamento D_t , e que cada exemplo $\vec{\xi}$ traz uma nova restrição na espaço dos possíveis candidatos, conhecido como *espaço de versões* V. De forma que a proposta da inferência Bayesiana consiste justamente em estimar medidas da geometria do espaço de versões, estimando os momentos da distribuição verdadeira $P(\vec{w}^*)$, por exemplo, o do centro de massa de V, ou seja, $\hat{w}$, que como vimos no final do capítulo 2 tende assintoticamente ao verdadeiro ponto fixo w^* . Vimos também que, para melhorar o desempenho dos algoritmos on-line, é necessário a introdução de um fator de ajuste η que decaía durante o aprendizado e que tal fator de ajuste aparece naturalmente por meio no algoritmo Bayesiano e que corresponde a covariância, o segundo momento da distribuição, que nos dá uma idéia de sua largura. Essa restrição no espaço de versões causada pelas informações $\vec{\xi}$ também pode ser vista nessa interpretação geométrica descrita nos últimos parágrafos.

Na figura 4.3 vemos que ambos os exemplos $\vec{\xi}_1$ e $\vec{\xi}_2$ são classificados positivamente ($\sigma = +1$) o que indica que $\vec{w}^*$ está na região entre os hiperplanos A e C, do mesmo lado dos exemplos, ou seja, o nosso espaço de versões é restringido para essa região por meio das informações contidas nos exemplos 1 e 2.

Infelizmente não podemos concluir a tempo a nossa simulação referente às equações de atualização descritas no apêndice C, que correspondem a considerarmos os coeficientes B_M com $|M| \leq 4$, de forma que só discutiremos agora o resultado obtido para uma parametrização por uma simples Gaussiana², que nos levam ao algoritmo:

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \sqrt{\frac{2}{N\pi\hat{x}}} \tilde{C}_{ii} \xi_i \sigma \frac{e^{-\tau^2/2\hat{x}}}{\text{erfc}(-\tau/\sqrt{2\hat{x}})}, \quad (4.4)$$

¹A probabilidade de $\vec{w}$ classificar incorretamente um novo exemplo $\vec{\xi}$

²Que é equivalente a considerar apenas o primeiro termo da expansão 2.35.

Figura 4.3: Restrições no espaço de versões.

$$\tilde{C}_{ii}(t+1) = \tilde{C}_{ii}(t) - \frac{1}{N} \frac{\tilde{C}_{ii}^2(t) \xi_i^2 e^{-\tau^2/2\tilde{x}}}{\tilde{x} \operatorname{erfc}(-\tau/\sqrt{2\tilde{x}})} \left(\tau \sqrt{\frac{2}{\pi\tilde{x}}} + \frac{2e^{-\tau^2/2\tilde{x}}}{\pi \operatorname{erfc}(-\tau/\sqrt{2\tilde{x}})} \right). \quad (4.5)$$

Usando esse algoritmo encontramos o comportamento assintótico em α para o erro de generalização que se aproxima, para $N \rightarrow \infty$, à $e_g = 0,88\alpha^{-1}$, onde definimos $\alpha = t/N$ sendo t o número de exemplos utilizado no treinamento, ou seja, o mesmo comportamento já observado por Solla e Winther [22] para o algoritmo Bayesiano on-line para o Perceptron sem a utilização de uma função potencial³, e que também foi obtido pelo método variacional on-line [19]. O comportamento assintótico do erro de generalização para o algoritmo Bayesiano off-line [23] é dado por $e_g = 0,44\alpha^{-1}$, exatamente o mesmo expoente de queda e a metade do coeficiente do algoritmo on-line. Uma boa expectativa para a utilização da expansão 2.35 é que o coeficiente do erro de generalização caminhe em direção à 0,44 a medida que sejam acrescentados mais termos na parametrização da distribuição posterior além do termo de B_0 . Tal implementação serviria apenas para apontar melhores esclarecimentos sobre as relações on-line/off-line como, por exemplo, esse fator de dois entre a aprendizagem ótima on-line e a off-line, visto que a complexidade do algoritmo cresce razoavelmente para o caso não Gaussiano: para cada novo exemplo precisamos atualizar a matriz de covariância C , da $\mathcal{O}(N^2)$, a matriz A também da $\mathcal{O}(N^2)$ e cada conjunto de hiperparâmetros B_M 's que para $|M|$ fixo nos dá $(|M| + N - 1)!/(|M|!(N - 1)!)$

³Semelhanamente à seção 2.1 e ao capítulo 3.

hiperparâmetros, o qual possui⁴ ordem de $N^{|M|}$. Assim temos $\mathcal{O}(N^2t)$ para $|M| \leq 2$ e $\mathcal{O}(N^{|M|}t)$ para $|M| \geq 2$ do número de iterações necessárias para o treinamento on-line com t exemplos, enquanto que para o treinamento off-line temos $\mathcal{O}(\hat{N}Nt)$, sendo $\hat{N}$ o número de iterações necessárias para obter soluções com o grau de precisão desejado, o qual em geral é da ordem de N .

A análise da complexidade computacional para o Perceptron simples com pesos sinápticos contínuos mostra que, para uma parametrização por uma Gaussiana pura, encontramos no algoritmo on-line a mesma complexidade do off-line mas com um desempenho inferior. Se incluirmos termos não Gaussianos teremos uma complexidade maior, pondo em *cheque* o principal argumento de se utilizar algoritmos on-line: a diminuição da complexidade no aprendizado, ou melhor, sua velocidade. Contudo, tal perda não é verdade frente a aproximações no algoritmo. Mostramos que tomando a matriz de covariância diagonal obtemos o mesmo desempenho que no caso sem aproximações [22], mas a complexidade cai para $\mathcal{O}(Nt)$, mesmo com a inclusão de termos não Gaussianos⁵. Essa aproximação porém nem é sempre válida para outras distribuições, onde se usa uma matriz de covariância que não se torna diagonal assintoticamente.

⁴Isso para $N \gg |M|$ e sem levar em consideração nenhuma redução por simetria ou pelos termos nulos B_{1_k} ou B_{2_k}

⁵No caso diagonal cada $B_M = \prod_{i=1}^N B_{m_i}$.

