Redes Neurais Artificiais e Máquinas de Aprendizado (Parte 1)

Índice

Nota: Este material contém contribuições dos Profs. Márcio Luiz de Andrade Netto e Leandro Nunes de Castro Silva, os quais já participaram do oferecimento da disciplina IA353 – Redes Neurais, junto ao Programa de Pós-Graduação da FEEC/Unicamp.

1. Leituras complementares e Referências Bibliográficas

Páginas WEB:

ftp://ftp.sas.com/pub/neural/FAQ.html (comp.ai.neural-nets FAQ)

https://papers.nips.cc/ (Anais on-line de "Neural Information Processing Systems (NIPS)")

http://ieeexplore.ieee.org/Xplore (Todas as publicação on-line do IEEE, inclusive de conferências em redes neurais artificiais, como a International Joint Conference on Neural Networks (IJCNN))

https://arxiv.org/ (ArXiv: ampla coleção de *e-prints* de áreas pertinentes)

Periódicos:

IEEE Transactions on Neural Networks and Learning Systems

Neural Computation (MIT Press)

International Journal of Neural Systems (World Scientific Publishing)

IEEE Transaction on Systems, Man, and Cybernetics (Part B)

IEEE Transactions on Pattern Analysis and Machine Intelligence

Learning & Nonlinear Models (SBIC - Brasil)

Journal of Machine Learning Research

Neural Networks (Pergamon Press)

Neurocomputing (Elsevier)

Biological Cybernetics (Springer)

Neural Processing Letters (Springer)

Information Sciences (Elsevier)

Cognitive Science (CSS)

Machine Learning (Springer)

Livros:

1. Arbib, M.A. (ed.) (2002) "The Handbook of Brain Theory and Neural Networks", The MIT Press, 2nd. edition, ISBN: 0262011972.
2. Bertsekas, D.P. & Tsitsiklis, J.N. (1996) "Neuro-Dynamic Programming", Athena Scientific, ISBN: 1886529108.
3. Bishop, C.M. (1996) "Neural Networks for Pattern Recognition", Oxford University Press, ISBN: 0198538642.
4. Bishop, C.M. (2007) "Pattern Recognition and Machine Learning", Springer, ISBN: 0387310738.
5. Braga, A.P., de Carvalho, A.P.L.F. & Ludermir, T.B. (2007) "Redes Neurais Artificiais – Teoria e Aplicações", Editora LTC, 2a. edição, ISBN: 9788521615644.
6. Chauvin, Y. & Rumelhart, D.E. (1995) "Backpropagation: Theory, Architectures, and Applications", Lawrence Erlbaum Associates, ISBN: 080581258X.
7. Cherkassky, V. & Mulier, F. (2007) "Learning from Data: Concepts, Theory, and Methods", 2nd edition, Wiley-IEEE Press, ISBN: 0471681822.
8. Chollet, F. (2017) "Deep Learning with Python", Manning Publications Co., ISBN: 9781617294433.
9. Cristianini N. & Shawe-Taylor, J. (2000) "An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods", Cambridge University Press, ISBN: 0521780195.
10. da Silva, I.N., Spatti, D.H. & Flauzino, R.A. (2010) "Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas", Artliber Editora Ltda., ISBN: 9788588098534.
11. Dayan, P. & Abbot, L.F. (2001) "Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems", The MIT Press, ISBN: 0262041995.

12. Duda, R.O., Hart, P.E. & Stork, D.G. (2000) "Pattern Classification", 2nd edition, Wiley-Interscience, ISBN: 0471056693.
13. Edelman, G.M. (1988) "Neural Darwinism: The Theory of Neuronal Group Selection", Basic Books, ISBN: 0465049346.
14. Fausett, L. (2004) "Fundamentals of Neural Networks: Architectures, Algorithms, and Applications", Dorling Kindersley India, ISBN: 8131700534.
15. Fiesler, E. & Beale, R. (1996) "Handbook of Neural Computation", Institute of Physics Publishing, ISBN: 0750303123.
16. Gardner, H. (2011) "Frames of Mind: The Theory of Multiple Intelligences", 3rd edition, BasicBooks, ISBN: 0465024335.
17. Goodfellow, I.; Bengio, Y. & Courville, A. (2016) "Deep Learning", The MIT Press, ISBN-13: 978-0262035613.
18. Gulli, A. & Pal, S. (2017) "Deep Learning with Keras: Implementing deep learning models and neural networks with the power of Python", Packt Pub., ISBN: 9781787128422.
19. Hassoun, M. (2003) "Fundamentals of Artificial Neural Networks", A Bradford Book, ISBN: 0262514672.
20. Hastie, T., Tibshirani, R. & Friedman, J.H. (2001) "The Elements of Statistical Learning", Springer, ISBN: 0387952845.
21. **Haykin, S. (2008) "Neural Networks and Learning Machines", 3rd edition, Prentice Hall, ISBN: 0131471392.**
22. Hecht-Nielsen, R. (1990) "Neurocomputing", Addison-Wesley Publishing Co., ISBN: 0201093553.

23. Hertz, J., Krogh, A. & Palmer, R. (1991) “Introduction to the Theory of Neural Computation”, Addison-Wesley, ISBN: 0201515601.
24. Kearns, M.J., Vazirani, U. (1994) “An Introduction to Computational Learning Theory”, The MIT Press, ISBN: 0262111934.
25. Kohonen, T. (1989) “Self-Organization and Associative Memory”, 3rd edition, Springer-Verlag, ISBN: 0387513876. (1st Edition: 1984; 2nd edition: 1988)
26. Kohonen, T. (2000) “Self-Organizing Maps”, 3rd Edition, Springer, ISBN: 3540679219.
27. Luenberger, D.G. (1984) “Linear and Nonlinear Programming”, 2nd edition, Addison-Wesley, ISBN: 0201157942.
28. Mackay, D.J.C. (2003) “Information Theory, Inference and Learning Algorithms”, Cambridge University Press, ISBN: 0521642981.
29. Mardia, K.V., Kent, J.T., Bibby, J.M. (1980) “Multivariate Analysis”. Academic Press, ISBN: 0124712525.
30. Marsland, S. (2009) “Machine Learning: An Algorithmic Perspective”, Chapman and Hall/CRC, ISBN: 1420067184.
31. Masters, T. (1995) “Advanced Algorithms for Neural Networks: A C++ Sourcebook”, John Wiley and Sons, ISBN: 0471105880.
32. Minsky, M.L. (1988) “The Society of Mind”, Simon & Schuster, ISBN: 0671657135.
33. Minsky, M.L. & Papert, S.A. (1988) “Perceptrons: Introduction to Computational Geometry”, Expanded edition, The MIT Press, ISBN: 0262631113. (1st edition: 1969)
34. Mitchell, T.M. (1997) “Machine Learning”, McGraw-Hill, ISBN: 0071154671.
35. Raschka, S. (2015) “Python Machine Learning”, Packt Publishing Ltd., ISBN-13: 978-1783555130.

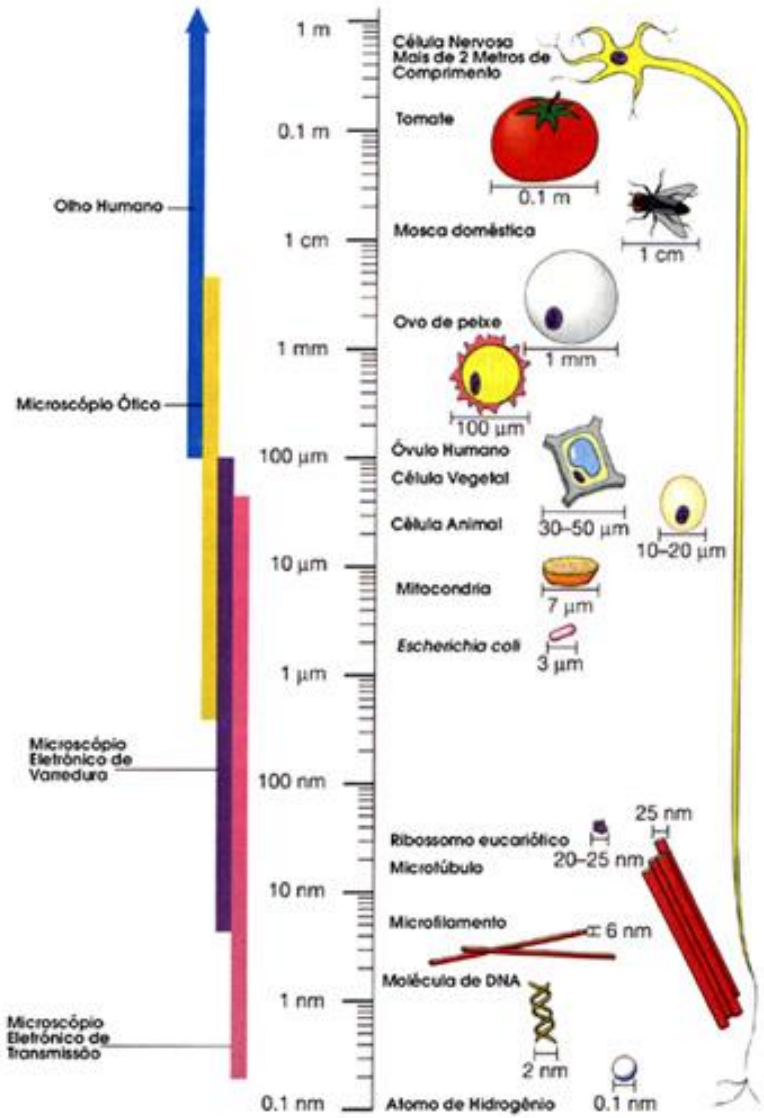
36. Ripley, B.D. (2008) “Pattern Recognition and Neural Networks”, Cambridge University Press, ISBN: 0521717701.
37. Rumelhart, D.E. & McClelland, J.L. (1986) “Parallel Distributed Processing: Explorations in the Microstructure of Cognition”, volumes 1 & 2. The MIT Press, ISBN: 026268053X.
38. Schalkoff, R.J. (1997) “Artificial Neural Networks”, The McGraw-Hill Companies, ISBN: 0071155546.
39. Schölkopf, B. & Smola, A.J. (2001) “Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond”, The MIT Press, ISBN: 0262194759.
40. Sutton, R.S. & Barto, A.G. (1998) “Reinforcement Learning: An Introduction”, The MIT Press, ISBN: 0262193981.
41. Vapnik V.N. (1998) “Statistical Learning Theory”, Wiley-Interscience, ISBN: 0471030031.
42. Vapnik V.N. (1999) “The Nature of Statistical Learning Theory”, 2nd edition, Springer, ISBN: 0387987800.
43. Weigend, A.S. & Gershenfeld, N.A. (eds.) (1993) “Time Series Prediction: Forecasting the Future and Understanding the Past”, Perseus Press, ISBN: 0201626020.
44. Wilson, R.A. & Keil, F.C. (eds.) (2001) “The MIT Encyclopedia of the Cognitive Sciences”, The MIT Press, ISBN: 0262731444.

Livros on-line:

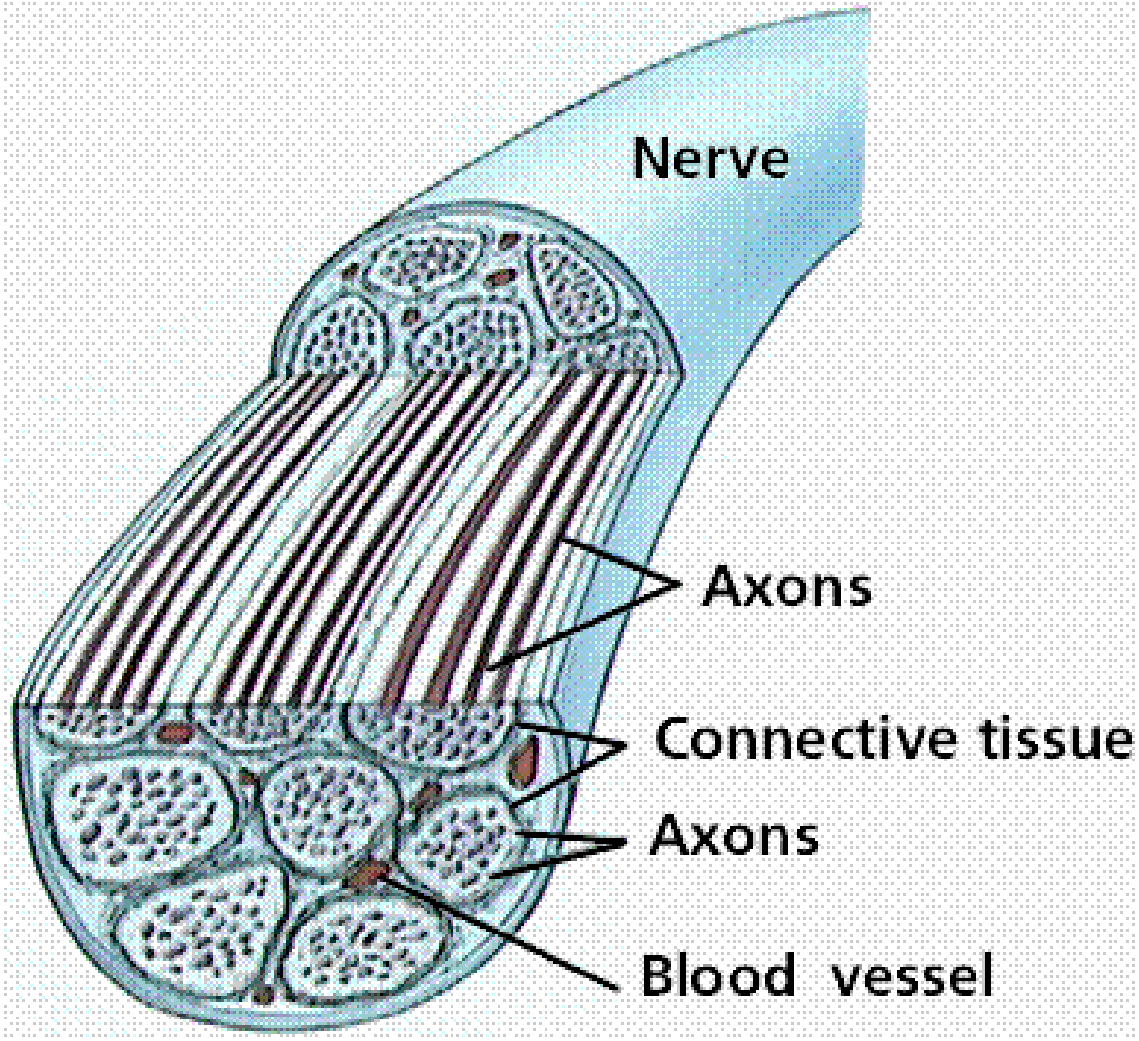
1. Nielsen, M. (2012-2019) “Neural Networks and Deep Learning”. (http://neuralnetworksanddeeplearning.com/)
2. Deep Learning Book Brasil, 2018. (http://deeplearningbook.com.br/)

2. Números e nomenclatura

- Descoberta do microscópio: ~1590
- Descoberta da célula: ~1680
- Célula como unidade constituinte dos seres vivos: ~1830
- Constituintes básicos do cérebro são os neurônios: Ramón y Cajál, ~1909
- O cérebro humano pesa ~1,5 quilos e consome ~20% da energia do corpo;
- 100 gramas de tecido cerebral requerem ~3,5ml de oxigênio por minuto;
- O cérebro humano apresenta ~10^{11} neurônios e ~10^{14} sinapses ou conexões, com uma média de ~1000 conexões por neurônio, podendo chegar a ~10000 conexões.
- Em seres humanos, 70% dos neurônios estão localizados no córtex;
- Tipos de células neurais: horizontal, estrelada, piramidal, granular, fusiforme.
- Classificação de acordo com a função: sensoriais, motoras, intrínsecas.



- O diâmetro do corpo celular de um neurônio mede de ~5µm (célula granular) a ~60µm (célula piramidal);
- Em termos fisiológicos, um neurônio é uma célula com a função específica de receber, processar e enviar informação a outras partes do organismo.
- Um nervo é formado por um feixe de axônios, com cada axônio associado a um único neurônio;
- Os nervos apresentam comprimentos variados, podendo chegar a metros.



Estrutura de um nervo

- A estrutura e as funcionalidades do cérebro são governadas por princípios básicos de alocação de recursos e otimização sujeita a restrições.

LAUGHLIN, S.B. & SEJNOWSKI, T.J. (2003) "Communication in neuronal networks", Science, vol. 301, no. 5641, pp. 1870–1874.

- O ser humano pode reagir simultaneamente a uma quantidade bem limitada de estímulos, o que pode indicar que mecanismos de alocação de recursos (e.g. glicose, oxigênio) baseados em prioridades são implementados no cérebro.

NORMAN, D.A. & BOBROW, D.G. (1975) "On data-limited and resource-limited processes", Cognitive Psychology, vol. 7, pp. 44-64.

- Alguns autores defendem que o córtex humano pode ser modelado na forma de uma rede "mundo pequeno" (BASSETT & BULLMORE, 2006; SPORNS & HONEY, 2006; SPORNS, 2010) ou então uma rede complexa (AMARAL & OTTINO, 2004).

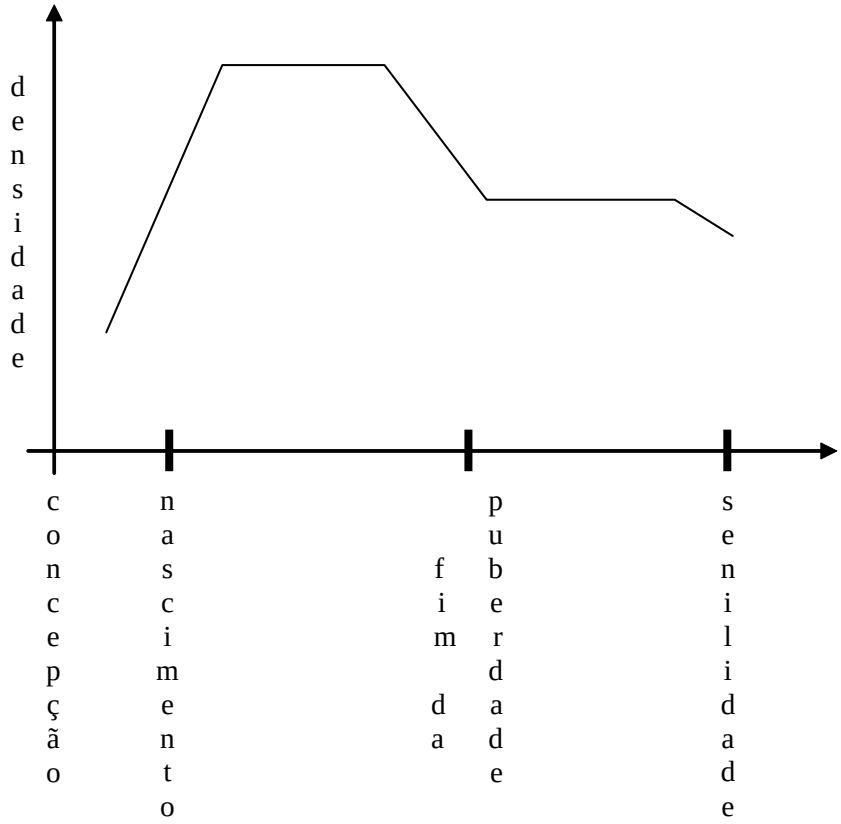
AMARAL, L. & OTTINO, J. (2004) "Complex networks", The European Physical Journal B – Condensed Matter and Complex Systems, vol. 38, no. 2, pp. 147-162.
BASSETT, D.S. & BULLMORE, E. (2006) "Small-world brain networks", Neuroscientist, vol. 12, no. 6, pp. 512-523.
SPORNS, O. & HONEY, C.J. (2006) "Small worlds inside big brains", Proceedings of the National Academy of Science, vol. 103, no. 51, pp. 19219-19220.
SPORNS, O. (2010) "Networks of the Brain", The MIT Press, ISBN: 0262014696.

- Há um expressivo aumento na densidade de conexões sinápticas da vida embrionária até a idade de 2 anos. Quando se atinge a idade de 2 anos, o ser humano apresenta a maior concentração de sinapses, a qual se mantém num nível elevado até o início da puberdade. Até o término da puberdade, há uma queda acentuada no número de sinapses.

- Esse processo de ampliação e redução de sinapses, contudo, não é homogêneo, pois nas regiões sensório-motoras este processo ocorre mais cedo, enquanto que ele é retardado em áreas associadas aos processos cognitivos.

- A redução de sinapses é dramática: o número de sinapses ao término da puberdade pode chegar a 50% do número existente com a idade de 2 anos. Há uma perda de até 100.000 sinapses por segundo na adolescência.

KOLB, B & WHISHAW, I.Q. (2008) "Fundamentals of Human Neuropsychology", Worth Publishers, 6th. edition, ISBN: 0716795868.



Evolução da densidade de sinapses ao longo da vida de um ser humano

- Acredita-se ser impossível que o código genético de um indivíduo seja capaz de conduzir todo o processo de organização topológica do cérebro. Apenas aspectos gerais dos circuitos envolvidos devem estar codificados geneticamente.