5

CONCLUSÕES E PERSPECTIVAS

Nessa dissertação nos ocupamos em estudar inferência através do cenário de aprendizado supervisionado. Em particular olhamos para uma rede de neurônios que aprende a partir da informação contida nos exemplos que formam um conjunto de treinamento D_t . Este é formado por pares ordenados que contém informações sobre vetores de entrada e suas respectivas saídas. Nesse cenário o processo de aprendizado ocorre pela determinação de uma configuração para os pesos sinápticos da rede que satisfaçam os vínculos do conjunto de treinamento, ao menos de forma aproximada. Isto é, para cada informação de entrada ξ_μ do conjunto D_t a rede de neurônios retorne a mesma saída σ_μ também de D_t , com probabilidade maior do que meio. Embora variações intermediárias também possam ser estudadas, tal aprendizado pode ser feito praticamente de duas formas, no que tange às propriedades dinâmicas:

- Treinamento off-line: durante o processo de treinamento o sistema extrai todas as informações possíveis, acessíveis ao algoritmo, por meio de um processo de minimização de uma função energia ou de uma distribuição posterior que envolve todos os exemplos contidos em D_t para só depois fazer estimativas sobre os parâmetros w 's. O sistema entra em equilíbrio termodinâmico após o treinamento.
- Treinamento on-line, onde não é dado tempo para o sistema entrar em equilíbrio, fazendo estimativas sobre os parâmetros w 's a medida que as informações chegam, por meio de uma minimização de uma função energia ou distribuição posterior envolvendo apenas o exemplo mais recente.

É intuitivo que algoritmos obtidos pela estratégia off-line tenham um melhor desempenho que seus análogos obtidos por meio de um treinamento on-line, visto que no último não se utiliza de todas as informações disponíveis a cada passo do

treinamento, usando uma mesma informação apenas uma vez na atualização dos pesos w 's e descartando-a em seguida, enquanto que nos algoritmos off-line todas as informações são mantidas durante o treinamento, o que leva a uma extração de todas as informações disponíveis no conjunto D_t . Por outro lado também é conhecido que algoritmos off-line são mais custosos, do ponto de vista de cálculo e processamento computacional, que algoritmos on-line, isso também devido ao mesmo motivo: que por sua própria definição somente um passo da dinâmica de minimização da energia¹ é realizado para cada exemplo no algoritmo on-line, enquanto que no off-line a minimização da energia é feita sobre todos os exemplos em conjunto.

Uma outra característica interessante da aprendizagem on-line é a sua adaptação para sistemas em que a regra de classificação $\vec{w}^*$ muda durante o treinamento. Como os exemplos de treinamento são apresentados sequencialmente fazendo estimativas a cada apresentação e descartando o exemplo em seguida, não temos uma dependência tão forte aos primeiros conceitos aprendidos de forma que a rede pode se adaptar a nova regra de acordo com quanto adaptativo for os estimadores. Tal readaptação já não é tão simples para algoritmos off-line, uma vez que todas as informações são processadas em paralelo.

Ao irmos além da aproximação de Oppen, incluindo correções não gaussianas, pretendemos entender alguns aspectos de um problema extremamente difícil, que é lidar com as conexões entre sistemas em equilíbrio e fora dele.

Nesta dissertação escrevemos uma série formal que descreve a atualização da distribuição posterior quando chega nova informação. Estudamos o caso em que a informação a priori que temos sobre os pesos nos leva a concluir que estes são variáveis reais que podem estar entre menos e mais infinito, de forma que escolhemos uma função Gaussiana para a parametrização e aproximação on-line proposta por Oppen. Para minimizar o erro introduzido por tal aproximação, a escolha (projeção) da nova Gaussiana é feita de maneira a minimizar a distância de Kullback-Leibler entre a candidata a nova Gaussiana e a distribuição construída a partir da distribuição anterior (com t exemplos) e o novo exemplo ($t+1$) por meio da regra de Bayes. Por meio de uma expansão em série da função posterior podemos seguir a mesma parametrização e aproximação de Oppen, mas agora incluindo extensões não Gaussianas. Devido a informação inicial concluímos que a série adequada para tal caso (variáveis reais distribuídas simetricamente) é a dos polinômios de Hermite, onde

¹Seja por meio do gradiente de um potencial ou pelo valor esperado da distribuição posterior do exemplo mais recente.

cada termo da série possui uma Gaussiana, e como estamos interessados em estudar problemas em N dimensões, é necessário usarmos o formalismo de Hagedorn para expansões em série de Hermite generalizadas a mais de uma dimensão. Em seguida, estudamos alguns exemplos de casos simples, com o objetivo de entender o efeito da inclusão desses termos de correção. Os resultados são bastante preliminares e seu estudo sistemático será deixado para uma fase posterior. Os resultados obtidos até o momento incluem os descritos a seguir.

No capítulo 3 estudamos as propriedades de um sistema modelo muito simples que aprende uma regra que ou está corrompida por ruído ou varia no tempo. Estudamos os efeitos de variações periódicas e também o caso em que a regra muda bruscamente no tempo. Também estudamos os efeitos de fazer erros na determinação do modelo, neste caso erros na estimativa do nível de ruído. A figura 3.5 mostra o expoente de decaimento do erro quadrático médio como função dos níveis reais e estimados de ruído. É interessante notar a semelhança com os diagramas de robustez encontrados por Copelli et al [24], representado qualitativamente na figura 5.1.

Figura 5.1: Diagrama de robustez em função dos níveis reais ϵ_0 e estimados ϵ de ruído na presença de ruído multiplicativo.