- Logo, para explicar as conformações sinápticas, recorre-se a dois mecanismos gerais de perda de sinapses: *experience expectant* e *experience dependent* (KOLB & WHISHAW, 2008).
- Podas baseadas em *experience expectant* estão vinculadas à experiência sensorial para a organização das sinapses. Geralmente, os padrões sinápticos são os mesmos para membros de uma mesma espécie. A formação de sinapses no córtex visual depende da exposição a atributos como linha de orientação, cor e movimento.
- Podas baseadas em *experience dependent* estão vinculadas a experiências pessoais únicas, tal como falar uma língua distinta. Defende-se que o padrão de conexões do lobo frontal seja formado por podas baseadas em *experience dependent*.
- De fato, a atividade do córtex pré-frontal tende a ser até 4 vezes mais intensa em crianças do que em adultos, o que permite concluir que poda de parte das conexões e fortalecimento de outras contribuem para a maturação cognitiva.

CASEY, B.J., TOTTENHAM, N., LISTON, C. & DURSTON, S. (2005) "Imaging the developing brain: what have we learned about cognitive development?", Trends in Cognitive Science, vol. 9, no. 3, pp. 104-110.

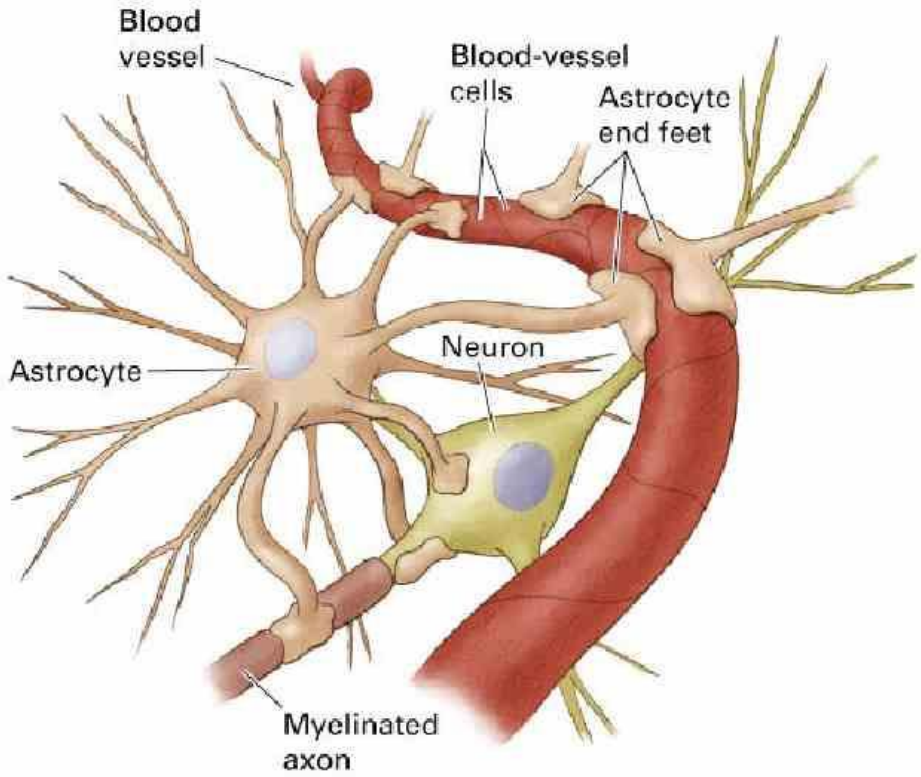
- Em síntese, é possível afirmar que o padrão de conexões no cérebro se inicia sem muita organização e com uma grande densidade de sinapses. Com a experiência de vida, um equilíbrio é atingido. Logo, como o padrão de conexões de um ser humano adulto é obtido a partir da experiência de vida, cada pessoa vai apresentar um padrão de conexões diferente, particularmente nas áreas especializadas em cognição. Por outro lado, o sistema sensório-motor em um adulto normal deve apresentar uma conformação similar à de outros adultos normais, visto que a poda nessas áreas é *experience expectant*.

FRANCO, A.R. (2009) "Resource Allocation of the human brain: a competitive equilibrium approach", Ph. D. Thesis, The University of New Mexico, Albuquerque, New Mexico, USA.

- Voltando agora a atenção para o neurônio biológico, pode-se afirmar que se trata de uma célula especializada em transmitir pulsos elétricos, sendo que as suas principais partes constituintes são:
  - ✓ Membrana celular: é a "pele" da célula;
  - ✓ Citoplasma: tudo que está envolvido pela membrana;

- ✓ Núcleo: contém os cromossomos (DNA);
- ✓ Ribossomos: geram proteínas a partir de mRNAs;
- ✓ Mitocôndria: gera energia para a célula (produz ATP);
- ✓ Soma: corpo celular, excluindo dendritos e axônio;
- ✓ Dendritos: parte do neurônio que recebe informação de outros neurônios;
- ✓ Axônio: parte do neurônio que transmite informação para outros neurônios;
- ✓ Bainha de mielina: revestimento externo lipídico do axônio, responsável por evitar a dispersão dos sinais elétricos, como uma capa isolante;
- ✓ Terminais pré-sinápticos: área do neurônio que armazena neurotransmissores, os quais são liberados por potenciais de ação.

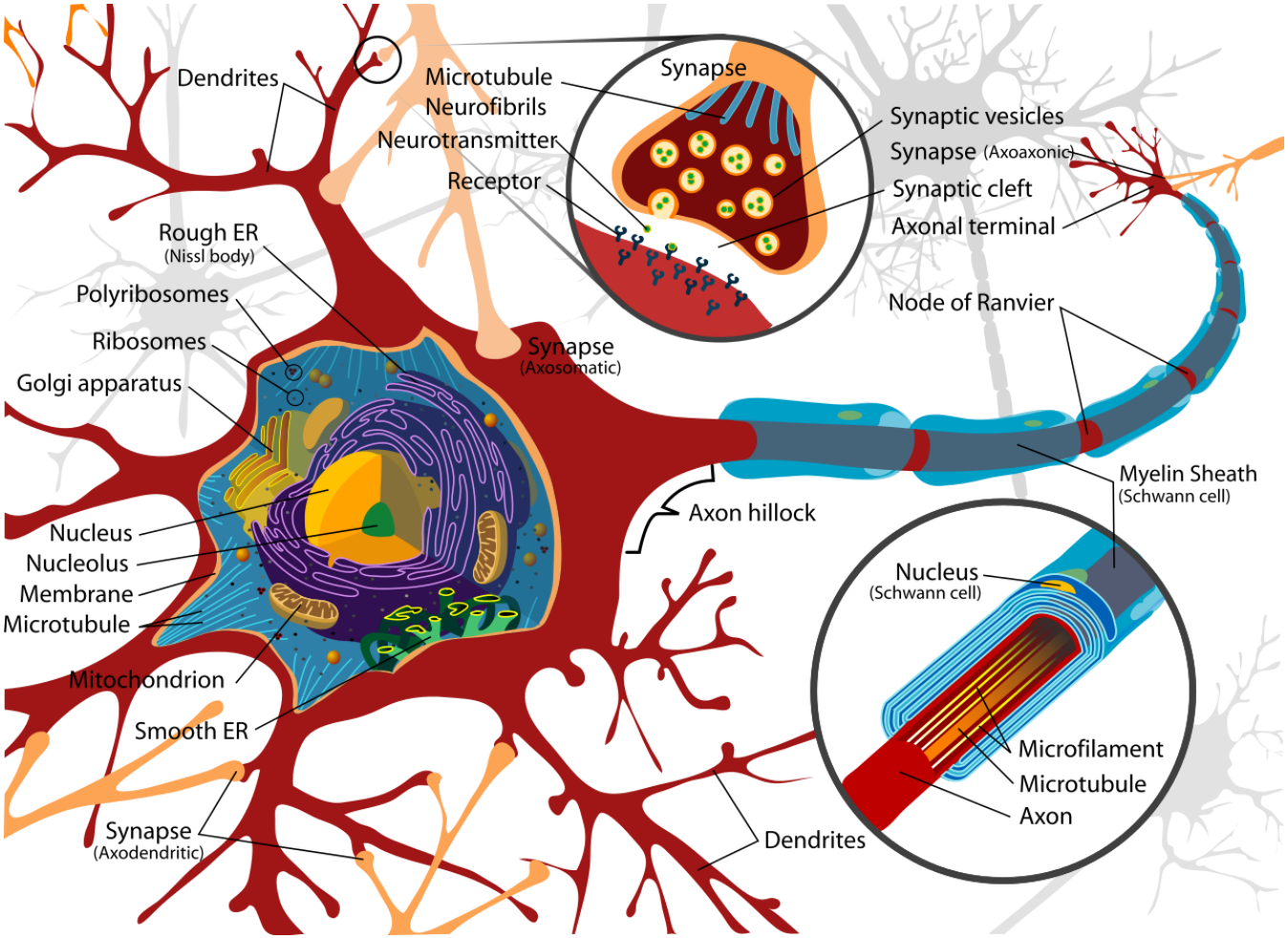
- Os neurônios sensoriais normalmente têm longos dendritos e axônios curtos. Por outro lado, os neurônios motores têm um longo axônio e dendritos curtos (transmitem informação para músculos e glândulas). Já os neurônios intrínsecos realizam a comunicação neurônio-a-neurônio e compõem o sistema nervoso central.



Brain photo

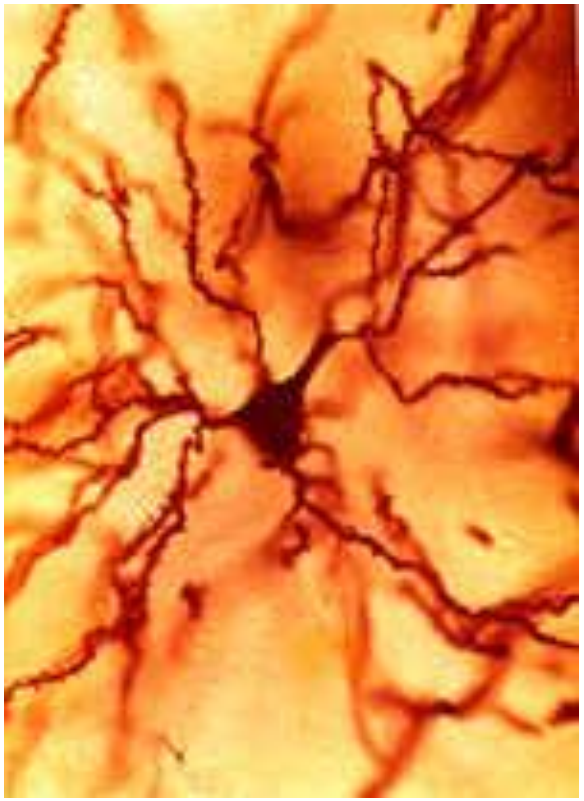
Cartoon reconstruction

- Além das células condutoras, o cérebro possui as células não-condutoras, formando a glia (neuróglia).
- Os astrócitos se caracterizam pela riqueza e dimensões de seus prolongamentos cito-plasmáticos, distribuídos em todas as direções. Funções: prover suporte estrutural, nutrientes e regulação química.
- Máxima distância de um neurônio a um vaso sanguíneo: ~50μm.



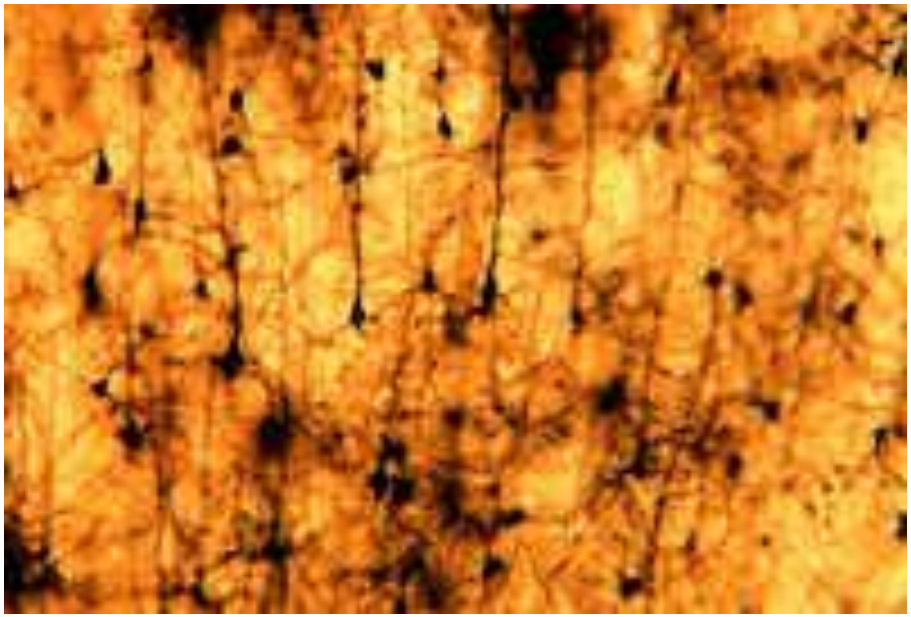
Elementos constituintes de um neurônio

Extraído de: [http://upload.wikimedia.org/wikipedia/commons/thumb/a/a9/Complete_neuron_cell_diagram_en.svg/1280px-Complete_neuron_cell_diagram_en.svg.png]



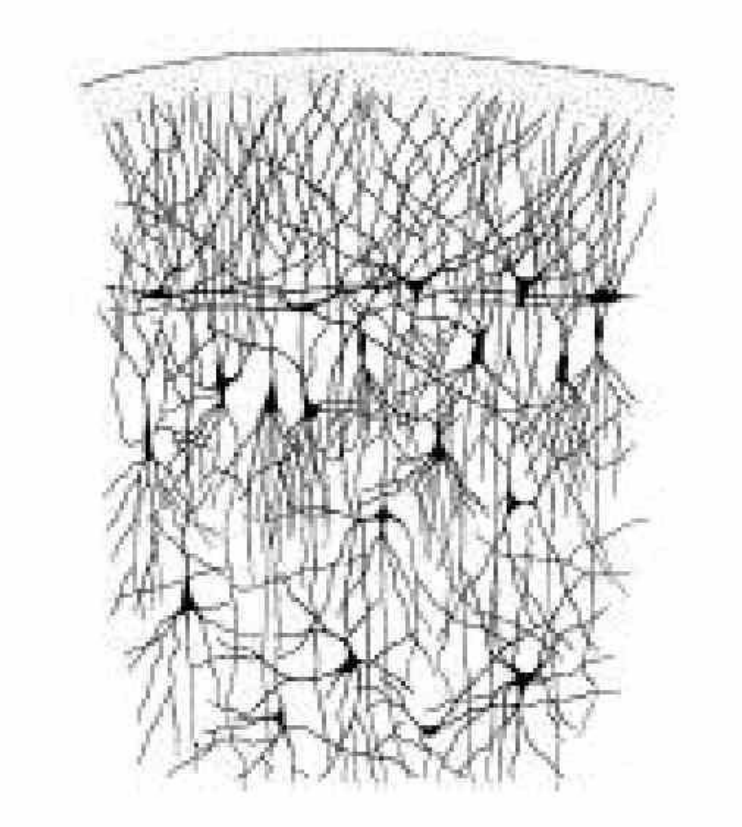
Neurônio piramidal do córtex de um hamster

⟨https://faculty.washington.edu/chudler/cellpyr.html⟩

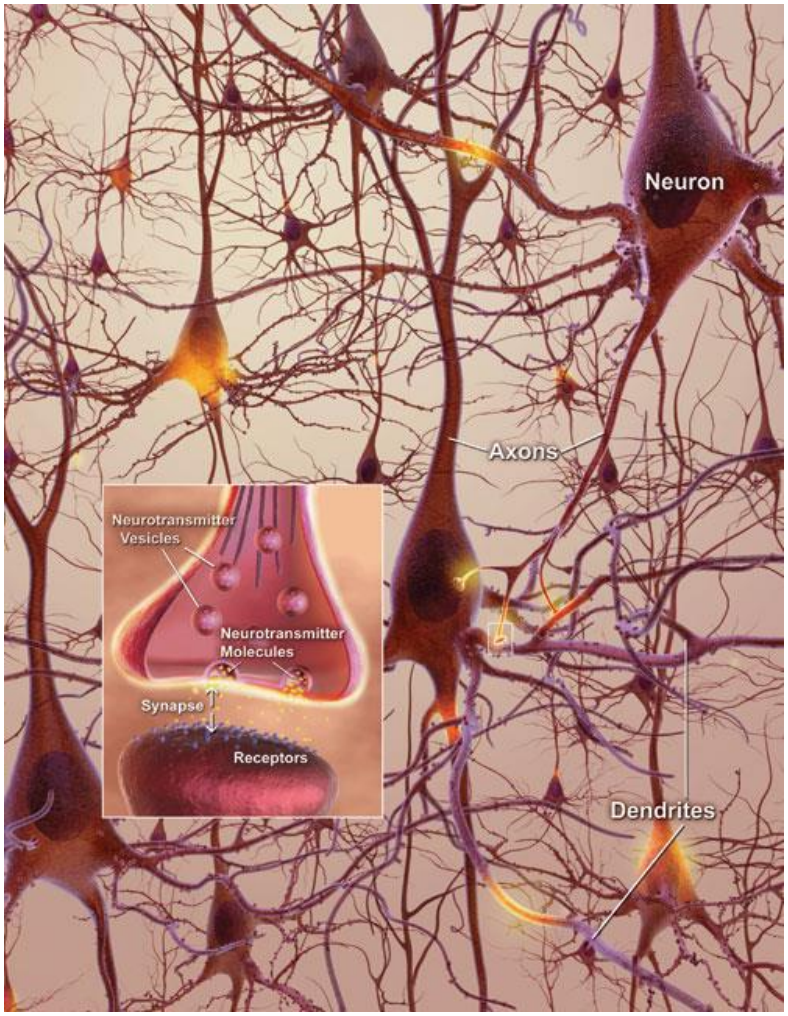


Rede de neurônios piramidais do córtex de um hamster

⟨https://faculty.washington.edu/chudler/cellpyr.html⟩



Desenho do córtex realizado por RAMÓN Y CAJÁL (1909)



Organização tridimensional de neurônios

Extraído de: [http://happypeopleteamblog.ru/wp-content/uploads/2014/01/01_neurons_lg.jpg]

Organização tridimensional de neurônios

Extraído de: [http://img-new.cgtrader.com/items/54567/pyramidal-neurons-scene_3d_model_max_555c4314-c9ae-4000-a710-67b4826a903f.jpg]

3. Alguns fatos históricos relevantes

- "Idade da Ilusão"
  - MCCULLOCH & PITTS (1943)
  - WIENER (1948): cibernética
  - MINSKY & PAPPERT (1969): a disputa entre as portas lógicas e os neurônios artificiais para determinar a unidade básica de processamento.

- "Idade das Trevas"
  - Entre 1969 e 1984, houve muito pouca pesquisa científica envolvendo redes neurais artificiais

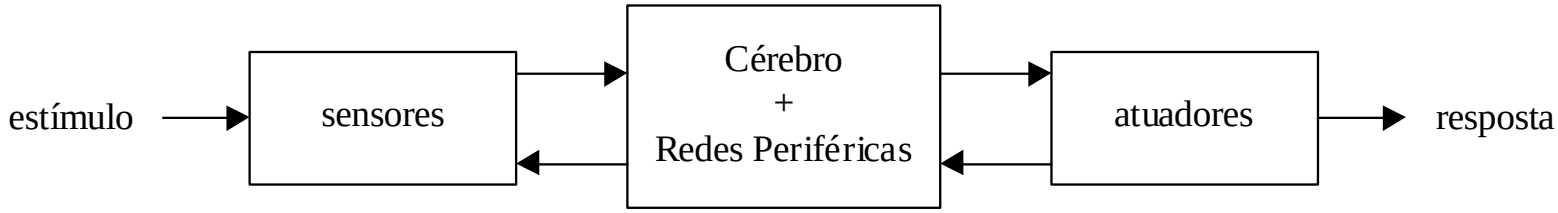
- "Renascimento"
  - HOPFIELD (1982)
  - RUMELHART & MCCLELLAND (1986)

- Desenvolvimento da capacidade de processamento e memória dos computadores digitais (simulação computacional / máquina virtual) (anos 80 e 90)
- GARDNER (1983; 2011): Múltiplas inteligências
  1. Vivacidade verbal
  2. Vivacidade matemático-lógica
  3. Aptidão espacial
  4. Gênio cinestésico
  5. Dons musicais
  6. Aptidão interpessoal (liderança e ação cooperativa)
  7. Aptidão intrapsíquica (modelo preciso de si mesmo)

- EDELMAN (1988): Neurodarwinismo
- MINSKY (1988): Sociedade da mente

4. Algumas questões operacionais

- O cérebro é capaz de perceber regularidades no meio e gerar abstrações e associações que capturam a estrutura destas regularidades, possibilitando a predição de observações futuras e o planejamento de ações visando o atendimento de múltiplos objetivos.
- Organização básica do sistema nervoso (visão de engenharia)



- O uso das mãos: propriedades inatas
- Tratamento da linguagem: propriedades não-inatas
- Nosso cérebro se desenvolve conectando células cerebrais individuais para criar vias neurais. As experiências de vida moldam a massa cefálica.