Naquele trabalho foi dado o diagrama de fases dinâmico de robustez para aprendizagem em máquinas multicamadas (comitê, paridade). Foram encontradas três fases: uma onde a estimativa do ruído é pouco errada e assintoticamente o sistema aprende, que seria a região em torno da diagonal ($\varepsilon = \varepsilon_0$) e limitada pela curva de linha cheia. Outra, onde o nível de ruído é bem maior que a estimativa e o sistema aprende parcialmente, que corresponde a região entre a curva de linha cheia e a reta tracejada. E a terceira fase, na região abaixo ($\varepsilon \leq \varepsilon_0$) da linha tracejada, onde o erro na estimativa é tão grande que nenhum tipo de aprendizagem é possível. No caso que aqui estudamos há somente mudanças no expoente de decaimento do erro que seguem qualitativamente o diagrama geral de robustez. Reconhecendo-se por comparação apenas as regiões em que $\varepsilon \geq \varepsilon_0$, uma superestimação, onde há um bom desempenho no aprendizado e uma subestimação que leva a perda de generalização, ou melhor de aprendizado, que corresponderia a região $\varepsilon_0 \leq \varepsilon$ abaixo da linha tracejada para a figura 5.1. Enfatizamos que essa comparação é feita apenas de forma qualitativa, uma vez que o diagrama da figura 5.1 foi obtido no limite termodinâmico, no qual o número de dimensões do vetor de peso sináptico é muito grande ($N \rightarrow \infty$), sendo interessante então reconhecer semelhanças de comportamento mesmo para situações distintas² e principalmente pelo fato de ter sido mostrado que dentro do limite termodinâmico o diagrama 5.1 é válido para diferentes estruturas de rede.

Em seguida olhamos para dois tipos de mudanças da regra. No primeiro, o professor descreve um caminho contínuo com pequenos passos que oscilam harmonicamente e com passos aleatórios simples, enquanto que no segundo a regra (professor) é constante por partes, sofrendo mudanças bruscas de forma a ficar descorrelacionada com a regra anterior. Como resultado observamos que na aprendizagem Bayesiana, o algoritmo obtido tem um bom desempenho para situações em que a rede professor é estacionária mas quando a regra muda durante o treinamento já não obtemos um desempenho tão bom, tanto pior quanto melhor tenha sido aprendido a regra anterior à mudança. Isso naturalmente vem em concordância com a Teoria Bayesiana da inferência: o seu desempenho realmente é comprometido pela distribuição prévia utilizada. Na simulação do capítulo 3 começamos com uma distribuição prévia igual a distribuição dos pesos sinápticos sendo a estimativa $\hat{w}$ inicial nula e a variância c inicial unitária. Durante o treinamento aprendemos que a variância está intima-

² $N = 1$ para o nosso caso e $N \rightarrow \infty$ para Copelli et al.

mente ligada com a confiança da rede aluno $\hat{w}$, ou seja, quanto mais o aluno aprende (se aproxima da regra verdadeira w_0) sua confiança aumenta, diminuindo assim a sua aceitação de novas informações diferentes da que tem aprendido, o que em termos da distribuição posterior equivale a uma diminuição de sua largura (a covariância c). Por isso é importante usar uma distribuição prévia com uma largura grande ($c(0) = 1$), para que o aluno não comece o aprendizado preconceituadamente, ou melhor, preso a estimativas iniciais $\hat{w}(0)$ do professor que podem estar totalmente erradas. Isso foi nitidamente observado no treinamento com mudanças bruscas (ver figura 3.10): quando o nível de ruído é baixo o aluno aprende muito bem a regra inicial porém quando essa muda bruscamente ele já não consegue aprender pois sua confiança em si mesmo já está muito alta ($c \ll 1$); enquanto que para níveis maiores de ruído como o aluno não consegue aprender tão bem, a sua confiança não fica tão alta ($c \simeq 1$) de forma que ele consegue aprender uma nova regra, mesmo que seja com um baixo desempenho devido ao ruído. Quando a regra é mudada, a última atualização da distribuição posterior será a nova distribuição prévia, de forma que estaremos começando um novo aprendizado com uma espaço de *versões* bastante restrito, centrado estreitamente na estimativa $\hat{w}(t - 1)$ da regra anterior, descartando os novos prováveis candidatos nas estimações seguintes por meio da equações 2.6 e 2.7.

Figura 5.2: Primeira e segunda derivada do potencial on-line Bayesian para o Perceptron.

Por último discutiremos mais um pouco os resultados encontrados no capítulo 4, onde tratamos do algoritmo Bayesiano on-line com potencial para o Perceptron.

Na figura 5.2 temos as curvas de correção para os parâmetros w_i (linha tracejada) e dos hiperparâmetros C_{ii} (linha cheia) em função da estabilidade posterior τ da rede na presença de ruído e a menos de alguns fatores, ver eqs. 4.4 e 4.5, onde identificamos três regiões limitadas pelos máximos das duas curvas. O limite de ruído nulo equivale a deslocarmos os picos de máximo para $\tau \rightarrow \infty$, e $\tau > 0$ significa que ambos aluno e professor concordam com a resposta à referente entrada enquanto que para $\tau < 0$ discordam. Analisemos primeiro a curva referente ao termo de correção Δw . Na região III temos que as correções do aluno são tanto maiores quanto for o erro ($\tau < 0$) e praticamente nulas quando acerta ($\tau > 0$), ressaltando que para grandes magnitudes de τ o aluno ignora a informação ($\Delta w \simeq 0$). Já na região II e I o aluno começa a perder a credibilidade no professor e suas correções começam a diminuir, chegando a serem nulas novamente quando a discrepância entre a sua estimativa da saída e a resposta do professor é muito grande. Para as correções na estimativa da covariância vemos um comportamento diferente na região III: o aluno sempre diminui a sua correção a menos que $\tau \gg 0$, como se esse estivesse muito confiante no seu aprendizado, mesmo quando suas respostas diferem do professor ($\tau < 0$), enquanto que na região II o aluno entra numa espécie de fase duvidosa e começa a aumentar a sua aceitação que cai na região I chegando novamente ao desprezo, ou melhor, correção nula, quando a magnitude de τ é muito grande³. Ao contrário das duas primeiras regiões (I,II) a correção dos pesos w_i 's e covariâncias c_{ij} na região III se comportam de maneira diferente: enquanto temos um aumento em Δw_i encontramos uma diminuição ΔC_{ii} . Então o que realmente estaria acontecendo, *haveria uma sobreposição de uma estimativa sobre a outra? Ou elas se compensariam?* Um possível esclarecimento de tais perguntas podem ser dadas sob um ponto de vista *geométrico*: cada momento da distribuição posterior está relacionado com a geometria do *espaço de versões* (EV), que corresponde ao espaço dos possíveis candidatos a professor. Assim, o centro de massa do EV nos daria uma estimativa dos pesos sinápticos que generalizaria o problema tão bem quanto um algoritmo Bayesiano, e a largura média do EV se relacionaria com a matriz de covariância C , o que está de acordo com o que aprendemos sobre esse hiperparâmetro, principalmente no capítulo 3. De forma que, embora na região III

³Tal comportamento é consistente com a idéia [10] que ajustes adaptativos das correções devem depender da estimativa do erro de generalização.

estejamos aumentando a nossa correção do centro de massa por meio de $d\mathcal{E}(\tau)$, ainda permanecemos bastante confiante com relação a largura do EV.

Qualquer estratégia de aprendizagem on-line (sem ruído) será melhor sucedida quanto mais informações guardar sobre a história do espaço de versões trazidas pelos exemplos, chegando no limite da aprendizagem off-line quando toda a informação disponível sobre EV⁴ no conjunto de treinamento for armazenada, e mesmo na presença de ruído podemos ainda manter a idéia do EV ao nos referir a distribuição posterior. Assim podemos considerar algoritmos de aprendizado que guardem essas informações sobre EV, se não no todo (off-line) ao menos parcialmente (on-line), de forma que as nossas perspectivas são as possibilidades abertas por tal caminho, como exemplo, o Perceptron em N-dimensões: *será que realmente teremos ganhos ao guardarmos mais informações sobre a geometria do espaço de versões?*⁵ Como vimos a complexidade dos algoritmos pode aumentar bastante ao mantermos mais informações sobre a forma do EV, o que já iria contra a idéia natural da aprendizagem on-line, mas por outro lado pode nos trazer mais esclarecimentos sobre as relações dessas duas distintas estratégias de aprendizagem.

⁴EV é o suporte da distribuição posterior exata.

⁵Para tal pergunta pretendemos continuar a implementação da dinâmica descrita no apêndice C.

A

DISTÂNCIA DE KULLBACK-LEIBLER

Nesse apêndice o nosso único objetivo será demonstrar que se considerarmos $p(\vec{w}|par_{t+1})$ Gaussiana, então minimizar

$$D_{KL}(p(\vec{w}|par_t, y_{t+1})||p(\vec{w}|par_{t+1})) = \int p(\vec{w}|par_t, y_{t+1}) \ln \left(\frac{p(\vec{w}|par_t, y_{t+1})}{p(\vec{w}|par_{t+1})} \right),$$

será equivalente a impor que as distribuições $p(\vec{w}|par_t, y_{t+1})$ e $p(\vec{w}|par_{t+1})$ tenham os mesmos primeiros momentos.

Para tanto vamos representar $p(\vec{w}|par_t, y_{t+1})$ por “ P ” e $p(\vec{w}|par_{t+1})$ por “ G ”, sendo $G = e^{-(w-\hat{w})^2/2C} / \sqrt{2\pi C}$.

$$D_{KL}(P||G) = \int P \ln P - \int P \ln G. \quad (\text{A.1})$$

Na eq. A.1 a distribuição P é fixa e apenas G pode ser ajustada, de forma que a variação que iremos fazer em A.1 será apenas com relação aos parâmetros de G , que no caso de ser uma Gaussiana constam de seu centro $\hat{w}$ e sua covariância C , logo

$$\delta D_{KL}(P||G) = - \int P \delta \ln G = - \int P \frac{1}{G} \delta G, \quad (\text{A.2})$$

onde

$$\delta G = \frac{\delta G}{\delta \hat{w}} \delta \hat{w} + \frac{\delta G}{\delta C} \delta C. \quad (\text{A.3})$$

Calculemos então cada variação:

$$\frac{\delta G}{\delta \hat{w}} = \frac{w - \hat{w}}{C} G; \quad (\text{A.4})$$

$$\frac{\delta G}{\delta C} = \frac{(w - \hat{w})^2}{2C^2} G - \frac{1}{2C} G. \quad (\text{A.5})$$

Substituindo as eqs. A.4 e A.5 na eq. A.2 teremos que¹

$$\delta D_{KL}(P||G) = - \int P \frac{w - \hat{w}}{C} \delta \hat{w} - 2 \int PC((w - \hat{w})^2 - C) \delta C = 0, \quad (\text{A.6})$$

de forma que devemos ter $\int Pw = \hat{w}$ e $\int P(w - \hat{w})^2 = C$ para que a eq. A.6 seja independente das variações $\delta \hat{w}$ e δC , ou seja, se G for uma Gaussiana centrada em $\hat{w}$ e com covariância C , minimizar a eq. A.1 será equivalente a igualarmos os dois primeiros momentos de P e G . Para o caso de dimensões maiores que um podemos diagonalizar a matriz C de forma que $G = \prod_{i=1}^N e^{-(w_i - \hat{w}_i)^2 / 2C_{ii}} / \sqrt{2\pi C_{ii}}$ transformando o problema num unidimensional a semelhança do que fizemos nesse apêndice.

¹Lembre-se que queremos minimizar D_{KL} e por isso fazemos $\delta D_{KL} = 0$.

B

POLINÔMIOS DE HERMITE.

Longe de expormos uma boa revisão sobre os polinômios de Hermite e as soluções do oscilador harmônico o nosso objetivo nesse apêndice é de apenas revermos algumas propriedades usadas no decorrer do capítulo 2. Para o leitor que realmente deseja fazer uma revisão completa sugerimos as seguintes referências:

- Arfken, G., (1966) Mathematical methods for physicist, *Academic Press, New York*;
- Cohen-Tannoudji, C., Diu, B. and Laloë, F., (1977) Quantum mechanics - vol I, *Jonh Wiley*.