- Assimetria cerebral → Aprendizado
- Hemisfério esquerdo (paradigma sequencial): lógica, produção e compreensão da linguagem, processamento serial (considera informações pontuais a cada instante), processamento simbólico, inferência, planejamento, noção de tempo.
- Hemisfério direito (paradigma paralelo): aprendizado e memória espacial, síntese da percepção, sentimentos e emoções, pensamento associativo, processamento global da informação, raciocínio por analogia, comparação e identificação de imagens.
- Grandes avanços no estudo do cérebro:
  - ✓ Neuroimagem funcional (ressonância magnética)
  - ✓ Neuroprótese
  - ✓ Células-tronco (se diferenciando em neurônios)
- Doenças do sistema nervoso:
  - ✓ Perda de memória
  - ✓ Alzheimer
  - ✓ Esclerose
  - ✓ Parkinson

- Nos anos 60, psicólogos e antropólogos realizaram estudos com populações do mundo ocidental e outras isoladas, como nativos de Papua-Nova Guiné, e concluíram que existem seis expressões corporais básicas comuns a todos os povos, de qualquer raça, origem ou etnia: felicidade, tristeza, surpresa, nojo, raiva e medo.

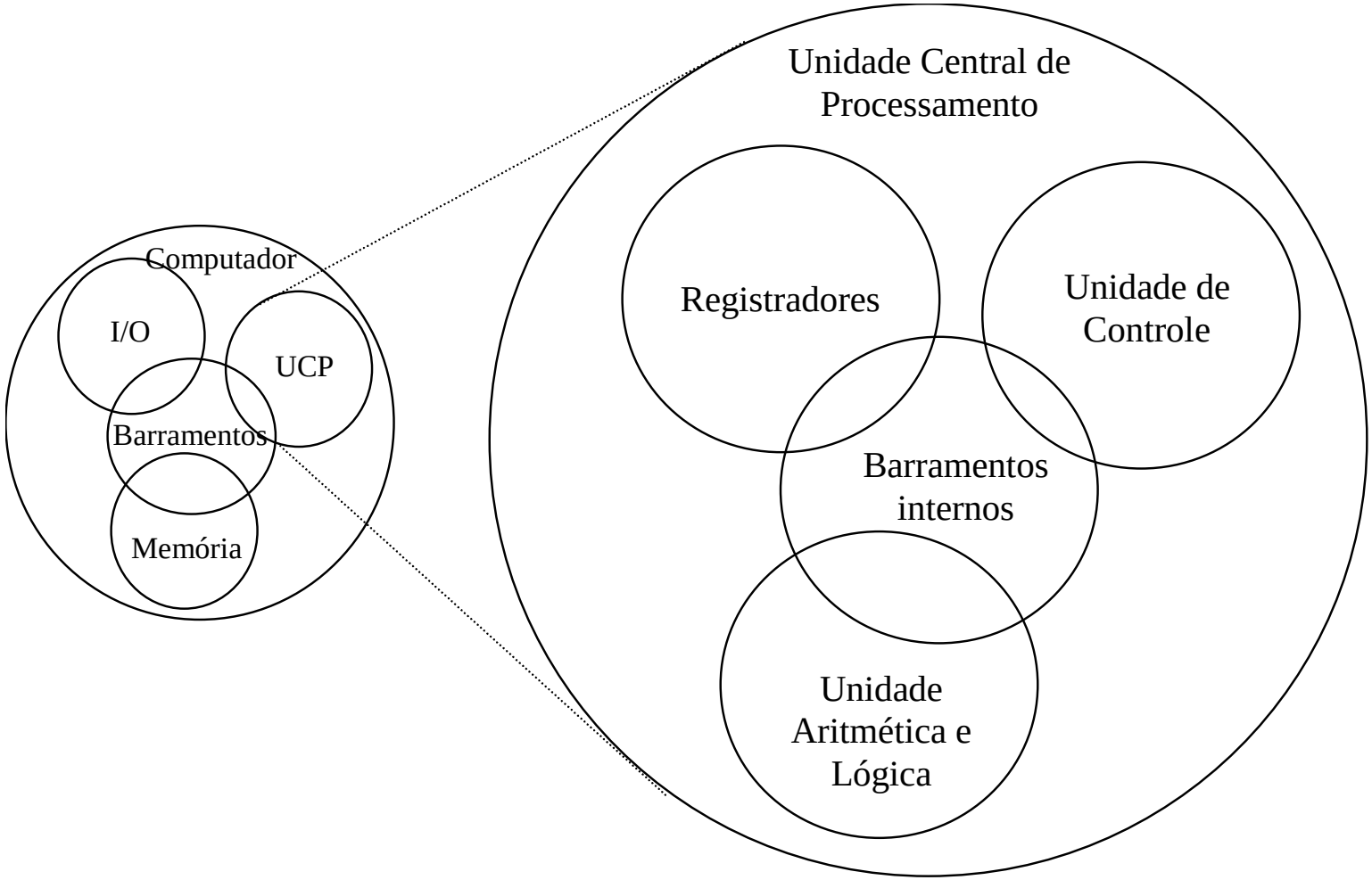
5. Cérebro e computação digital

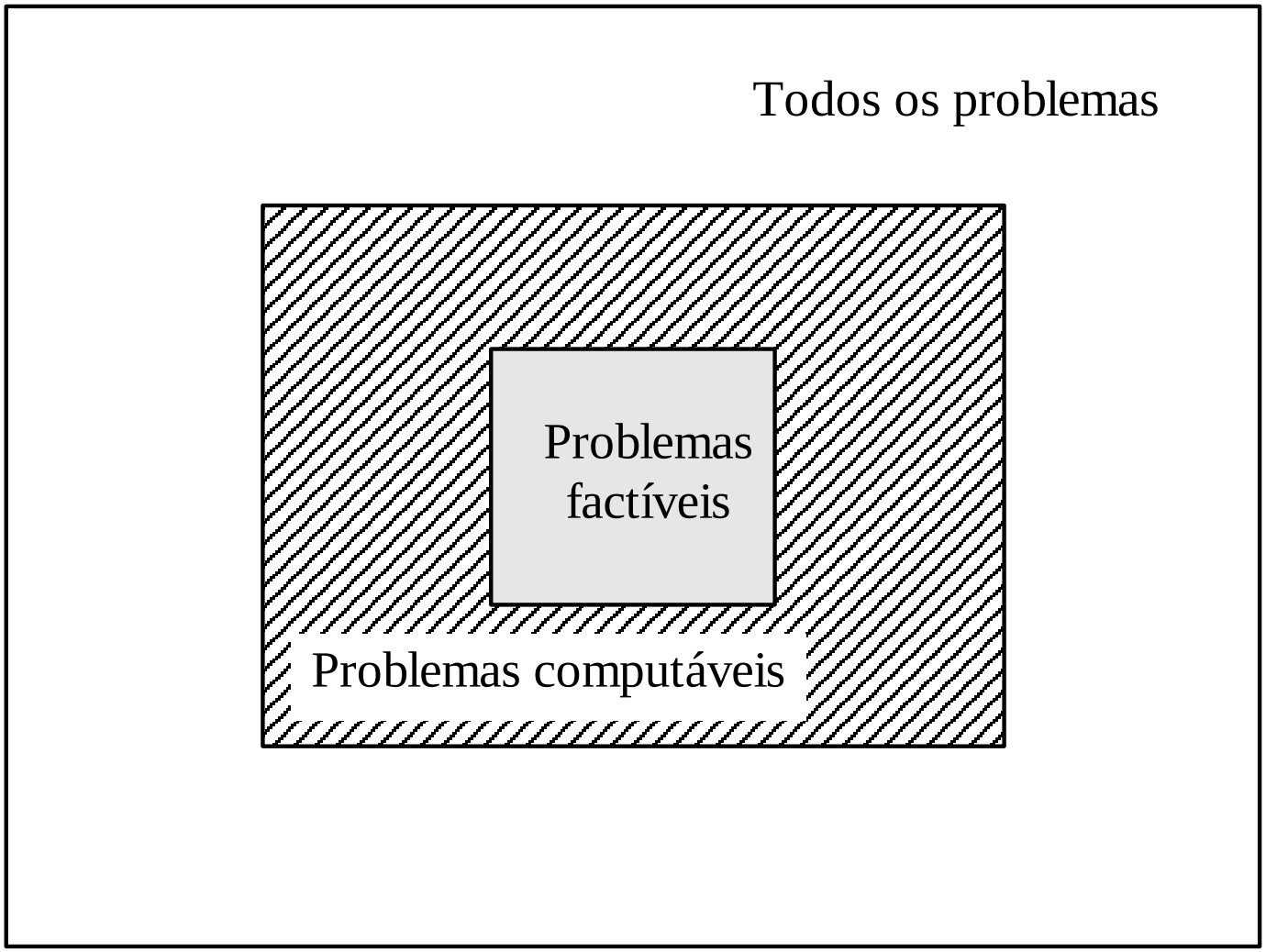
- Os computadores digitais eram chamados de cérebros eletrônicos, pois acreditava-se que eles representavam um caminho direto para a reprodução da inteligência.
- Neurônios são de 6 a 7 ordens de magnitude mais lentos do que portas lógicas de silício (10^{-3} seg. $\times$ 10^{-9} seg.)
- Eficiência energética: Cérebro $\rightarrow 10^{-16}$ Joules/operação/seg
  Computador $\rightarrow 10^{-6}$ Joules/operação/seg
- Surgimento de uma nova e promissora metodologia com a efetividade da lei de Moore: simulação computacional (máquina virtual).

- Embora o computador tenha se transformado em um dos maiores sucessos tecnológicos da história da humanidade, ele não atendeu as expectativas de reproduzir comportamento inteligente. Muitos são os exemplos de tarefas que são fáceis para o homem e difíceis para a máquina, e vice-versa.
- Já está disponível um conhecimento avançado da arquitetura fisiológica do cérebro, mas ainda é um mistério o mecanismo fundamental empregado pelo cérebro para realizar computação de alto nível.
- Da mesma forma que é inviável determinar, a partir do acompanhamento sequencial do estado lógico de seus componentes e das micro-operações realizadas, a tarefa (computação de alto nível) que está sendo executada por um computador digital, também é inviável deduzir os mecanismos de processamento de alto nível do cérebro a partir do acompanhamento da atividade cerebral, em termos de sinais produzidos pela ativação dos neurônios (SCHALKOFF, 1997). É possível, no entanto, isolar regiões funcionais e estudá-las mais detalhadamente.

6. Neurocomputação

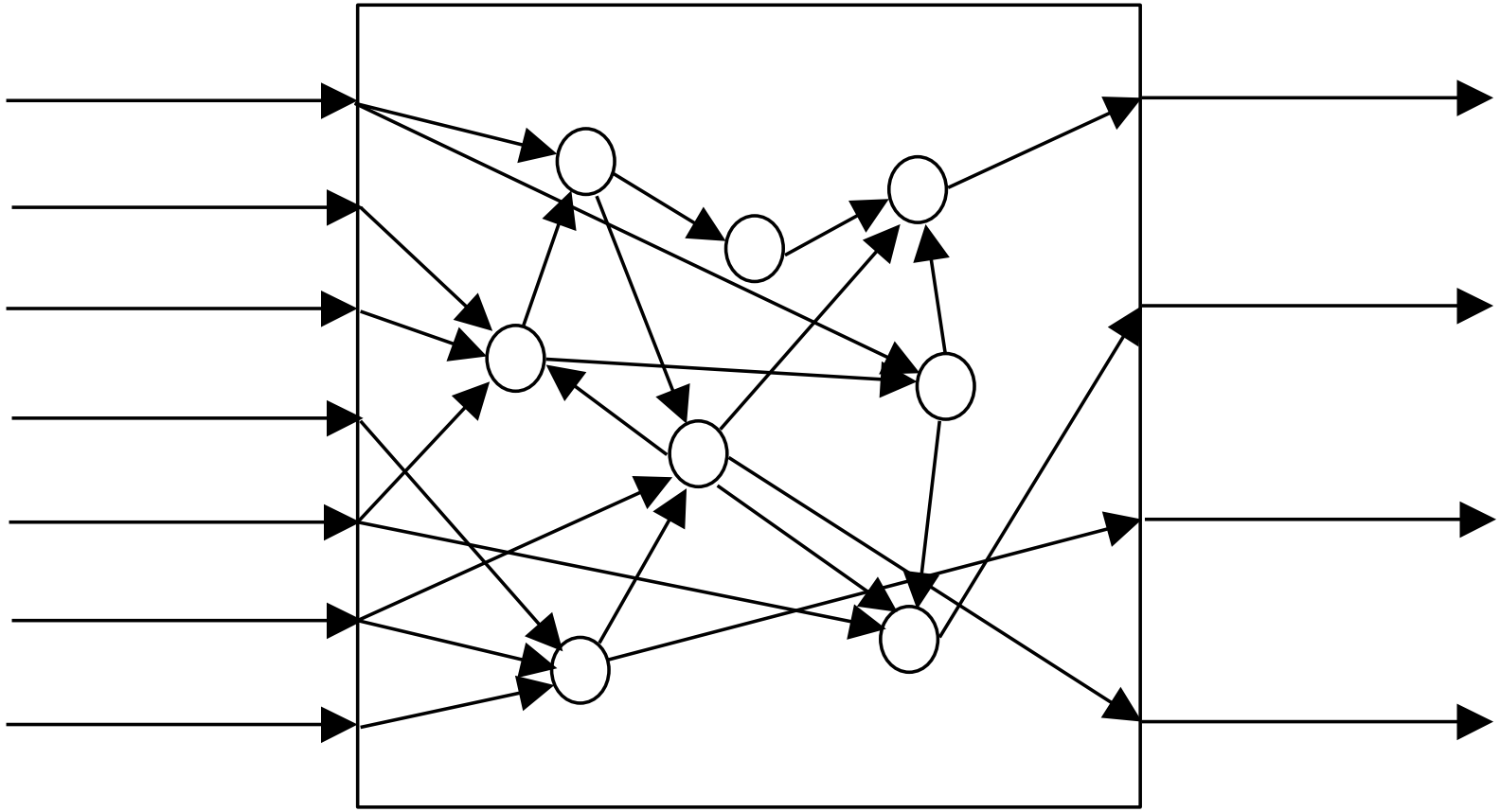
- É muito comum a associação do conceito de computação com aquele predominante no caso dos computadores com arquitetura do tipo von Neumann: algoritmos são elaborados e, em seguida, implementados na forma de programas de computador a serem executados.
- No entanto, a computação realizada pelo cérebro requer um outro tipo de definição, que contemple processamento paralelo e distribuído, além de aprendizado.
- Uma arquitetura neurocomputacional é baseada na interconexão de unidades de processamento simples e similares, denominadas neurônios artificiais e dotadas de grande poder de adaptação.
- Há uma diferença de paradigmas entre computadores com arquitetura do tipo von Neumann e redes neurais artificiais (RNAs): os primeiros realizam processamento e armazenagem de dados em dispositivos fisicamente distintos, enquanto RNAs usam o mesmo dispositivo físico para tal.


Computador
I/O
UCP
Barramentos
Memória
Unidade Central de Processamento
Registradores
Unidade de Controle
Barramentos internos
Unidade Aritmética e Lógica

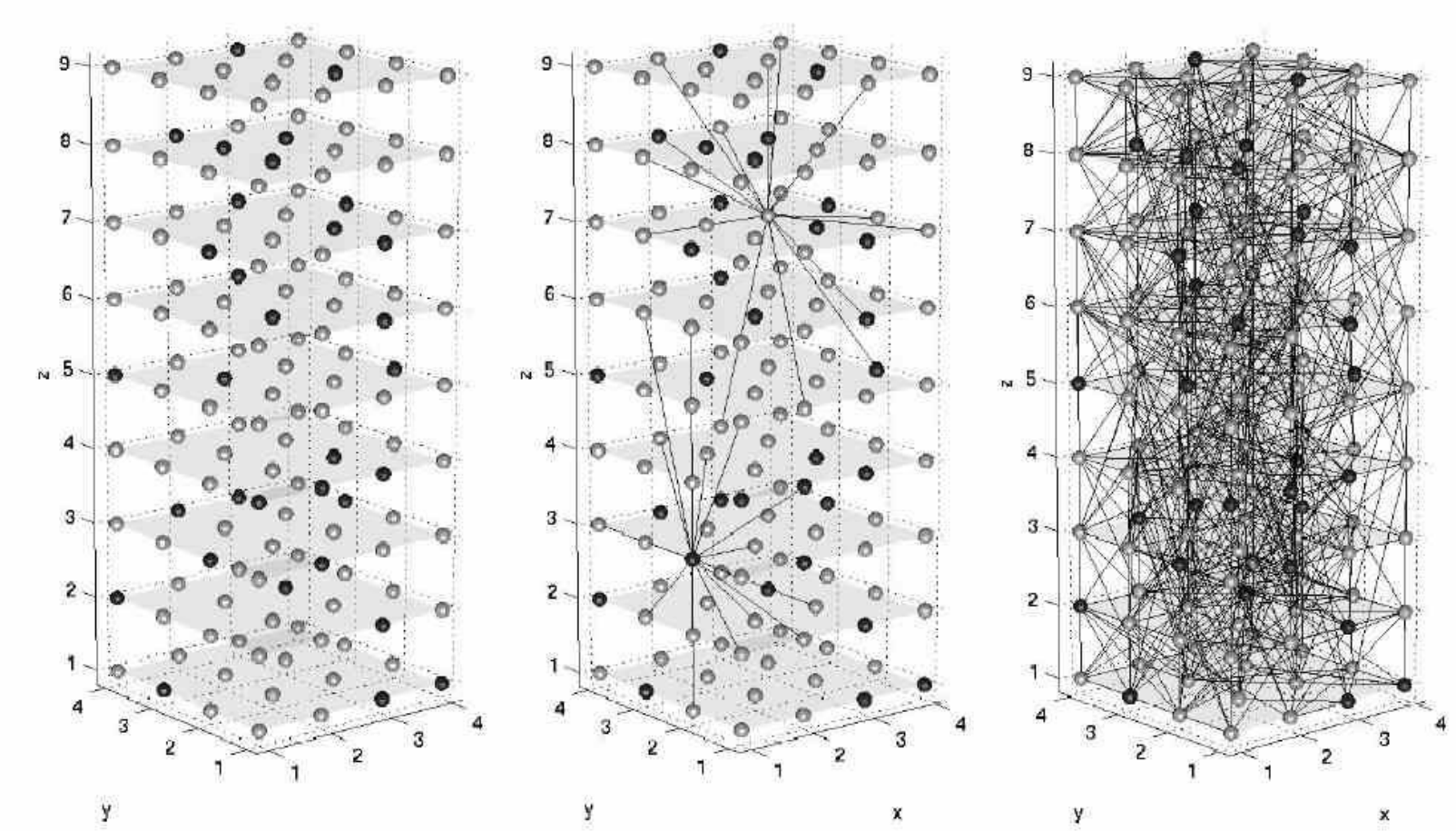


Cenário desafiador para a computação digital

Como abordar os problemas na região hachurada?