Uma vez que sabemos que $\varphi(x) = \text{cte } e^{-x^2/2}$ é solução da equação diferencial unidimensional:

$$(x + \frac{d}{dx})\varphi(x) = 0, \quad (\text{B.1})$$

o que é facilmente provado e onde “cte” é uma constante de normalização, queremos encontrar a solução da equação diferencial 2.28:

$$\mathcal{A}(A, \vec{a})\varphi_0(A, \vec{a}, \vec{x}) = 0, \quad (\text{B.2})$$

onde o operador $\mathcal{A}$ tem suas componentes definidas como sendo

$$\mathcal{A}_j(A, a) = \frac{1}{\sqrt{2}} \sum_n \left(A_{jn}^{-1}(x_n - a_n) + A_{jn} \frac{\partial}{\partial x_n} \right).$$

A eq. B.2, que na verdade abusa um pouco da notação, significa que: se todas as direções estão com ocupação nula ($M = 0$), devemos encontrar em resultado nulo quando aplicarmos qualquer componente do operador de aniquilação $\mathcal{A}_j(A, a)$.

Mas encontrar uma solução para B.2 não parece ser uma tarefa fácil, pois cada componente $\mathcal{A}_j(A, a)$ atua não somente em sua direção mas como em todas as outras, de forma que devemos encontrar um referencial no qual os operadores $\mathcal{A}_j(A, a)$ atuem apenas em suas respectivas direções. Isso é conseguido fazendo a seguinte transformação no vetor $\vec{x}$:

$$\vec{z} = A^{-1}(\vec{x} - \vec{a}), \quad (\text{B.3})$$

o que transforma $\mathcal{A}_j(A, a)$ em

$$\mathcal{A}_j(A, a) = z_j + \frac{\partial}{\partial z_j}.$$

Com isso a eq. B.2 se transforma na B.1 em cada direção, o que nos leva a solução:

$$\varphi_0(A, \vec{a}, \vec{x}) = \frac{\pi^{-\frac{N}{4}}}{\sqrt{\det A}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{a})A^{-2}(\vec{x} - \vec{a})\right), \quad (\text{B.4})$$

onde o fator $\sqrt{\det A}$ vem justamente devido ao jacobiano da mudança e $\pi^{-\frac{N}{4}}$ da normalização de $\varphi_0(A, \vec{a}, \vec{x})$.

Agora queremos construir os demais estados $\varphi_k(A, \vec{a}, \vec{x})$ aplicando o operador de criação $\mathcal{A}^\dagger(A, a)$ recursivamente em $\varphi_0(A, \vec{a}, \vec{x})$, de forma que os tenhamos ortonormais. Para isso, vejamos uma simples propriedade que irá nos ajudar adiante:

$$(\mathcal{A}\mathcal{A}^\dagger - \mathcal{A}^\dagger\mathcal{A})F(z) = \frac{1}{2}(z + \partial_z)(z - \partial_z)F(z) - \frac{1}{2}(z - \partial_z)(z + \partial_z)F(z) = F(z),$$

ou seja: “ $\mathcal{A}\mathcal{A}^\dagger - \mathcal{A}^\dagger\mathcal{A} = 1$ ”.

Definamos então que $\varphi_{k+1} = c_{k+1}\mathcal{A}^\dagger\varphi_k$ e calculemos quanto deve ser cada coeficiente c_{k+1} para que tenhamos $\int \varphi_{k+1}\varphi_{k+1} = 1$. Começemos então pelos primeiros estados:

$$\begin{aligned} \int \varphi_1\varphi_1 &= \int |c_1|^2\varphi_0\mathcal{A}\mathcal{A}^\dagger\varphi_0 \\ &= |c_1|^2 \int \varphi_0(\mathcal{A}^\dagger\mathcal{A} + 1)\varphi_0 = 1, \end{aligned} \quad (\text{B.5})$$

ou seja, $c_1 = 1$ e consecutivamente $\varphi_1 = \mathcal{A}^\dagger\varphi_0$. Vejamos agora para $\varphi_2 = c_2\mathcal{A}^\dagger\varphi_1$:

$$\begin{aligned} \int \varphi_2\varphi_2 &= \int |c_2|^2\varphi_1\mathcal{A}\mathcal{A}^\dagger\varphi_1 \\ &= |c_2|^2 \int \varphi_1(\mathcal{A}^\dagger\mathcal{A} + 1)\varphi_1 = 1, \end{aligned} \quad (\text{B.6})$$

portanto:

$$\varphi_2 = \frac{1}{\sqrt{2}} \mathcal{A}^\dagger \varphi_1 = \frac{(\mathcal{A}^\dagger)^2}{\sqrt{2}} \varphi_0,$$

de forma que, se seguirmos adiante pelo mesmo caminho, encontraremos:

$$\varphi_{n+1} = \frac{1}{\sqrt{n}} \mathcal{A}^\dagger \varphi_n. \quad (\text{B.7})$$

A eq. B.7 pode ser ainda escrita na forma

$$\begin{aligned} \varphi_{n+1} &= \frac{1}{\sqrt{2^n n!}} (z - \partial_z)^n \varphi_0 \\ &= \frac{1}{\sqrt{2^n n!}} H_n \varphi_0, \end{aligned} \quad (\text{B.8})$$

sendo H_n os polinômios de Hermite e gerados, segundo sua definição em B.8, por:

$$H_n \varphi_0 = (z - \partial_z)^n \varphi_0. \quad (\text{B.9})$$

O que nos leva a relação

$$H_{n+1} \varphi_0 = (z - \partial_z)^{n+1} \varphi_0 = (z - \partial_z) H_n \varphi_0 = z H_n \varphi_0 - \partial_z (H_n \varphi_0),$$

que multiplicando por $(H_n \varphi_0)$ resultará em:

$$\begin{aligned} H_{n+1} \varphi_0^2 &= z H_n \varphi_0^2 - \varphi_0 \partial_z (H_n \varphi_0) \\ &= -H_n \varphi_0 \partial_z \varphi_0 - \varphi_0 \partial_z (H_n \varphi_0) \\ &= -\partial_z (H_n \varphi_0^2). \end{aligned} \quad (\text{B.10})$$

A qual recursivamente nos levará a famosa relação de Rodrigues para os polinômios de Hermite:

$$H_n = \varphi_0^{-2} (-\partial_z)^n \varphi_0^2. \quad (\text{B.11})$$

Por último vamos demonstrar a relação

$$\partial^k H_n = 2^k \frac{n!}{(n-k)!} H_{n-k} \delta_{n \geq k}. \quad (\text{B.12})$$

Para isso usemos a fórmula de Rodrigues:

$$\begin{aligned} H_{n+1} \varphi_0^2 &= (-\partial_z)^n (H_1 \varphi_0^2) \\ &= (-1)^n \sum_{s=0}^n \frac{n!}{s!(n-s)!} (\partial_z^s H_1) (\partial_z^{n-s} \varphi_0^2) \\ &= H_1 (-\partial_z)^n \varphi_0^2 + n (\partial_z H_1) (\partial_z^{n-1} \varphi_0^2) \\ &= H_1 H_n \varphi_0^2 - 2n H_{n-1} \varphi_0^2, \end{aligned} \quad (\text{B.13})$$

mas por outro lado

$$\begin{aligned} H_{n+1}\varphi_0^2 &= (-\partial_z)(H_1\varphi_0^2) \\ &= -\partial_z H_n\varphi_0^2 + H_n H_1\varphi_0^2, \end{aligned} \tag{B.14}$$

de forma que substituindo a eq. B.13 em B.14 encontraremos

$$\partial_z H_n = 2n H_{n-1}, \tag{B.15}$$

que facilmente pode ser generalizada por recursão à eq B.12.

C

EQUAÇÕES DE ATUALIZAÇÃO.

Aqui estaremos apenas expondo as equações de atualização para o sistema descrito no capítulo 4, o Perceptron N-dimensional. Abaixo temos umas tabelas para os coeficientes γ e algumas abreviações que usamos para simplificar a escrita. Lembramos apenas que no presente sistema a matriz de covariância C é diagonal, o que acarreta em

$$\Gamma_{ML} = \prod_{i=1}^N \gamma_{m_i l_i}, \quad \Lambda_{ij} = \Lambda_{ii} \delta_{ij} \quad \text{e} \quad \Delta' \hat{w}_i(t) = A_{ii}^{-1}(t+1) \Delta \hat{w}_i(t).$$

Coeficientes de Translação Temporal.	
$\gamma_{3,0}$	$4\Delta' \hat{w}_i(3 - 3\Lambda_{ii}^2 - 2\Delta' \hat{w}_i^2)$
$\gamma_{3,1}$	$12\Lambda_{ii}(2\Delta' \hat{w}_i^2 - 1 + \Lambda_{ii}^2)$
$\gamma_{3,2}$	$-48\Lambda_{ii}^2 \Delta' \hat{w}_i$
$\gamma_{3,3}$	$48\Lambda_{ii}^3$
$\gamma_{4,0}$	$16\Delta' \hat{w}_i^4 - 48\Delta' \hat{w}_i^2 + 12 - 24\Lambda_{ii}^2 + 48\Delta' \hat{w}_i^2 \Lambda_{ii}^2 + 12\Lambda_{ii}^4$
$\gamma_{4,1}$	$32\Lambda_{ii} \Delta' \hat{w}_i(3 - 3\Lambda_{ii}^2 - 2\Delta' \hat{w}_i^2)$
$\gamma_{4,2}$	$96\Lambda_{ii}^2(2\Delta' \hat{w}_i^2 - 1 + \Lambda_{ii}^2)$
$\gamma_{4,3}$	$-384\Lambda_{ii}^3 \Delta' \hat{w}_i$
$\gamma_{4,4}$	$384\Lambda_{ii}^4$

Abreviações: $dn = \partial_\tau^n \exp(-\varepsilon_x(\tau))$

d0	$\operatorname{erfc}(-\tau/\sqrt{2x})$
d1	$\sqrt{\frac{2}{\pi x}} e^{-\frac{\tau^2}{2x}}$
d2	$-\tau x^{-3/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d3	$(\frac{\tau^2}{x} - 1) x^{-3/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d4	$(3 - \frac{\tau^2}{x}) \tau x^{-5/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d5	$(3 - 6\frac{\tau^2}{x} + \frac{\tau^4}{x^2}) x^{-5/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d6	$(-15 + 10\frac{\tau^2}{x} - \frac{\tau^4}{x^2}) \tau x^{-7/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d7	$(-15 + 45\frac{\tau^2}{x} - 15\frac{\tau^4}{x^2} + \frac{\tau^6}{x^3}) x^{-7/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$
d8	$(105 - 105\frac{\tau^2}{x} + 21\frac{\tau^4}{x^2} - \frac{\tau^6}{x^3}) \tau x^{-9/2} \sqrt{\frac{2}{\pi}} e^{-\frac{\tau^2}{2x}}$

Observamos que na tabela dos coeficientes γ_{ij} deve-se ler $\Delta' \hat{w}_i = \Delta' \hat{w}_i(t)$ e $\Lambda_{ii} = A^{-1}(t+1)A(t)$.

A seguir veremos as equações de atualização para tal sistema, aonde abreviaremos a função de partição por Ξ :

$$\Xi = \operatorname{erfc}(-\tau/\sqrt{2x}) + \sum_{i=1}^N \frac{B_{3i}}{48} (A_{ii} \sigma \xi_i / \sqrt{N})^3 d3 + \sum_{i=1}^N \frac{B_{4i}}{384} (A_{ii} \sigma \xi_i / \sqrt{N})^4 d4.$$

Enfatizamos que devido a diagonalização temos a equivalência $B_M = \prod_{i=1}^N B_{m_i}$ e que, quando não especificado, estamos tomando as variáveis¹ no tempo t .