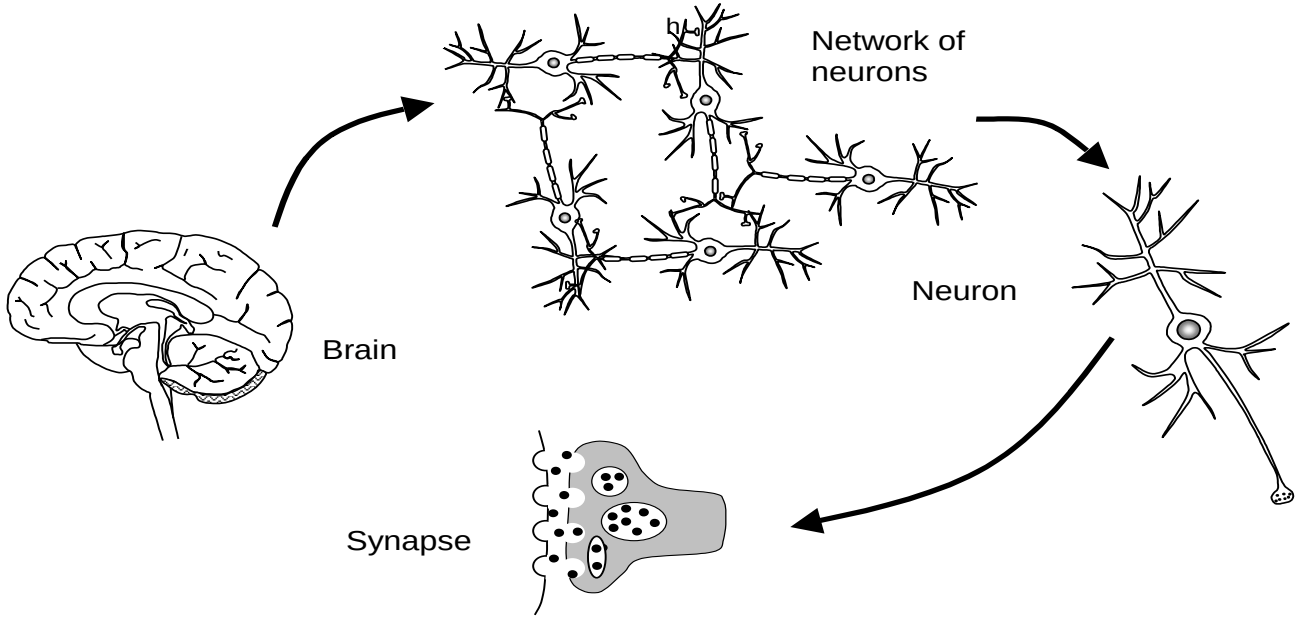
Exemplo genérico de um neurocomputador



Rede = nós + conexões (paradigma conexionista)

7. Níveis de Organização no Sistema Nervoso

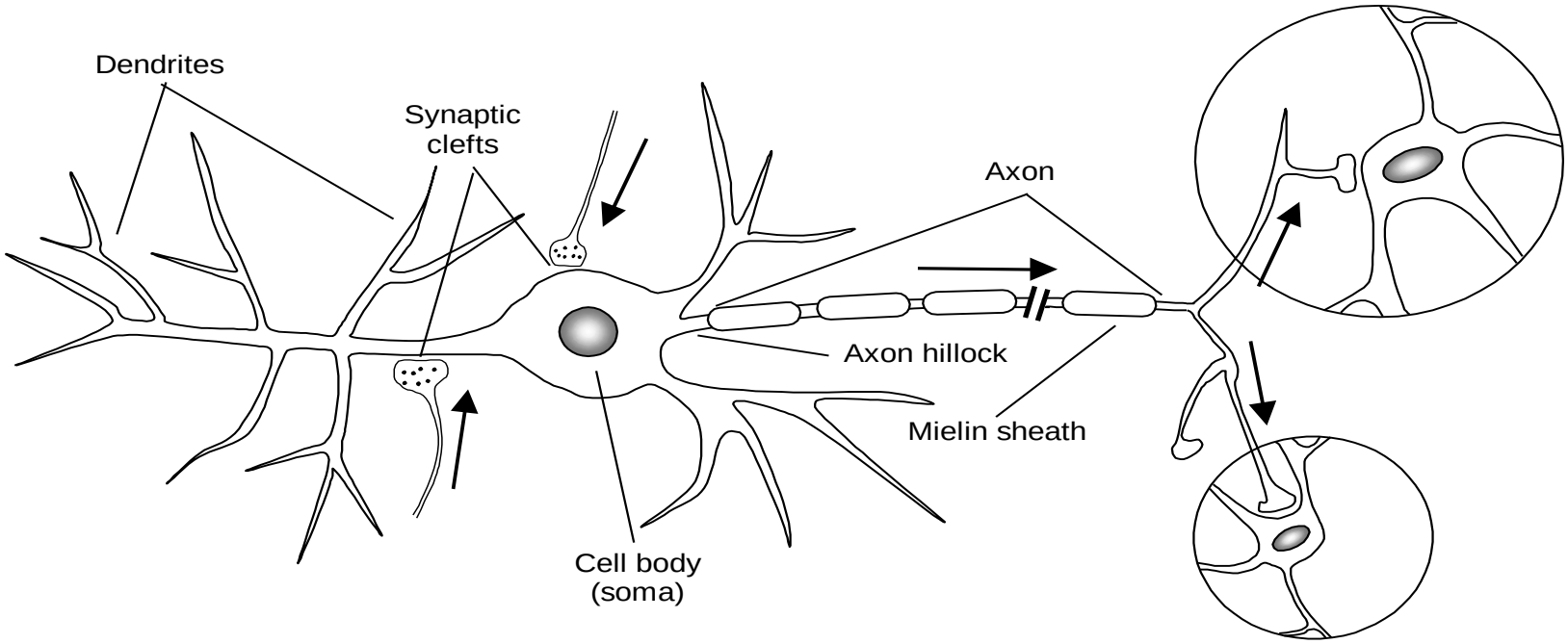
- O sistema nervoso pode ser organizado em diferentes níveis: *moléculas*, *sinapses*, *neurônios*, *camadas*, *mapas* e *sistemas*.
- Uma estrutura facilmente identificável no sistema nervoso é o neurônio, especialista em processamento de sinais.
- Dependendo das condições de operação, os neurônios são capazes de gerar um *sinal*, mais especificamente um *potencial elétrico*, que é utilizado para transmitir informação a outras células.



7.1 Neurônios e Sinapses

- Os neurônios utilizam uma variedade de mecanismos bioquímicos para o processamento e transmissão de informação, incluindo os *canais iônicos*.
- Os canais iônicos permitem um fluxo contínuo de entrada e saída de *correntes (elétricas)*, a liberação de *neurotransmissores* e a geração e propagação de *potenciais de ação*.
- O processo de transmissão de sinais entre neurônios é fundamental para a capacidade de processamento de informação do cérebro.
- Uma das descobertas mais relevantes em neurociência foi a de que a *efetividade* da transmissão de sinais pode ser modulada, permitindo que o cérebro se adapte a diferentes situações.
- A *plasticidade sináptica*, ou seja, a capacidade de sinapses sofrerem modificações, é o ingrediente-chave para o aprendizado da maioria das RNAs.
- Os neurônios podem receber e enviar sinais de/para vários outros neurônios.

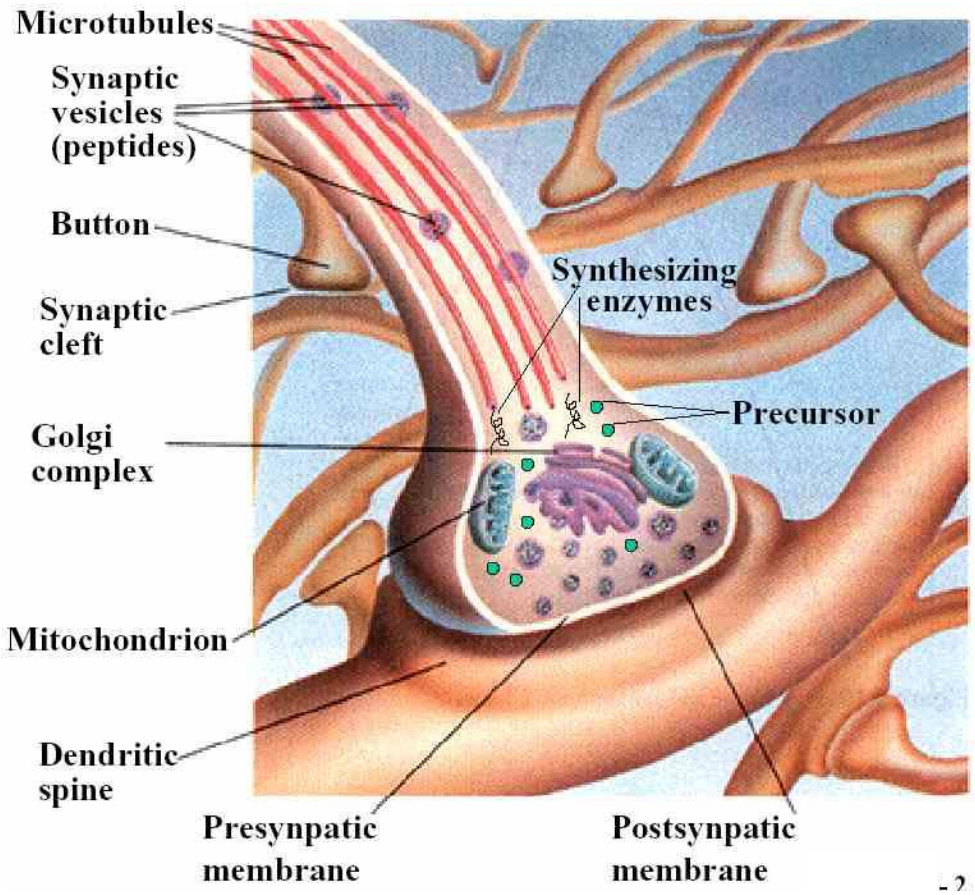
- Os neurônios que enviam sinais, chamados de neurônios *pré-sinápticos* ou "*enviadores*", fazem contato com os neurônios *receptores* ou *pós-sinápticos* em regiões especializadas, denominadas de *sinapses*.



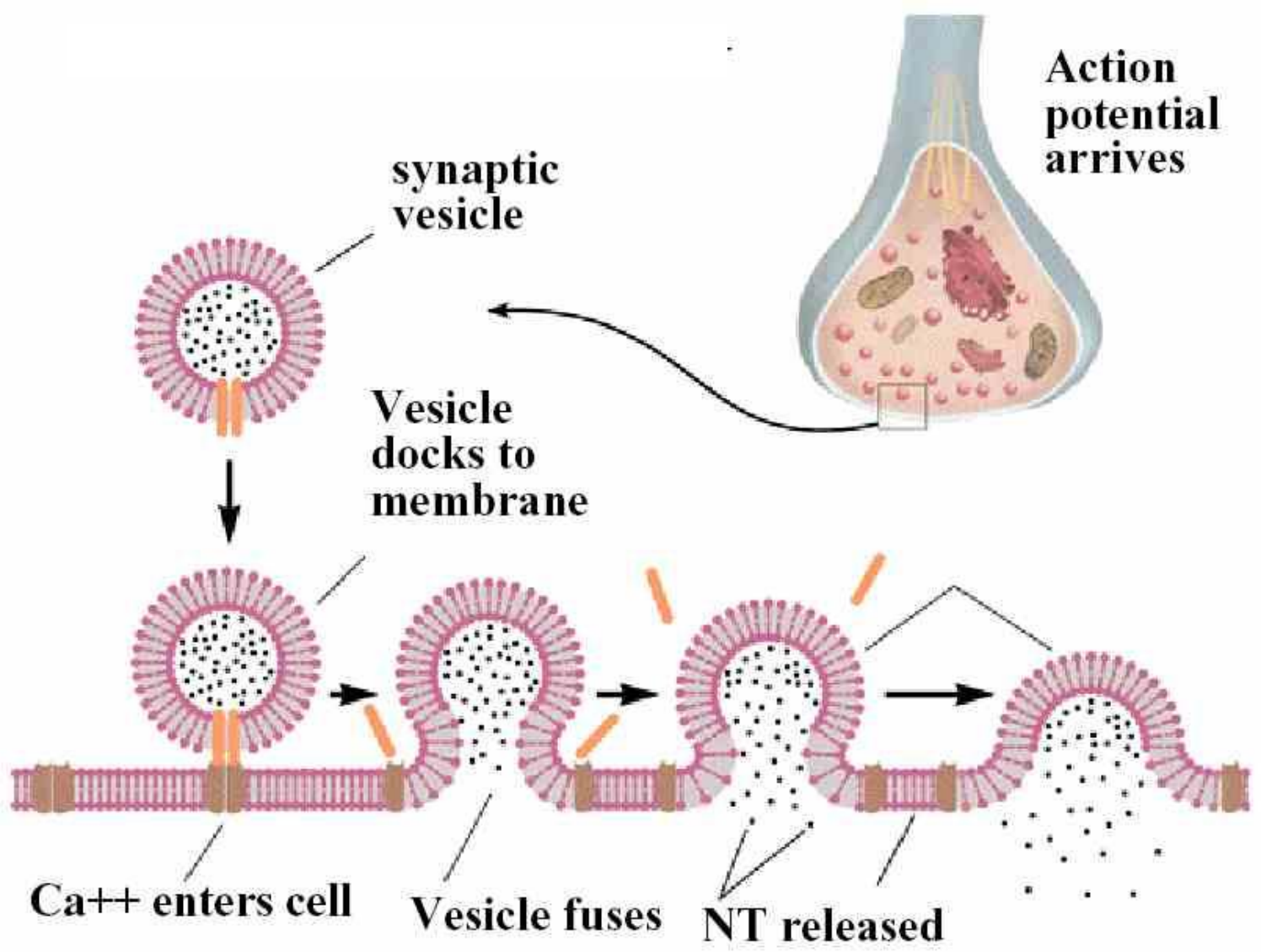
- A sinapse é, portanto, a junção entre o axônio de um neurônio pré-sináptico e o dendrito ou corpo celular de um neurônio pós-sináptico (ver figura acima).

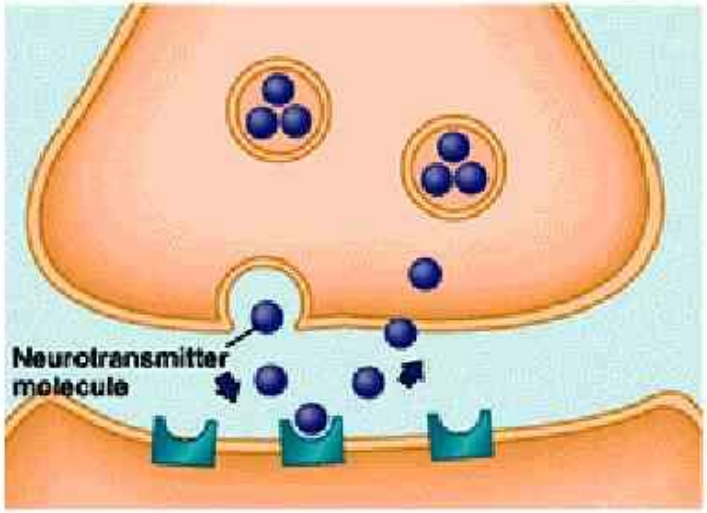
- A capacidade de processamento de informação das sinapses permite que elas alterem o estado de um neurônio pós-sináptico, eventualmente gerando um pulso elétrico, denominado *potencial de ação*, no neurônio pós-sináptico.
- Logo, um neurônio pode ser visto como um dispositivo capaz de receber estímulos (de entrada) de diversos outros neurônios e propagar sua única saída, função dos estímulos recebidos e do estado interno, a vários outros neurônios.
- Existem diversos mecanismos envolvidos na transmissão de informação (sinais) entre neurônios. Como os neurônios são células encapsuladas por membranas, pequenas aberturas nestas membranas (*canais*) permitem a transferência de informação entre eles.
- Os mecanismos básicos de processamento de informação são baseados no movimento de átomos carregados, ou *íons*:
  - ✓ Os neurônios habitam um ambiente líquido contendo uma certa concentração de íons, que podem entrar ou sair do neurônio através dos canais.

- ✓ Um neurônio é capaz de alterar o potencial elétrico de outros neurônios, denominado de *potencial de membrana*, que é dado pela diferença do potencial elétrico dentro e fora do neurônio.
- ✓ Quando um potencial de ação chega ao final do axônio, ele promove a liberação de neurotransmissores (substâncias químicas) na fenda sináptica, os quais se difundem e se ligam a receptores no neurônio pós-sináptico.
- ✓ Essa ligação entre neurotransmissores e receptores conduz à abertura dos canais iônicos, permitindo a entrada de íons na célula. A diferença de potencial resultante apresenta a forma de um pulso elétrico.
- ✓ Esses pulsos elétricos se propagam pelo neurônio pós-sináptico e são integrados no corpo celular. A ativação do neurônio pós-sináptico irá se dar no caso do efeito resultante destes pulsos elétricos integrados ultrapassar um dado *limiar*.
- ✓ Alguns neurotransmissores possuem a capacidade de *ativar* um neurônio enquanto outros possuem a capacidade de *inibir* a ativação do neurônio.

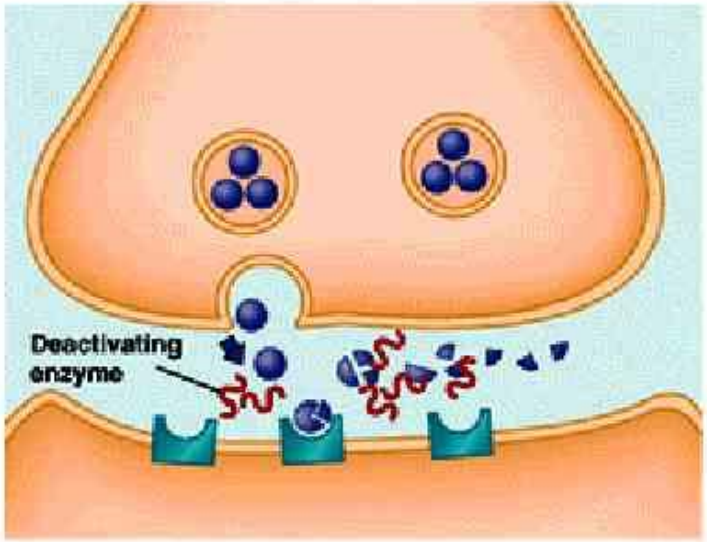


- A sinapse é uma fenda entre os terminais pré-sináptico e pós-sináptico, medindo ~20 nm.

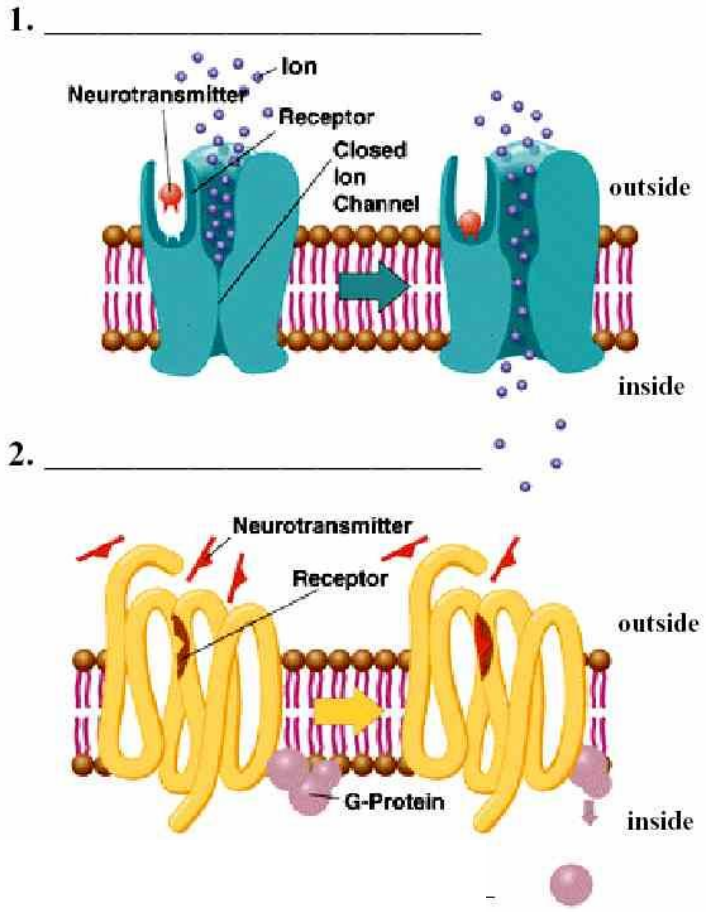

Action potential arrives
synaptic vesicle
Vesicle docks to membrane
Ca++ enters cell
Vesicle fuses
NT released