Atualização dos pesos sinápticos.

$$\begin{aligned} \hat{w}_i(t+1) = & \hat{w}_i + \sigma \xi_i \frac{C_{ii}}{\sqrt{N}} \left\{ d1 + \sum_{j=1}^N \frac{B_{3j}}{48} (\sigma \xi_j \sqrt{2C_{jj}/N})^3 d4 \right. \\ & \left. + \sum_{j=1}^N \frac{B_{4j}}{384} (\sigma \xi_j \sqrt{2C_{jj}/N})^4 d5 \right\} / \Xi \\ & + \sqrt{2C_{ii}} \left\{ \frac{B_{3i}}{8N} \xi_i^2 C_{ii} d2 + \frac{B_{4i}}{96} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \right\} / \Xi \end{aligned} \quad (\text{C.1})$$

¹ $B, A, \sigma, \xi \dots$

Atualização dos hiperparâmetros: C.

$$\begin{aligned}
C_{ii}(t+1) = & C_{ii} + \xi_i^2 \frac{C_{ii}^2}{N\Xi} d2 + \frac{C_{ii}}{24\Xi} \left\{ \frac{1}{4} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 B_{3i} d5 \right. \\
& + 6(\sigma \xi_i \sqrt{2C_{ii}/N})^3 B_{3i} d3 + 6(\sigma \xi_i \sqrt{2C_{ii}/N}) B_{3i} d1 \\
& + \frac{1}{4} \sum_{j \neq i} (\sigma \xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^2 B_{3j} d5 \\
& + \frac{1}{32} (\xi_i \sqrt{2C_{ii}/N})^6 B_{4i} d6 + (\xi_i \sqrt{2C_{ii}/N})^4 B_{4i} d4 \\
& \left. + \frac{3}{2} (\xi_i \sqrt{2C_{ii}/N})^2 B_{4i} d2 + \frac{1}{32} \sum_{j \neq i} (\xi_j \sqrt{2C_{jj}/N})^4 (\xi_i \sqrt{2C_{ii}/N})^2 B_{4j} d6 \right\} \\
& - (\hat{w}_i(t+1) - \hat{w}_i(t))^2 \tag{C.2}
\end{aligned}$$

Atualização dos hiperparâmetros: B.

$$\begin{aligned}
B_{3i}(t+1) = & \gamma_{3,0}^i + \frac{\gamma_{3,1}^i}{2\Xi} \left\{ \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{3i}}{8} (\xi_i \sqrt{2C_{ii}/N})^2 d2 \right. \\
& + \frac{B_{3i}}{48} (\xi_i \sqrt{2C_{ii}/N})^4 d4 + \frac{B_{4i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\
& + \frac{B_{4i}}{384} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \sum_{j \neq i} \left[\frac{B_{3j}}{48} (\xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N}) d4 \right. \\
& \left. + \frac{B_{4j}}{384} (\xi_j \sqrt{2C_{jj}/N})^4 (\sigma \xi_i \sqrt{2C_{ii}/N}) d5 \right] \left\} + \frac{\gamma_{3,2}^i}{8\Xi} \left\{ (\xi_i \sqrt{2C_{ii}/N})^2 d2 \right. \\
& + \frac{B_{3i}}{2} \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{3i}}{4} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\
& + \frac{B_{3i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \frac{B_{4i}}{8} (\xi_i \sqrt{2C_{ii}/N})^2 d2 + \frac{B_{4i}}{24} (\xi_i \sqrt{2C_{ii}/N})^4 d4 \\
& + \frac{B_{4i}}{384} (\xi_i \sqrt{2C_{ii}/N})^6 d6 + \sum_{j \neq i} \left[\frac{B_{3j}}{48} (\sigma \xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^2 d5 \right. \\
& \left. + \frac{B_{4j}}{384} (\xi_j \sqrt{2C_{jj}/N})^4 (\xi_i \sqrt{2C_{ii}/N})^2 d6 \right] \left\} + \frac{\gamma_{3,3}^i}{16\Xi} \left\{ \frac{1}{3} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \right. \\
& + \frac{B_{3i}}{3} d0 + \frac{B_{3i}}{2} (\xi_i \sqrt{2C_{ii}/N})^2 d2 + \frac{B_{3i}}{8} (\xi_i \sqrt{2C_{ii}/N})^4 d4 \\
& + \frac{B_{3i}}{144} (\xi_i \sqrt{2C_{ii}/N})^6 d6 + \frac{B_{4i}}{6} \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{4i}}{8} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\
& + \frac{B_{4i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \frac{B_{4i}}{1152} (\sigma \xi_i \sqrt{2C_{ii}/N})^7 d7 \\
& \left. + \sum_{j \neq i} \left[\frac{B_{3j}}{144} (\xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^3 d6 \right. \right.
\end{aligned}$$

$$+ \frac{B_{4i}}{1152} (\xi_j \sqrt{2C_{jj}/N})^4 (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d7 \} \quad (C.3)$$

$$\begin{aligned} B_{4i}(t+1) = & \gamma_{4,0}^i + \frac{\gamma_{4,1}^i}{2\Xi} \left\{ \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{3i}}{8} (\xi_i \sqrt{2C_{ii}/N})^2 d2 \right. \\ & + \frac{B_{3i}}{48} (\xi_i \sqrt{2C_{ii}/N})^4 d4 + \frac{B_{4i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\ & + \frac{B_{4i}}{384} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \sum_{j \neq i} \left[\frac{B_{3j}}{48} (\xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N}) d4 \right. \\ & + \frac{B_{4j}}{384} (\xi_j \sqrt{2C_{jj}/N})^4 (\sigma \xi_i \sqrt{2C_{ii}/N}) d5 \left. \right] + \frac{\gamma_{4,2}^i}{8\Xi} \left\{ (\xi_i \sqrt{2C_{ii}/N})^2 d2 \right. \\ & + \frac{B_{3i}}{2} \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{3i}}{4} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\ & + \frac{B_{3i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \frac{B_{4i}}{8} (\xi_i \sqrt{2C_{ii}/N})^2 d2 + \frac{B_{4i}}{24} (\xi_i \sqrt{2C_{ii}/N})^4 d4 \\ & + \frac{B_{4i}}{384} (\xi_i \sqrt{2C_{ii}/N})^6 d6 + \sum_{j \neq i} \left[\frac{B_{3j}}{48} (\sigma \xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^2 d5 \right. \\ & + \frac{B_{4j}}{384} (\xi_j \sqrt{2C_{jj}/N})^4 (\xi_i \sqrt{2C_{ii}/N})^2 d6 \left. \right] + \frac{\gamma_{4,3}^i}{16\Xi} \left\{ \frac{1}{3} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \right. \\ & + \frac{B_{3i}}{3} d0 + \frac{B_{3i}}{2} (\xi_i \sqrt{2C_{ii}/N})^2 d2 + \frac{B_{3i}}{8} (\xi_i \sqrt{2C_{ii}/N})^4 d4 \\ & + \frac{B_{3i}}{144} (\xi_i \sqrt{2C_{ii}/N})^6 d6 + \frac{B_{4i}}{6} \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{4i}}{8} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\ & + \frac{B_{4i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \frac{B_{4i}}{1152} (\sigma \xi_i \sqrt{2C_{ii}/N})^7 d7 \\ & + \sum_{j \neq i} \left[\frac{B_{3j}}{144} (\xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^3 d6 \right. \\ & + \frac{B_{4j}}{1152} (\xi_j \sqrt{2C_{jj}/N})^4 (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d7 \left. \right] + \frac{\gamma_{4,4}^i}{16\Xi} \left\{ (\xi_i \sqrt{2C_{ii}/N})^4 \frac{d4}{24} \right. \\ & + \frac{B_{3i}}{6} \sigma \xi_i \sqrt{2C_{ii}/N} d1 + \frac{B_{3i}}{8} (\sigma \xi_i \sqrt{2C_{ii}/N})^3 d3 \\ & + \frac{B_{3i}}{48} (\sigma \xi_i \sqrt{2C_{ii}/N})^5 d5 + \frac{B_{3i}}{1152} (\sigma \xi_i \sqrt{2C_{ii}/N})^7 d7 + \frac{B_{4i}}{24} d0 \\ & + \frac{B_{4i}}{12} (\xi_i \sqrt{2C_{ii}/N})^2 d2 + \frac{B_{4i}}{32} (\xi_i \sqrt{2C_{ii}/N})^4 d4 + \frac{B_{4i}}{288} (\xi_i \sqrt{2C_{ii}/N})^6 d6 \\ & + \frac{B_{4i}}{9216} (\xi_i \sqrt{2C_{ii}/N})^8 d8 + \sum_{j \neq i} \left[\frac{B_{3j}}{1152} (\sigma \xi_j \sqrt{2C_{jj}/N})^3 (\xi_i \sqrt{2C_{ii}/N})^4 d7 \right. \\ & + \frac{B_{4j}}{9216} (\xi_j \sqrt{2C_{jj}/N})^4 (\xi_i \sqrt{2C_{ii}/N})^4 d8 \left. \right] \left. \right\} \quad (C.4) \end{aligned}$$