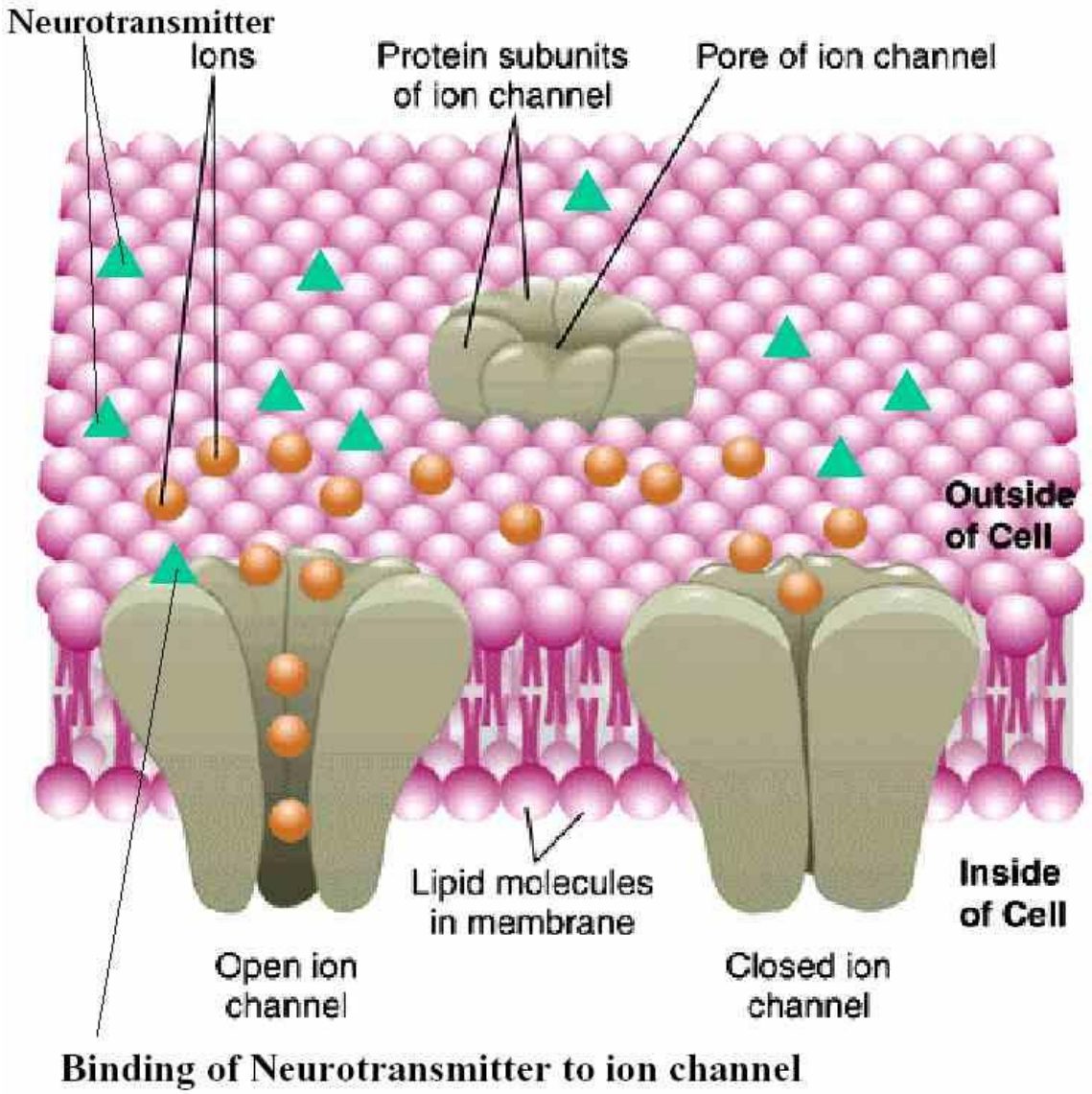
Reuptake



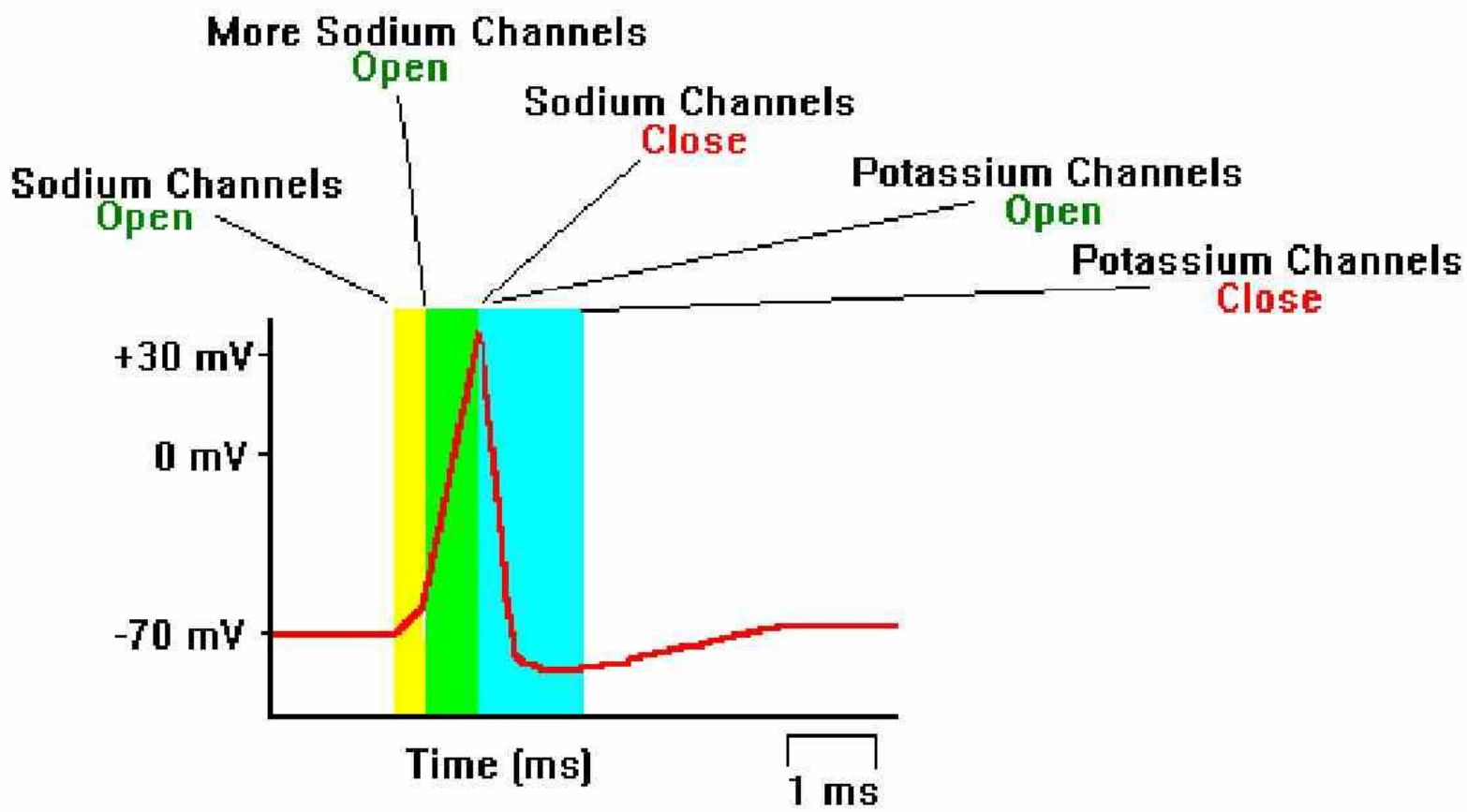
Deactivating Enzymes



- Neurotransmissores putativos: serotonina, endorfina, dopamina, etc. Ao todo, são mais de 30 compostos orgânicos.
- O mal de Parkinson, por exemplo, é atribuído a uma deficiência de dopamina.


Neurotransmitter
Ions
Protein subunits
of ion channel
Pore of ion channel
Outside
of Cell
Inside
of Cell
Lipid molecules
in membrane
Open ion
channel
Closed ion
channel
Binding of Neurotransmitter to ion channel

- A ativação de um neurônio é também denominada de *spiking*, *firing*, ou *disparo de um potencial de ação* (*triggering of an action potential*).

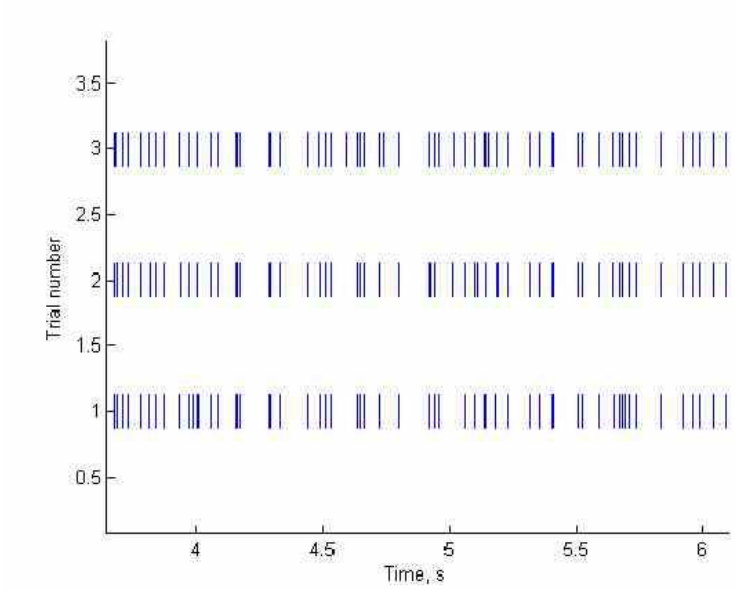


- Passos envolvidos no estabelecimento e extinção do potencial de ação:

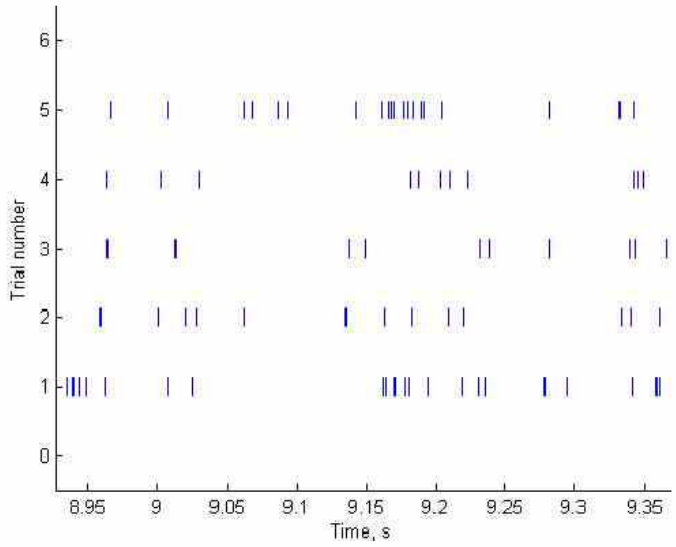
    1. Em uma célula em repouso, a parte externa da membrana é mais positiva que a parte interna, havendo mais íons de potássio dentro da célula e mais íons de sódio fora da célula.
    2. Pela ação dos neurotransmissores na sinapse, íons de sódio se movem para dentro da célula, causando uma diferença de potencial denominada potencial de ação. Com esta entrada de íons de sódio, o interior da célula passa a ser mais positivo que o exterior.
    3. Em seguida, íons de potássio fluem para fora da célula, restaurando a condição de interior mais negativo que exterior.
    4. Com as bombas de sódio-potássio, é restaurada finalmente a condição de maior concentração de íons de potássio dentro da célula e maior concentração de íons de sódio fora da célula.

- Segue-se um período refratário, durante o qual a membrana não pode ser estimulada, evitando assim a retropropagação do estímulo.

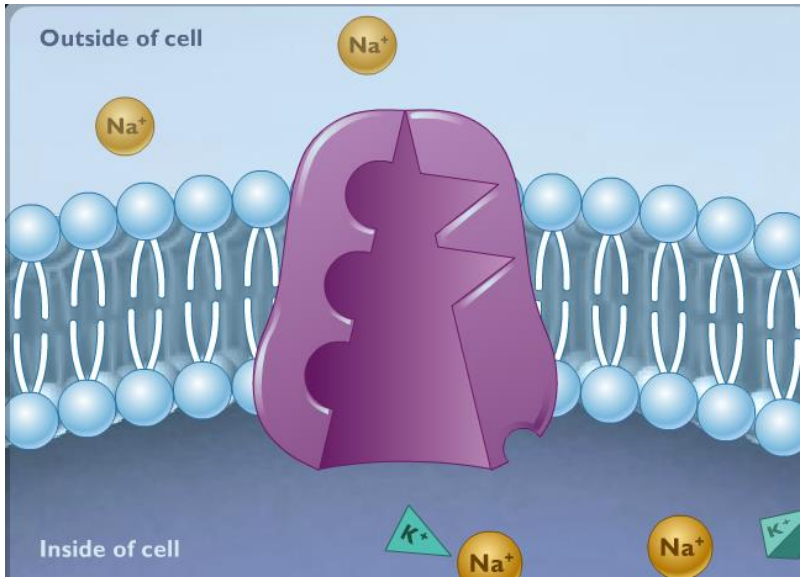
- Bombas de sódio e potássio: os íons de sódio que haviam entrado no neurônio durante a despolarização, são rebombeados para fora do neurônio mediante o funcionamento das bombas de sódio e potássio, que exigem gasto de energia.
- Para cada molécula de ATP empregada no bombeamento, 3 íons de sódio são bombeados para fora e dois íons de potássio são bombeados para dentro da célula. Esta etapa ocorre após a faixa azul da figura anterior.

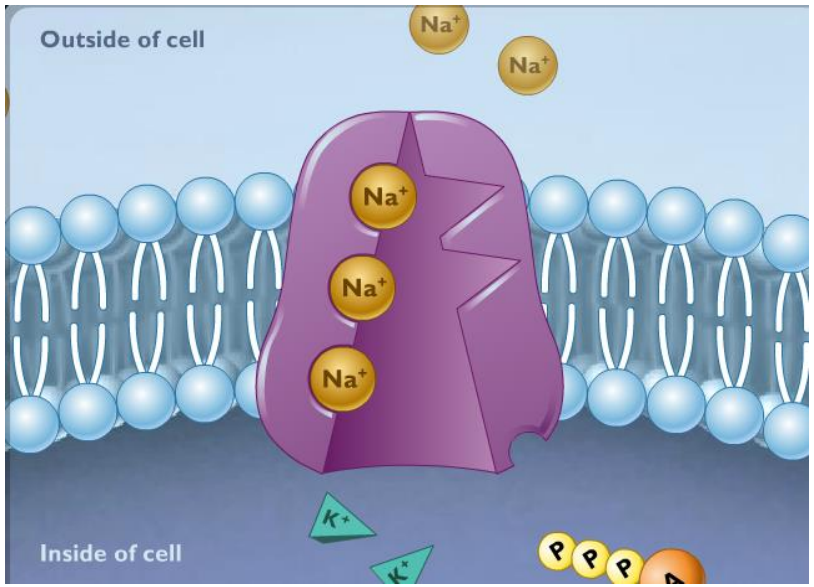


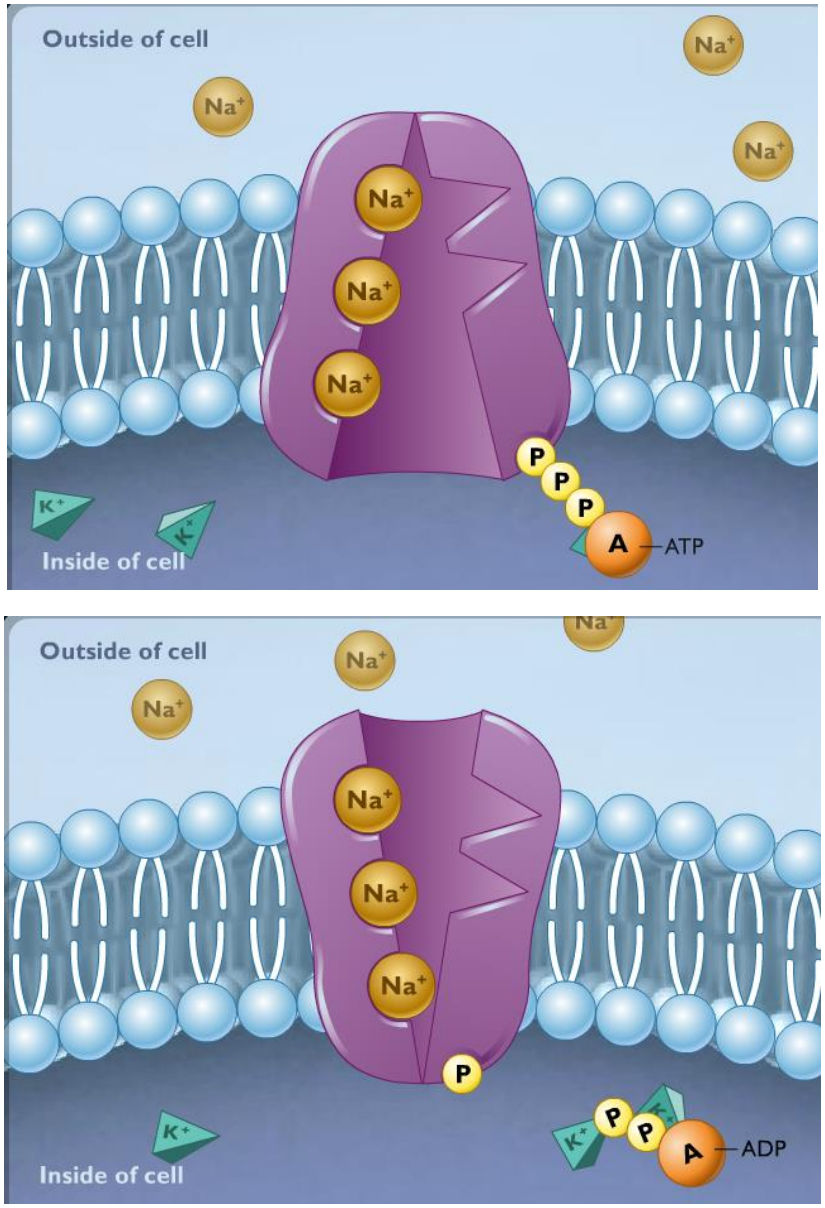
Neurônio periférico

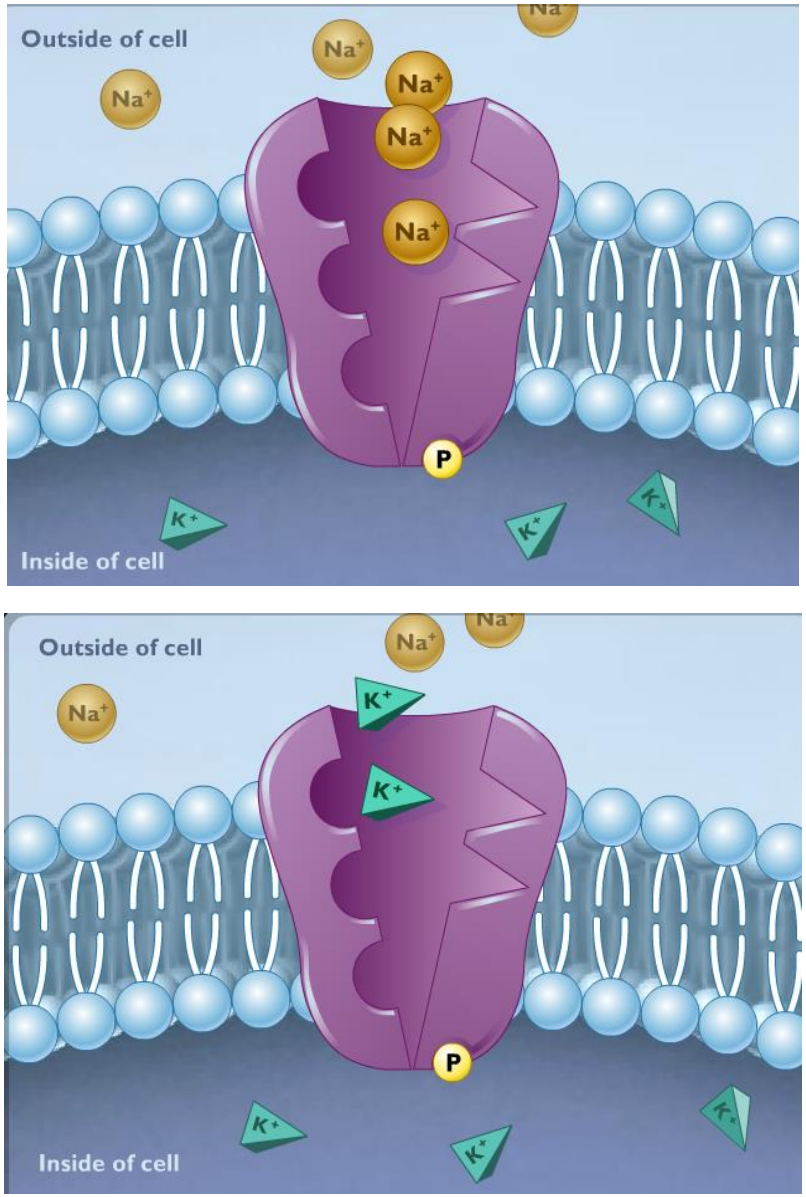


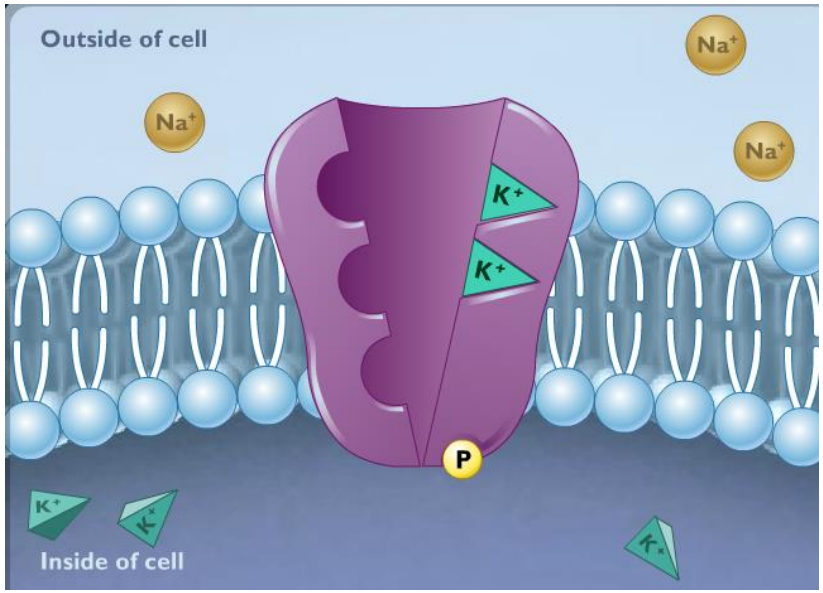
Neurônio do córtex

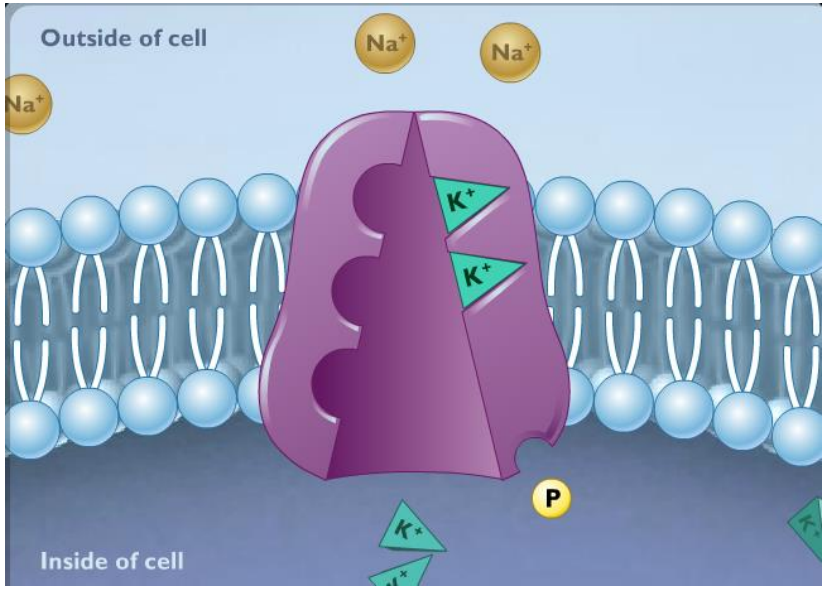

Outside of cell
Na+
Na+
Inside of cell
K+
Na+
Na+
K+

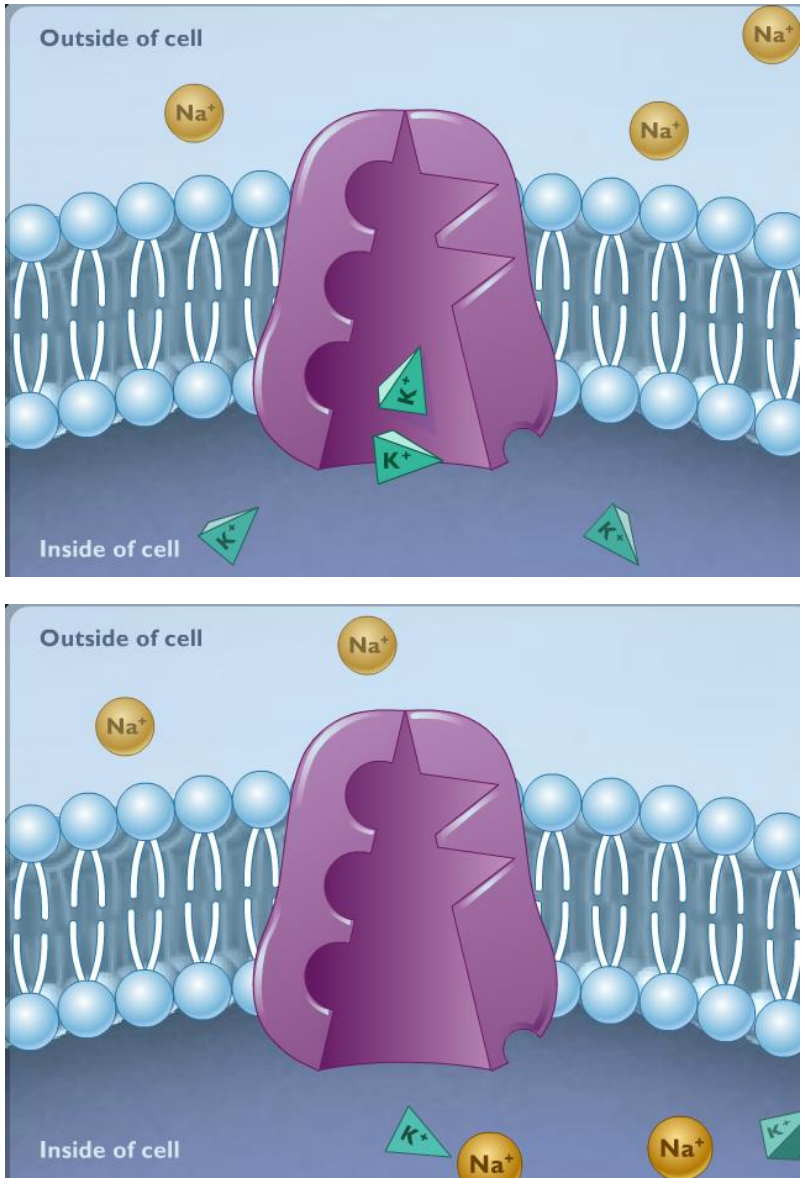

Outside of cell
Na+
Na+
Na+
Na+
Na+
Inside of cell
K+
K+
P
P
P


Outside of cell
Na+
Na+
Na+
Na+
Na+
Na+
P
P
P
A
ATP
K+
K+
Inside of cell
Outside of cell
Na+
Na+
Na+
Na+
Na+
P
K+
K+
P
P
A
ADP
Inside of cell

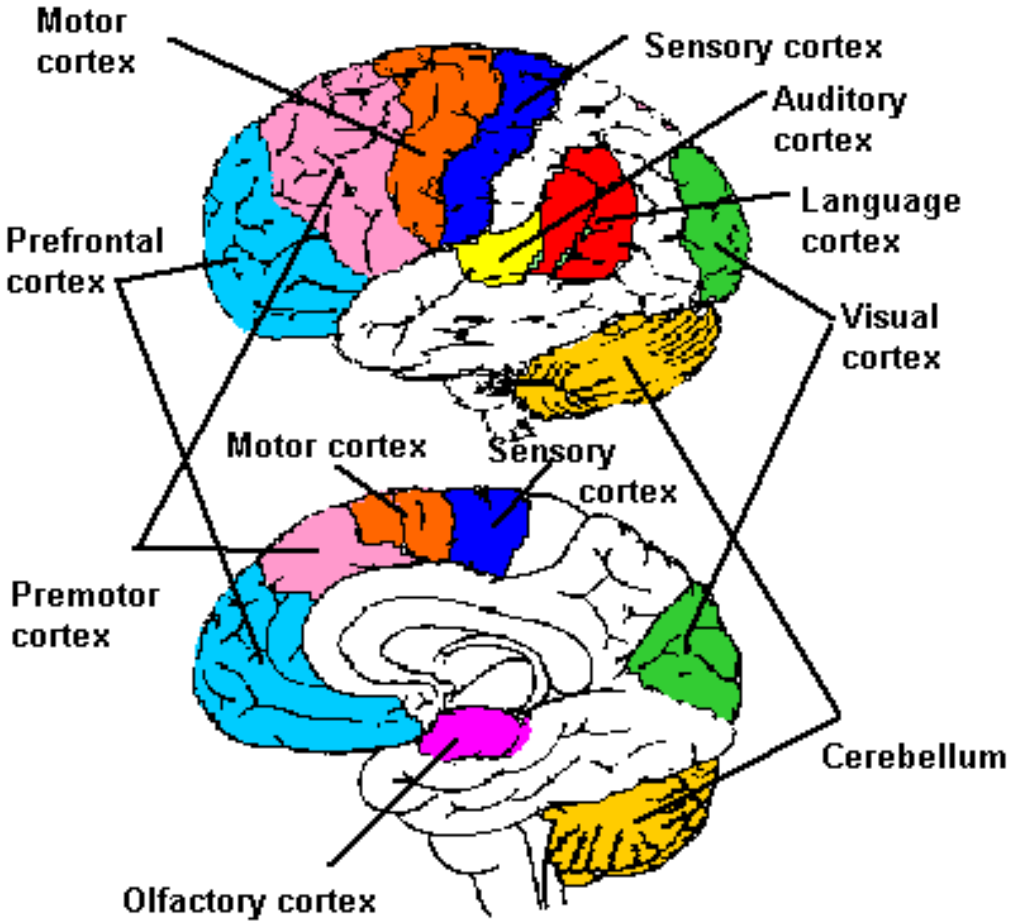

Outside of cell
Na+
Na+
Na+
Na+
Na+
P
K+
K+
Inside of cell
Outside of cell
Na+
Na+
K+
K+
Na+
P
K+
K+
Inside of cell

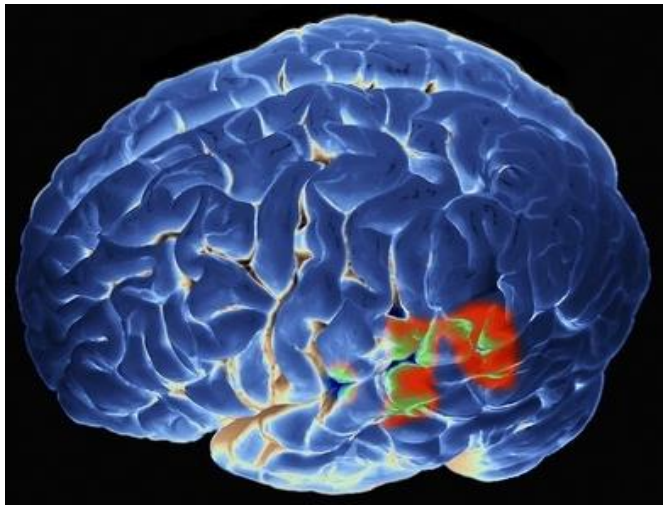

Outside of cell
Na+
Na+
Na+
K+
K+
P
K+
K+
K+
Inside of cell


Outside of cell
Na+
Na+
Na+
K+
K+
P
K+
Inside of cell

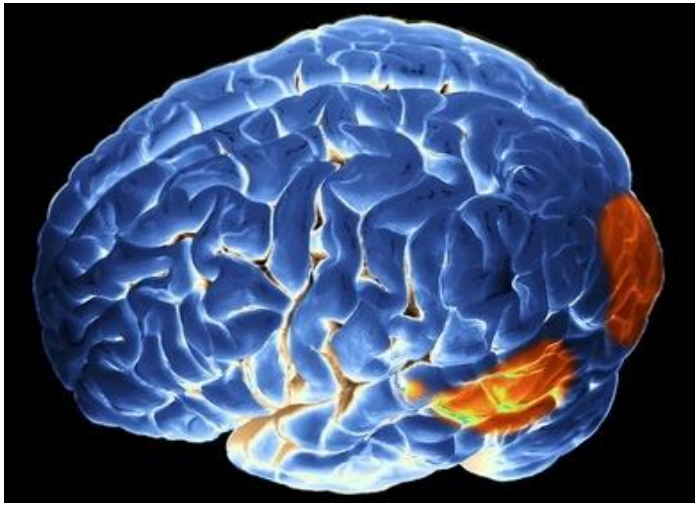

Outside of cell
Na+
Na+
Na+
K+
K+
K+
K+
Inside of cell
Outside of cell
Na+
Na+
K+
Na+
Na+
Inside of cell

- O córtex corresponde à superfície externa do cérebro: uma estrutura predominantemente bi-dimensional com vários dobramentos, fissuras e elevações.
- Diferentes partes do córtex possuem diferentes funções (ver figura abaixo).



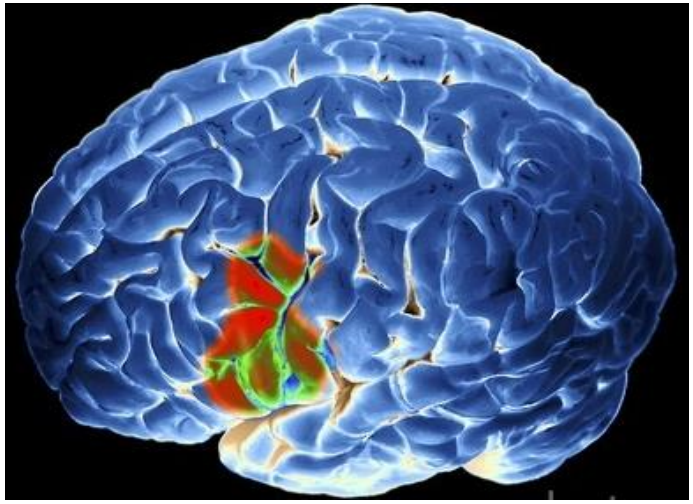


Ouvindo palavras



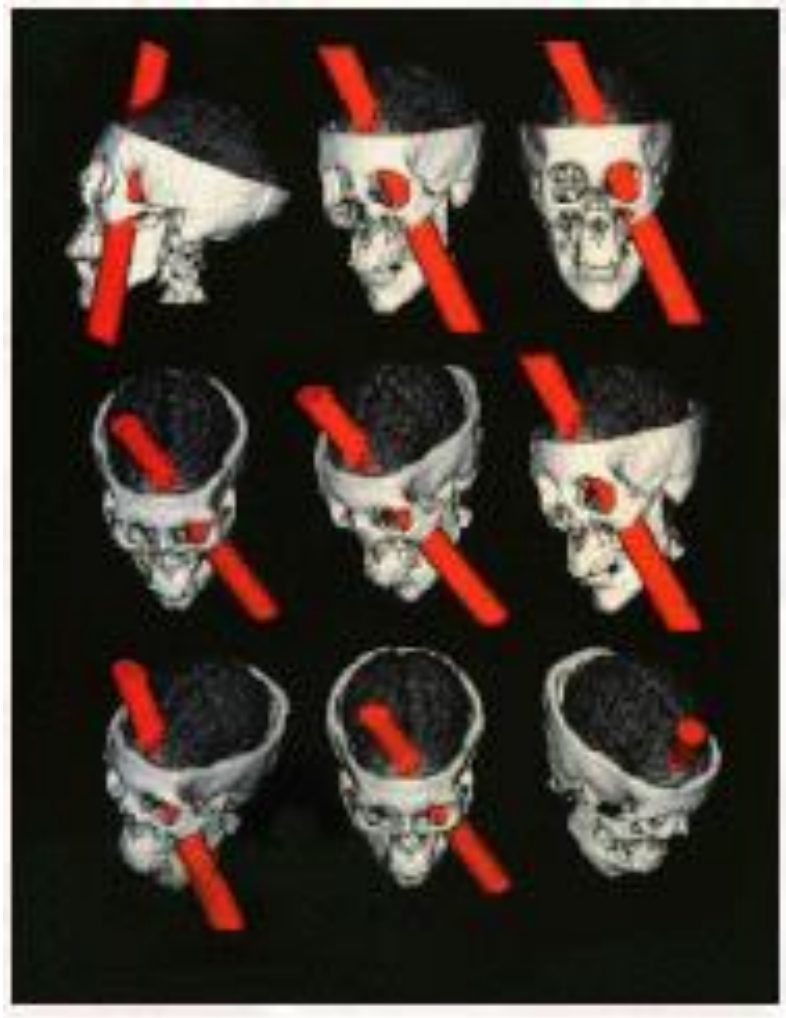
Lendo palavras

Falando palavras

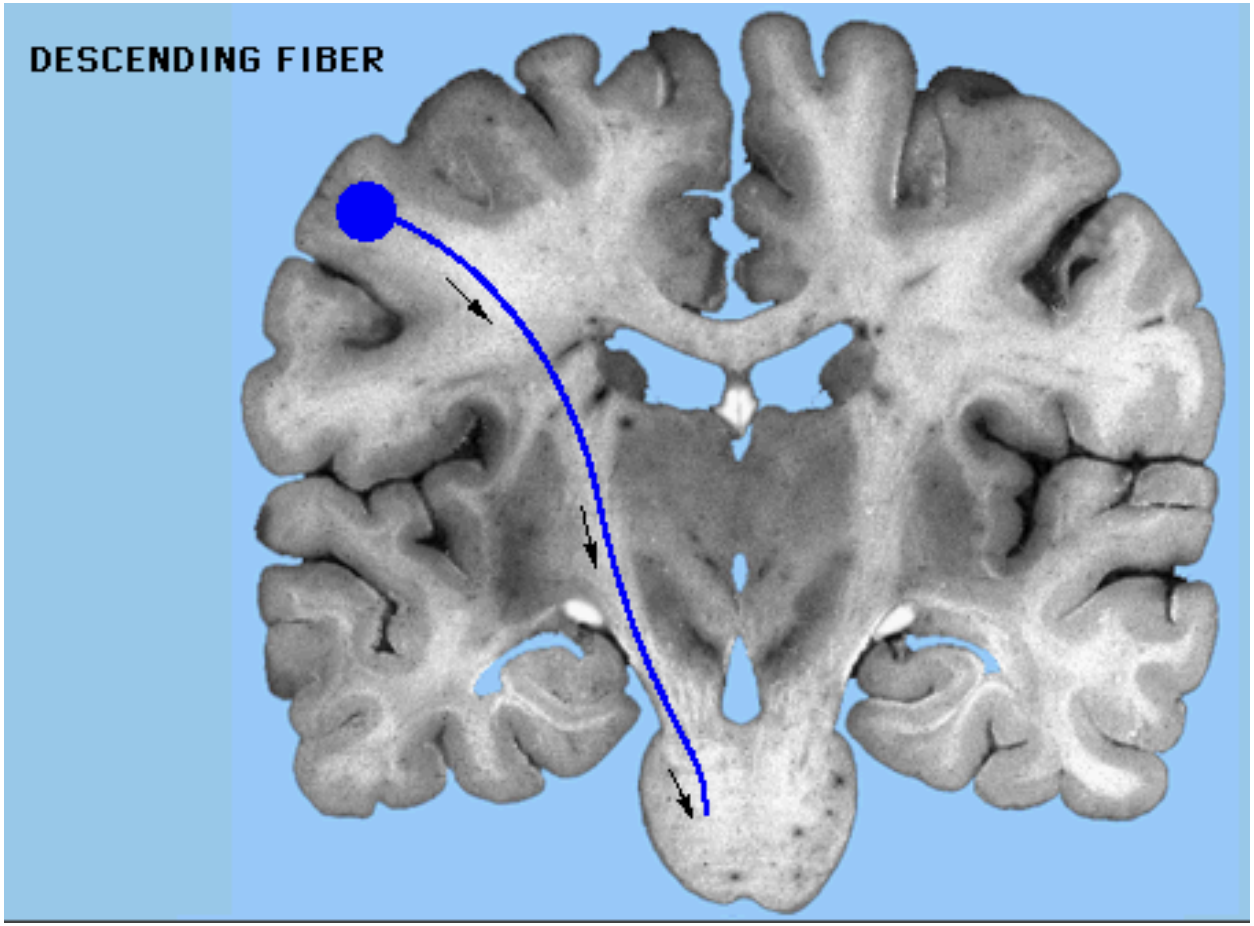


Pensando em palavras

Functional magnetic resonance imaging or functional MRI (*fMRI*)



Reconstrução computadorizada da destruição do cérebro de Phineas Gage por uma barra de ferro no ano de 1848.



Presença de vias de comunicação no cérebro

7.2 Base Biológica e Física da Aprendizagem e Memória

- O sistema nervoso está continuamente sofrendo modificações e atualizações. Virtualmente todas as suas funções, incluindo percepção, controle motor, regulação térmica e raciocínio, são modificadas por estímulos.
- Observações comportamentais permitiram verificar graus de plasticidade do sistema nervoso: existem mudanças rápidas e superficiais, mudanças lentas e profundas, e mudanças mais permanentes (porém, ainda modificáveis).
- Em geral, efeitos globais são resultantes de alterações locais nos neurônios.
- Existem diversas formas de modificação em uma rede neural:
  - ✓ Dendritos podem ser criados, assim como podem ser eliminados;
  - ✓ Alguns dendritos e o axônio podem se esticar ou ser encolhidos permitindo ou eliminando, respectivamente, a conexão com outras células;
  - ✓ Novas sinapses podem ser criadas ou sofrerem alterações;
  - ✓ Sinapses também podem ser removidas;

- ✓ Todo neurônio pode sofrer regeneração ou pode morrer;
- ✓ Novos neurônios podem ser gerados e incorporados ao sistema nervoso.

- Toda esta vasta gama de adaptações estruturais pode ser convenientemente condensada simplesmente referindo-se às sinapses, pois estas modificações envolvem a modulação sináptica de forma direta ou indireta. Sendo assim, a *aprendizagem via modulação sináptica* é o mecanismo mais importante para as redes neurais, sejam elas biológicas ou artificiais.
- A modulação sináptica poderá depender de mecanismos de adaptação de neurônios individuais e de redes neurais como um todo.
- Assim como a aprendizagem, a *memória* também é resultado de um processo adaptativo das sinapses. Ela é causada por variações da eficiência sináptica de alguns neurônios, como resultado da atividade neural.
- Estas alterações resultam em caminhos novos ou facilitados de desenvolvimento e transmissão de sinais através dos circuitos neurais.

- Na verdade, um dos resultados de um processo de aprendizagem é a criação de um padrão de conexões sinápticas mais permanente, que por sua vez resulta na memorização (aprendizagem) de uma determinada experiência.
- Note, portanto, que a diferença entre aprendizagem e memória é sutil: a aprendizagem pode ser vista como o processo adaptativo que resulta em uma mudança da eficiência e estrutura sináptica, enquanto a memória pode ser interpretada como o resultado deste processo adaptativo.

8. Neurônio artificial

- Modelo matemático: Simplificações da realidade com o propósito de representar aspectos relevantes de um sistema em estudo, sendo que detalhes de menor significância são descartados para viabilizar a modelagem.