BIBLIOGRAFIA

- [1] Amari, S., (1967) A theory of adaptive pattern classifiers, *IEEE Trans. Elect. Comput EC*-**16**, 299-307.
- [2] Oppor, M., (1998) A bayesian approach to on-line learning, *On-line learning in neural networks*, Ed. Saad, D., Cambridge Univ. Press.
- [3] Rosenblatt, F., (1962) Principles of Neurodynamics, *Spartan, New York*.
- [4] McCulloch, W. S., Pitts, W., (1943), A logical calculus of ideas immanent in nervous activity, *Bull. Math. Biophys.*,**5**, 115.
- [5] Caianiello, E. R., (1961) Outline of a theory of thought processes and thinking machines, *J. Theor. Biol.*, **1**, 204.
- [6] Hebb, D. O., (1949) The organization of behavior: A neurophysiological theory, *J. Wiley, New York*.
- [7] Watkin, T.L.H., Rau A. and Biehl, M. (1993). The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65, 499-556.
- [8] Oppor, M., (1996) Online versus offline learning from examples: general results, *Phys. Rev. Lett.*, **77**, 4671-4674.
- [9] Caticha, N., Oliveira, E. A., (2000) Gradient descent learning in and out of equilibrium. *Submitted to Physical Review E*.
- [10] Kinouchi, O., Caticha, N., (1993) Lower bounds on generalization errors for drifting rules, *J. Phys. A: Math. Gen.*, **26**, 6161-6171.
- [11] Kinouchi, O., Caticha, N., (1996) Learning algorithm which gives the bayes generalization limit for perceptrons, *Phys. Rev. E*, **54**, R54.

-
- [12] Hertz, J., Krogh, A. and Palmer, R. G., (1991) Introduction to the theory of neural computation, *Addison-Wesley*.
- [13] Barkai, N., Seung, H., Sompolinsky, H., (1995) Local and global convergence of on-line learning, *Phys. Rev. Lett.*, **75**, 1415-1418.
- [14] Amari, S. and Murata, N. (1993) Statistical theory of learning curves under entropic loss criterion. *Neural Computation*, **5**, 140-143
- [15] Yang, H. H. and Amari, S., (1997) Adaptive on-line learning algorithms for blind separation – maximum entropy and minimum mutual information. *To appear in Neural Computation - 1997*.
- [16] Hagedorn, A. G., (1998) Raising and lowering operators for semiclassical wave packets, *Ann. of Phys.*, **269**, 77-104.
- [17] Levin, E., Tishby, N., Solla, S., (1990), A statistical approach to learning and generalization in layered neural networks, *Proc. IEEE*, **78**, 1568-1574.
- [18] Reif, F. (1965) Fundamentals of statistical and thermal physics, *McGraw-Hill*.
- [19] Kinouchi, O., Caticha, N., (1992) Optimal generalization in perceptrons, *J. Phys. A: Math. Gen*, **25**, 6243-6250.
- [20] Van den Broeck, C., Riemann, P., (1996) Unsupervised learning by examples: on-line versus off-line, *Phys. Rev. Lett.*, **76**, 2188-219.
- [21] Vicente, R., Kinouchi, O., Caticha, N., (1998) Statistical mechanics of online learning of drifting concepts: a variational approach, *Machine Learning*, **32**, 179-201.
- [22] Solla, S., Winther, O., (1998) Optimal perceptron learning: an on-line bayesian approach, *On-line learning in neural networks*, Ed. Saad, D., Cambridge Univ. Press.
- [23] Oppor, M. and Haussler, D., (1991), Generalization performance of Bayes optimal classification algorithm for learning a perceptron, *Phys. Rev. Lett.*, **66**, 2677.

-
- [24] Copelli, M., Eichhorn, R., Kinouchi, O., Biehl, M., Simonetti, R., Riegler, P., Caticha, N., (1997) Noise robustness in multilayer neural networks, *Europhys. Lett*, **37**, 427-432.

*“O temor do Senhor é o
princípio do saber, mas os
loucos desprezam a sabedo-
ria e o ensino”*

Pv 1:7