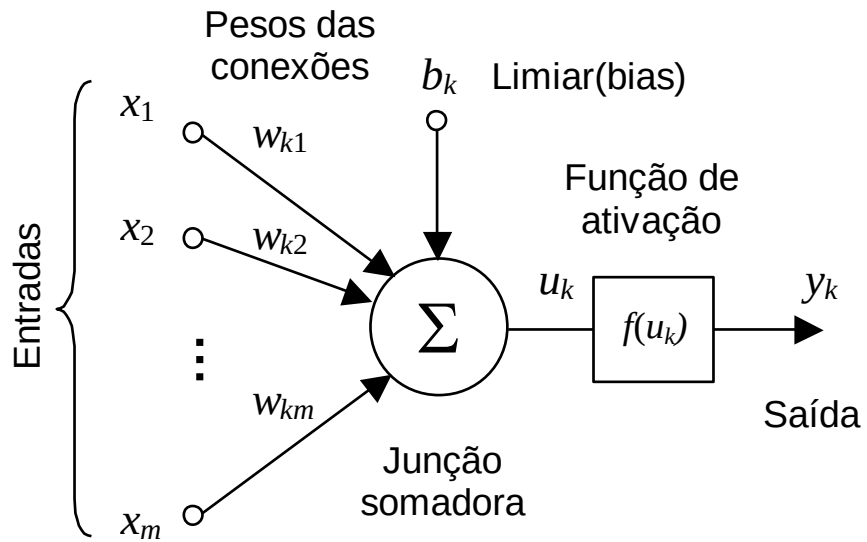


Figura 1 – Modelo matemático de um neurônio artificial

- A saída do neurônio *k* pode ser descrita por: $y_k = f(u_k) = f\left(\sum_{j=1}^{m} w_{kj} x_j + b_k\right)$

- É possível simplificar a notação acima de forma a incluir o bias simplesmente definindo um sinal de entrada de valor $x_0 = 1$ com peso associado $w_{k0} = b_k$:

$$y_k = f(u_k) = f\left(\sum_{j=0}^{m} w_{kj} x_j\right) = f\left(\mathbf{w}^T \mathbf{x}\right)$$

9. Exemplos mais usuais de funções de ativação

$$y_k = f(u_k) = \frac{e^{pu_k}}{e^{pu_k} + 1} = \frac{1}{1 + e^{-pu_k}} \qquad \frac{\partial y_k}{\partial u_k} = p y_k (1 - y_k) > 0$$

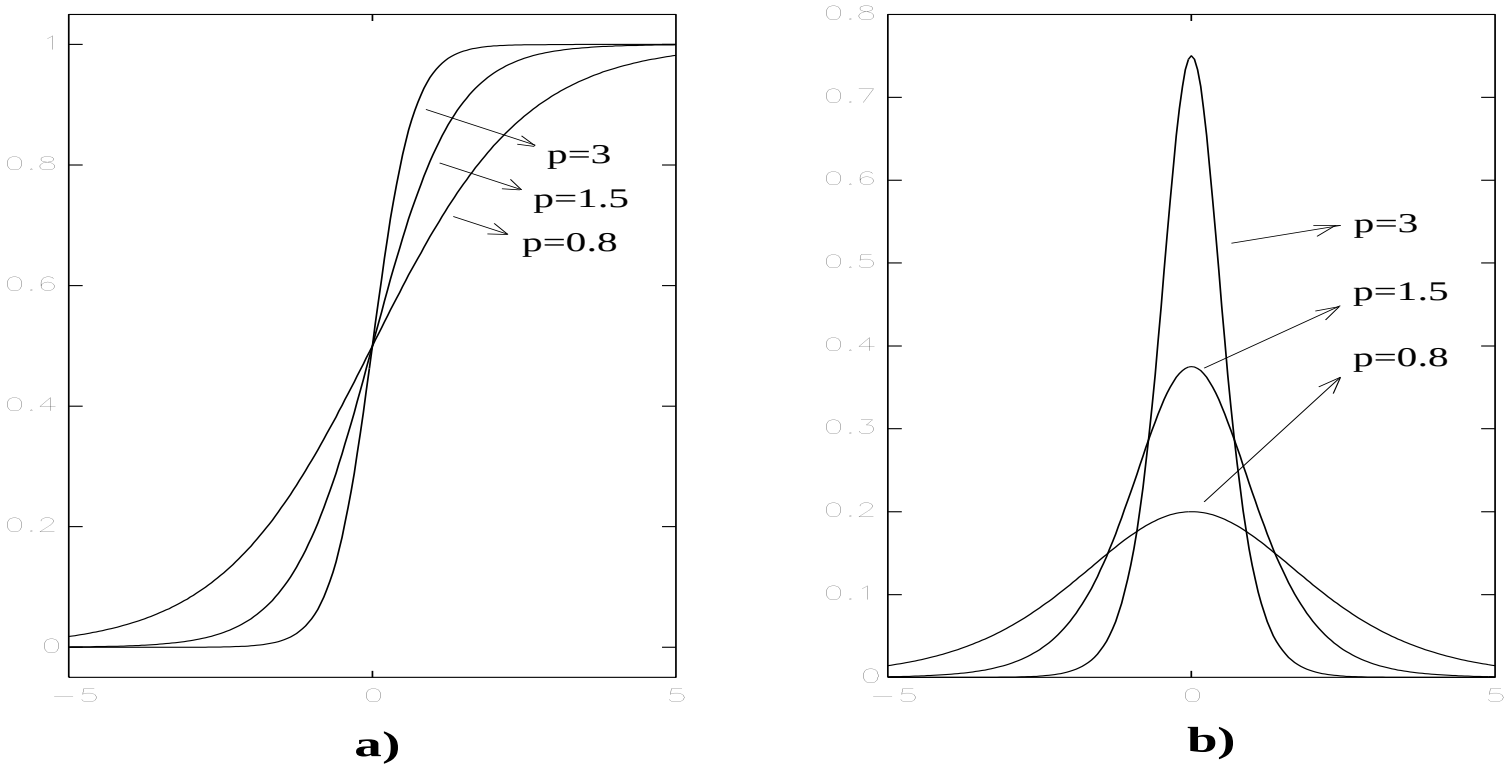


Figura 2 – Função logística (a) e sua derivada em relação à entrada interna (b)

$$y_k = f(u_k) = \tanh(pu_k) = \frac{e^{pu_k} - e^{-pu_k}}{e^{pu_k} + e^{-pu_k}} \qquad \frac{\partial y_k}{\partial u_k} = p\left(1 - y_k^2\right) > 0$$

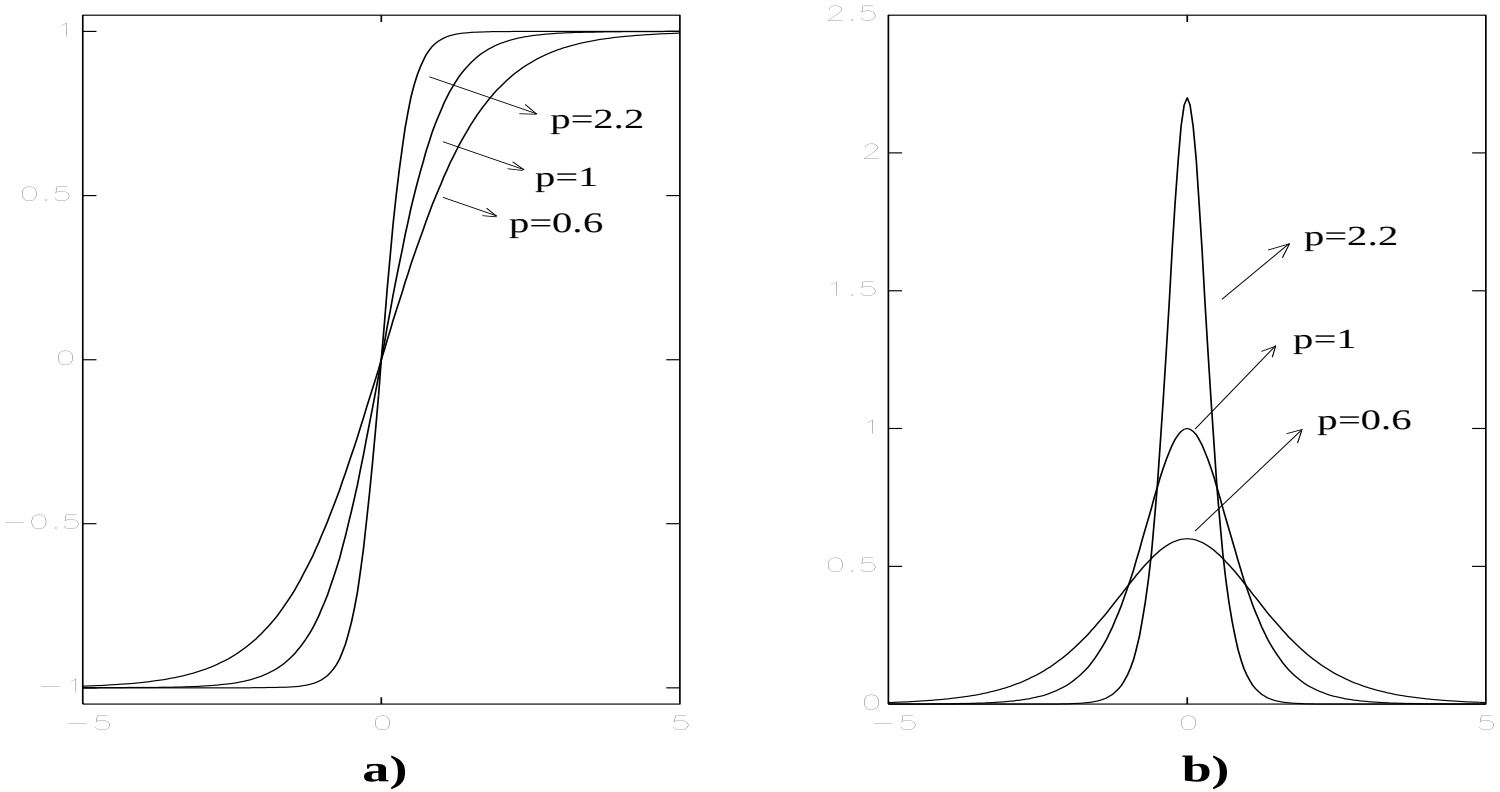
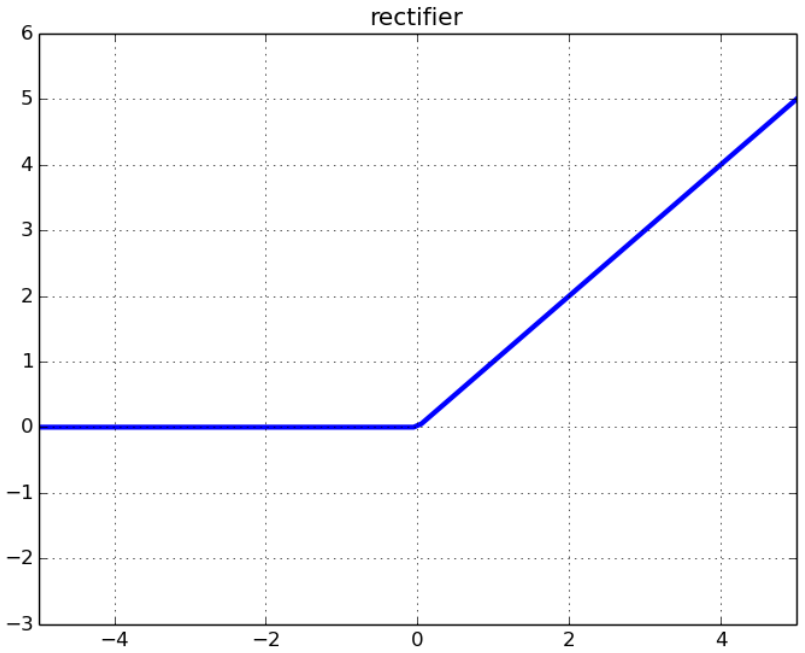


Figura 3 – Função tangente hiperbólica (a) e sua derivada em relação à entrada interna (b)

- Uma função de ativação que passou a ser largamente utilizada em *deep learning* é a Rectified Linear Unit (ReLU), apresentada a seguir e cuja equação é dada por:

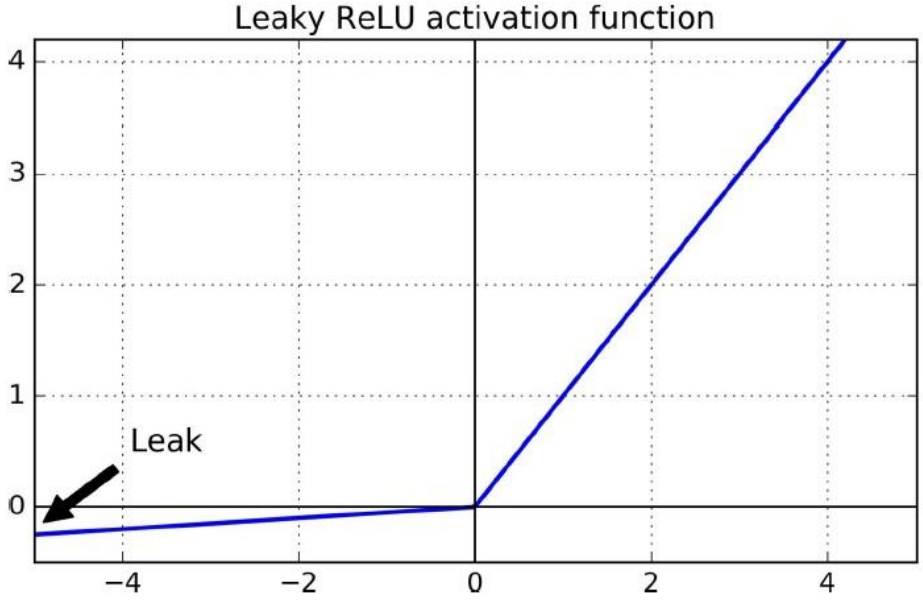
$$y_k = f(u_k) = \max(0, u_k)$$

- Trata-se de uma função linear por partes, que facilita o cálculo do gradiente e que opera como uma porta que deixa ou não passar adiante a ativação interna do neurônio.



- No entanto, a ReLU apresenta uma desvantagem em potencial. Caso muitos neurônios ativem com valores negativos para muitos estímulos de entrada, eles deixam de participar do processamento da rede neural e não podem ter suas conexões sinápticas ajustadas, pois a função de ativação tem derivada nula. Com isso, uma opção válida é adotar a função Leaky ReLU, na forma:

$$y_k = \begin{cases} u_k & \text{se } u_k > 0 \\ \alpha u_k & \text{se } u_k \leq 0 \end{cases}, \text{ para } \alpha \text{ próximo de zero.}$$



10. Produto interno e projeção

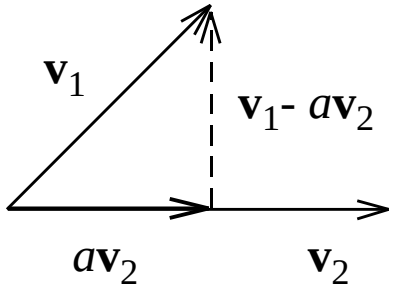


Figura 4 – Projeção realizada pelo produto interno no $\Re^2$

- Sejam $\mathbf{v}_1, \mathbf{v}_2 \in \Re^2$ elementos não-nulos. Considere um escalar a tal que $a\mathbf{v}_2$ corresponda à projeção de $\mathbf{v}_1$ na direção de $\mathbf{v}_2$. Então, pode-se afirmar que

$$a\mathbf{v}_2 \perp \mathbf{v}_1 - a\mathbf{v}_2,$$

  conduzindo a

$$\langle a\mathbf{v}_2, \mathbf{v}_1 - a\mathbf{v}_2 \rangle = 0 .$$

- Logo,

$$a\langle \mathbf{v}_2, \mathbf{v}_1 \rangle - a^2 \langle \mathbf{v}_2, \mathbf{v}_2 \rangle = 0,$$

permitindo obter *a* na forma

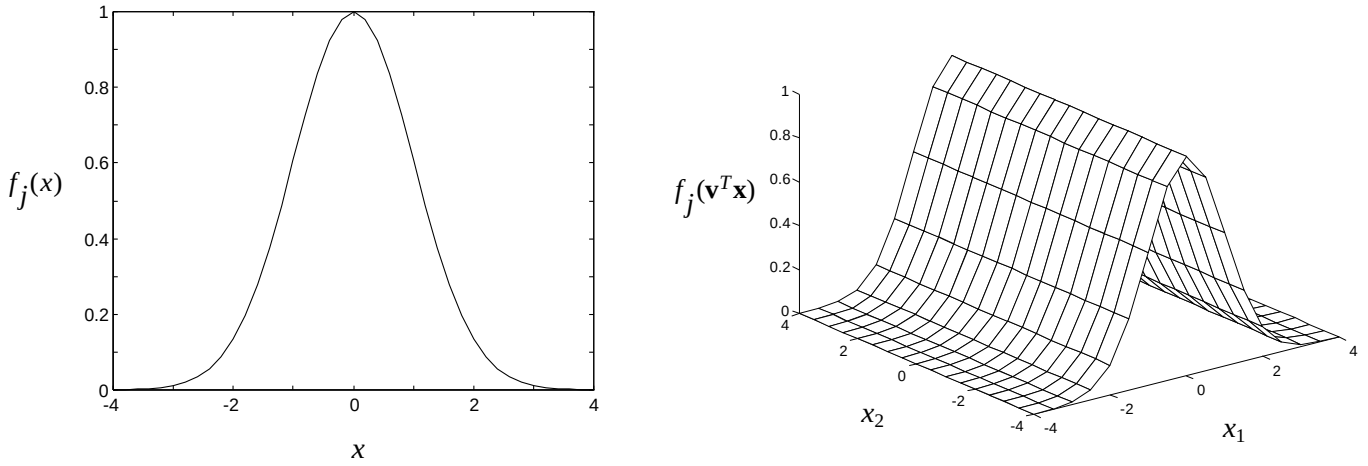
$$a = \frac{\langle \mathbf{v}_2, \mathbf{v}_1 \rangle}{\langle \mathbf{v}_2, \mathbf{v}_2 \rangle}.$$

- Isto significa que a projeção de $\mathbf{v}_1$ na direção de $\mathbf{v}_2$ ($\mathbf{v}_2 \neq \mathbf{0}$) assume a forma:

$$\text{proj}_{\mathbf{v}_2}(\mathbf{v}_1) = \frac{\langle \mathbf{v}_2, \mathbf{v}_1 \rangle}{\langle \mathbf{v}_2, \mathbf{v}_2 \rangle} \mathbf{v}_2$$

- Mantendo constante o módulo de $\mathbf{v}_1$, a sua projeção na direção de $\mathbf{v}_2$ é tão maior quanto mais colineares forem esses dois vetores.

11. Função de expansão ortogonal



(a) $f_j(x) = e^{-0,5 \cdot x^2}$ (b) $f_j(\mathbf{v}^T\mathbf{x}) = e^{-0,5 \cdot \left([1\ 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)^2}$

Figura 5 – Função de expansão ortogonal em que $\mathbf{v} = [1\ 0]^T$ e $\mathbf{x} = [x_1\ \ x_2]^T$

- A função de expansão ortogonal é conhecida na literatura em língua inglesa como *ridge function.*

12. Redes neurais e perceptron com uma camada intermediária

- O processo de conexão entre neurônios artificiais leva à geração de sinapses e à construção de redes neurais artificiais.

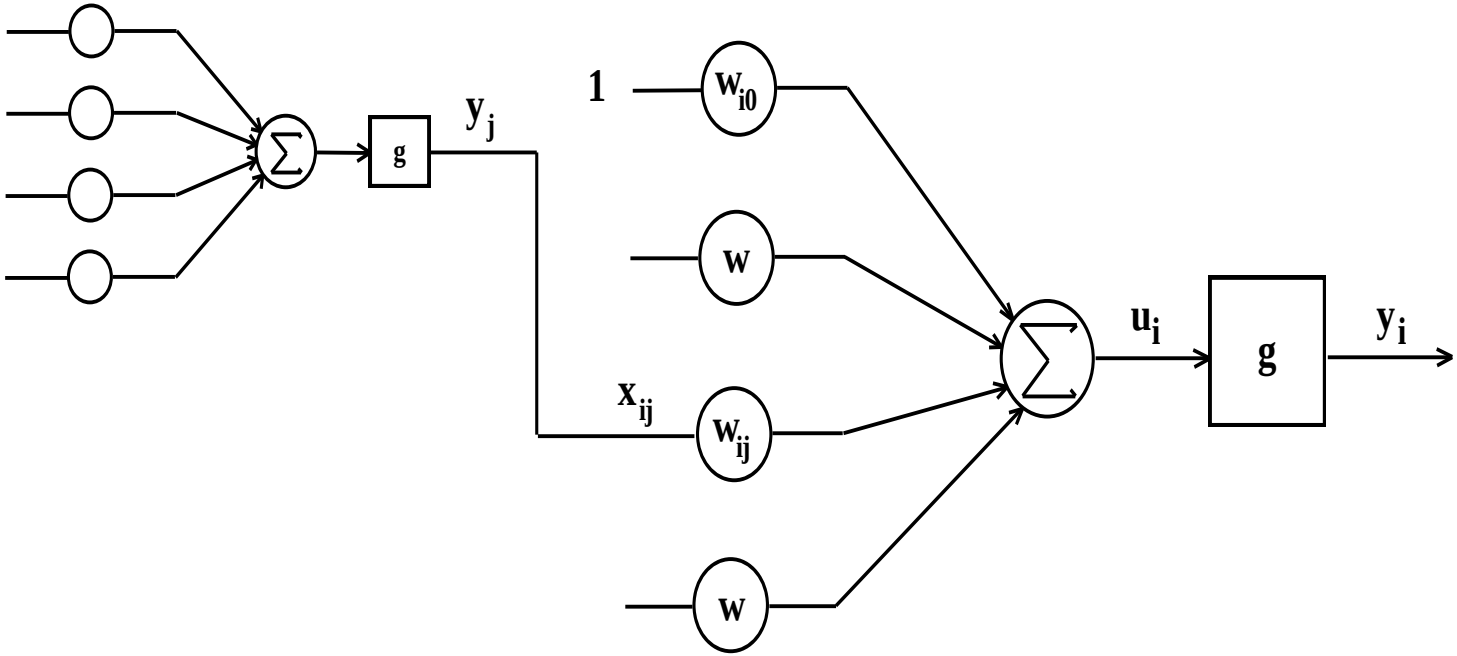


Figura 6 – Estabelecimento de conexão entre dois neurônios artificiais

- As estruturas mais conhecidas são em camadas, onde a saída de cada neurônio de uma camada precedente é entrada para todos os neurônios da camada seguinte.

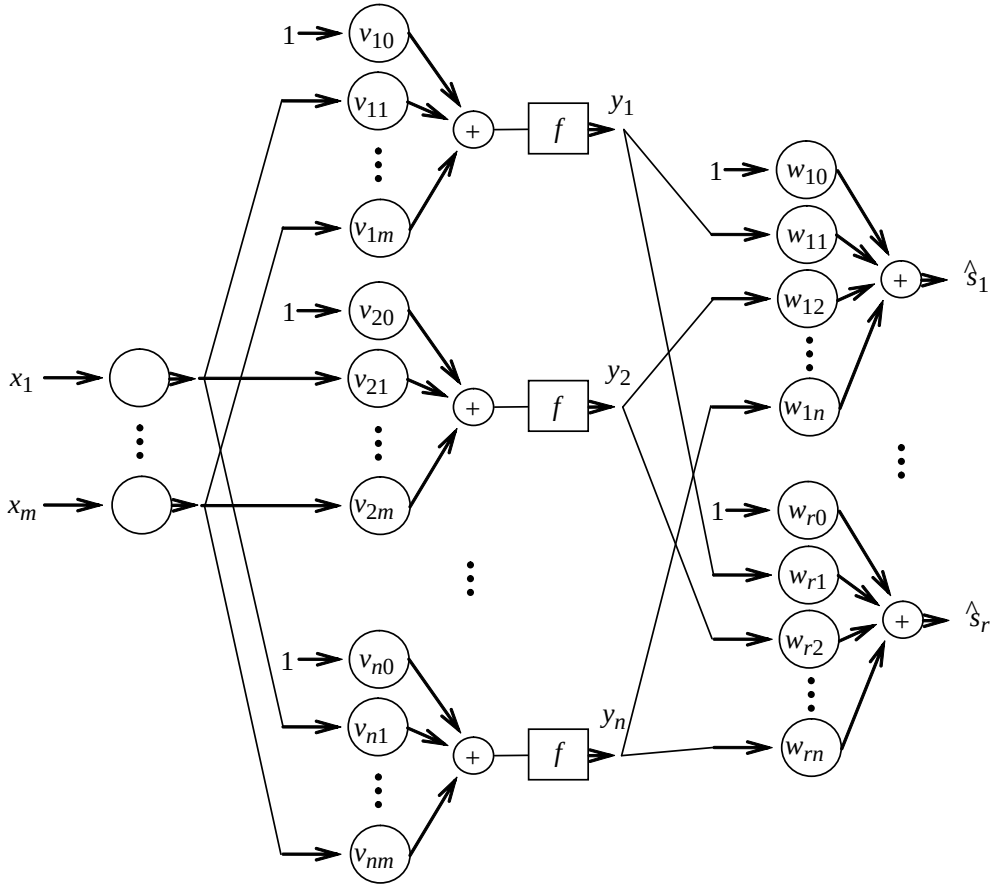
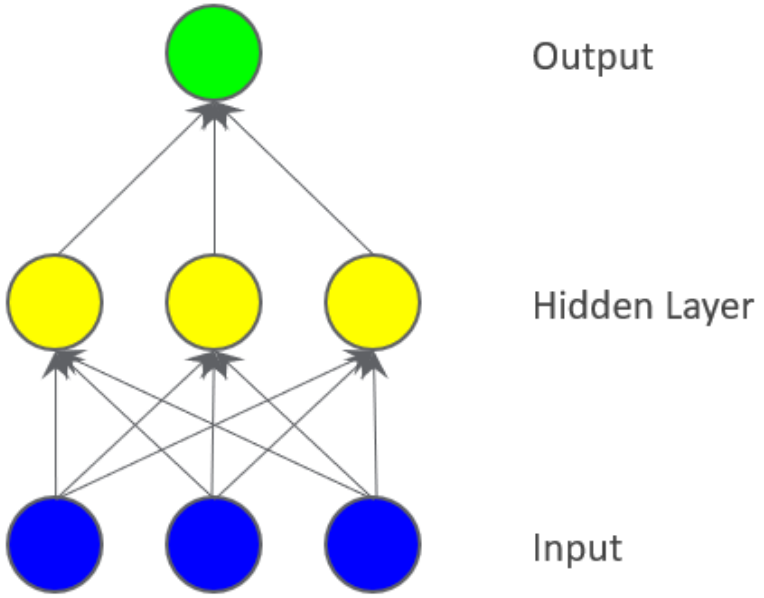


Figura 7 – Rede neural perceptron com uma camada intermediária
Do inglês *Multilayer Perceptron* (MLP)

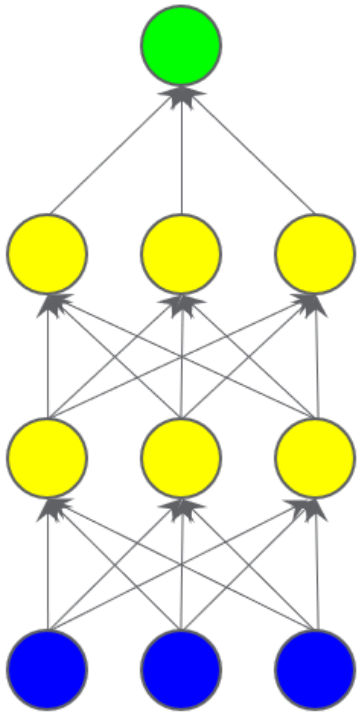
$$\hat{s}_k = \sum_{j=0}^{n} w_{kj} f\left(\sum_{i=0}^{m} v_{ji} x_i\right) = \sum_{j=0}^{n} w_{kj} f\left(\mathbf{v}_j^T \mathbf{x}\right) = \hat{g}_k(\mathbf{x}, \theta),\ k = 1,\ldots,r$$

13. Múltiplas camadas com função de ativação linear

- Iremos mostrar nesta seção que, na eventualidade de se considerar apenas funções de ativação lineares, o uso de múltiplas camadas intermediárias não trás nenhum benefício ao processamento.



$$y = Wx + b$$



$$y = Wx + b$$

$$z = Vy + c$$

$$z = V(Wx+b) + c$$

$$= VWx + (Vb + c)$$

which is just another linear operation

14. Contribuição de cada neurônio em uma rede MLP

- O mapeamento não-linear realizado por uma rede neural do tipo perceptron com uma camada intermediária é uma **combinação linear de funções de expansão ortogonal**, ou seja, funções que têm a forma de tangente hiperbólica em uma direção e são constantes nas demais direções ortogonais a esta única direção em que a forma da função se manifesta.
- Como um exemplo, vamos tomar amostras de um mapeamento do $\Re^2$ para o $\Re^1$, e utilizar uma rede neural com 5 neurônios na camada intermediária para buscar aproximar este mapeamento, o qual pode ser visualizado no $\Re^3$.
- Os pesos sinápticos resultantes do processo de treinamento estão apresentados na sequência, sendo que a rede neural tem ao todo $3 \times 5 + 6 \times 1 = 21$ pesos ajustáveis. São 2 entradas, 5 neurônios na camada intermediária e 1 saída, mais as entradas constantes (entradas de polarização) de todos os 6 neurônios da rede neural.

- Pesos sinápticos da camada intermediária após o treinamento (cada coluna representa os pesos de um neurônio):

-0.20008939714462 -0.70051908010040 0.39699221844113 -0.10003863267278 0.69606262467282

0.70018168528932 0.10015860417667 0.19860028823484 -0.29996195303800 0.29869112235480

-0.30006398146599 0.80022209855791 0.49372400421686 0.50005427222963 0.89515012131364

- Pesos sinápticos da camada de saída:

0.99989340388393

0.79971888341317

0.90007841696146

0.38564988369799

0.79996881679466

0.71442550587375

- Obs: O peso de polarização é o primeiro peso de cada neurônio.

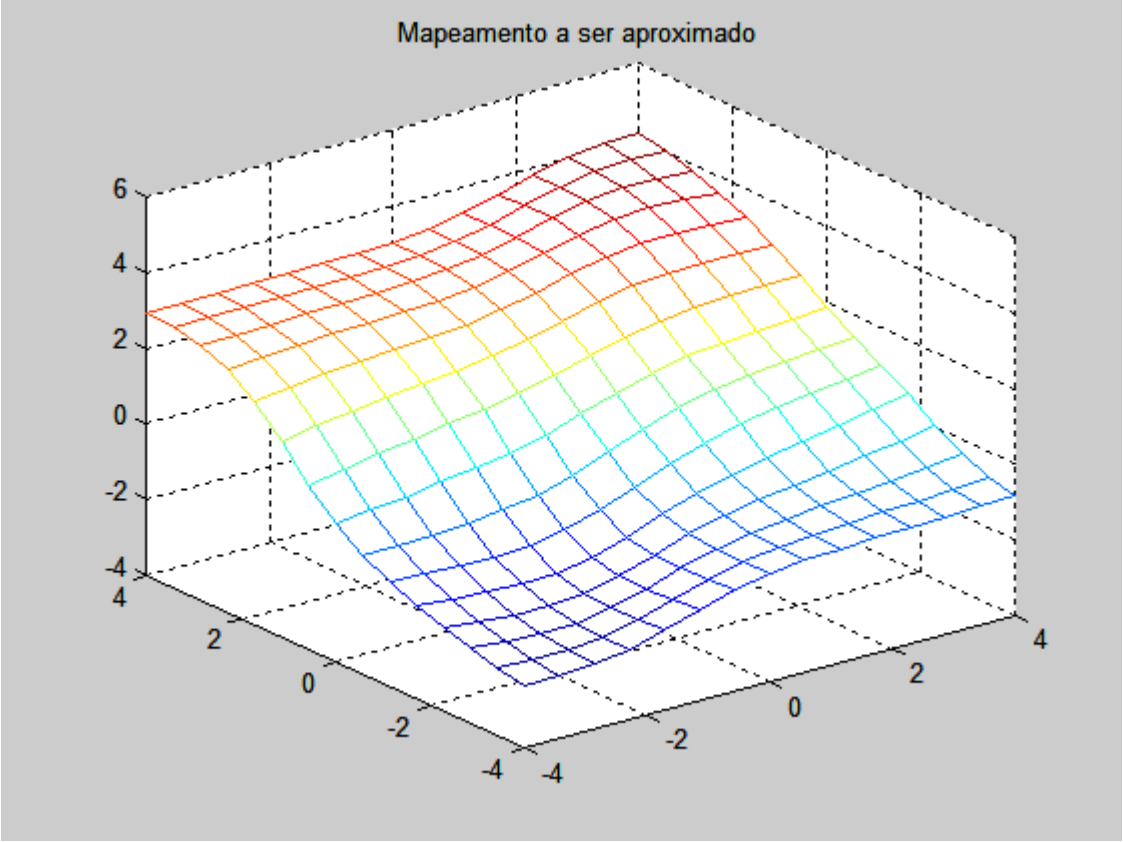


Figura 8 – Mapeamento a ser aproximado

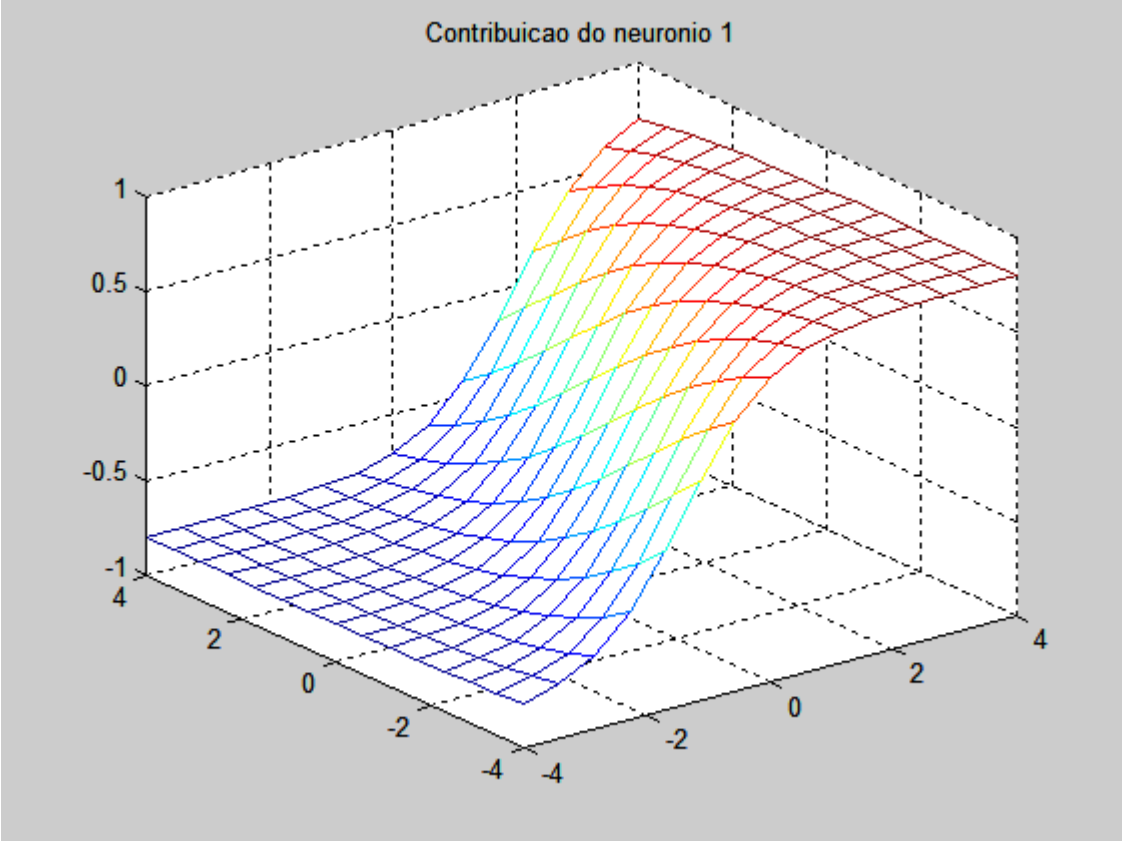


Figura 9 – Contribuição do neurônio 1, já multiplicada pelo peso do neurônio de saída.

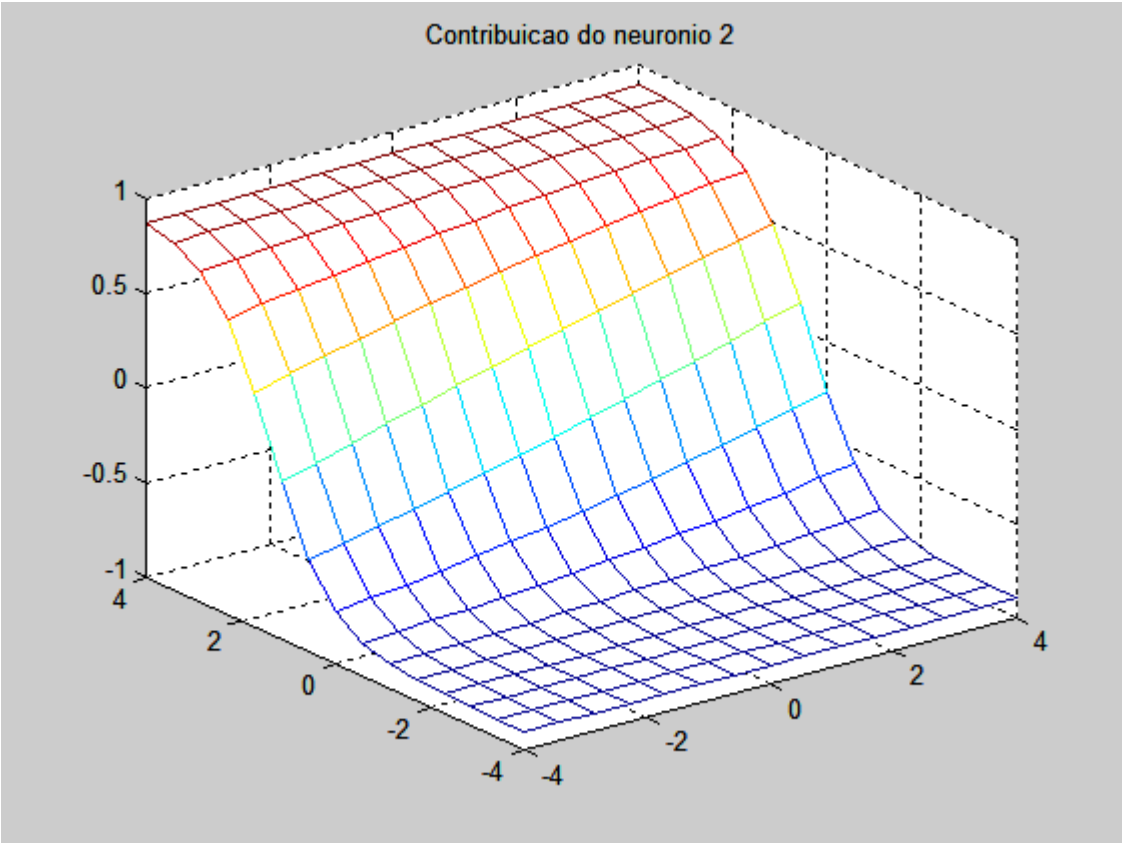


Figura 10 – Contribuição do neurônio 2, já multiplicada pelo peso do neurônio de saída.

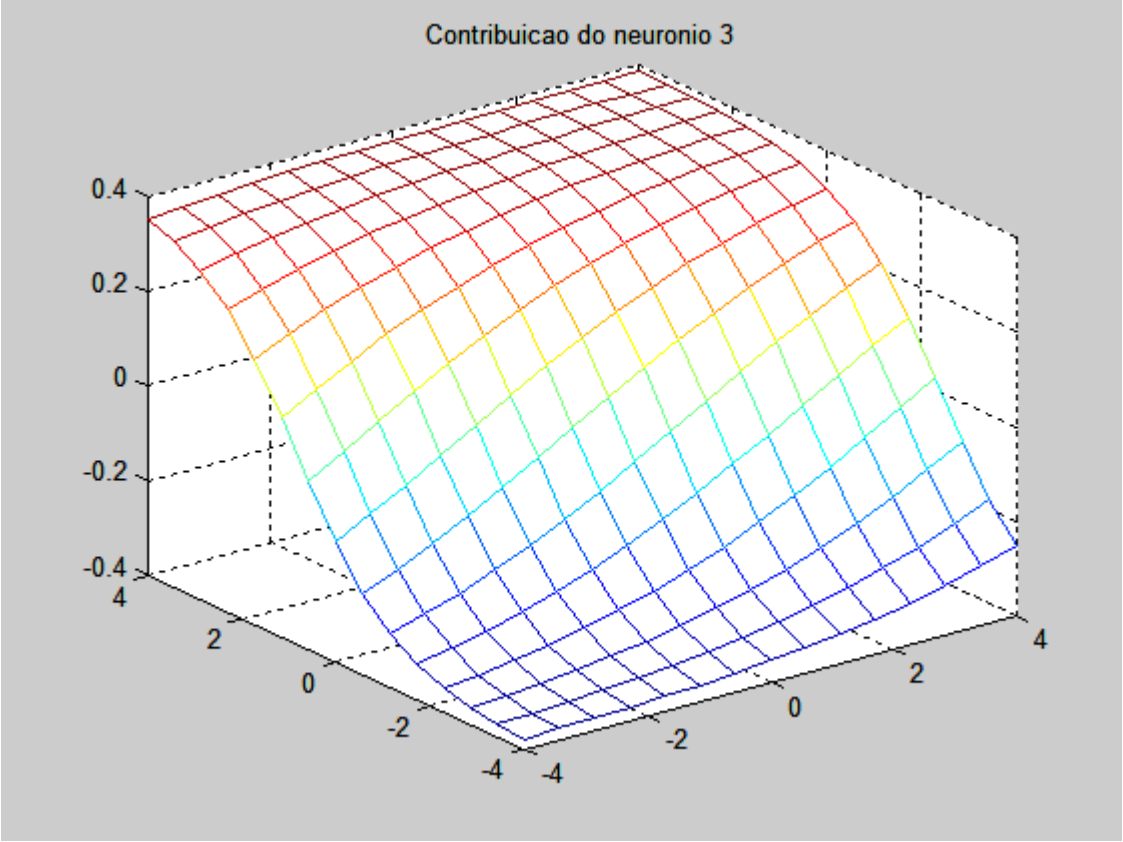


Figura 11 – Contribuição do neurônio 3, já multiplicada pelo peso do neurônio de saída.

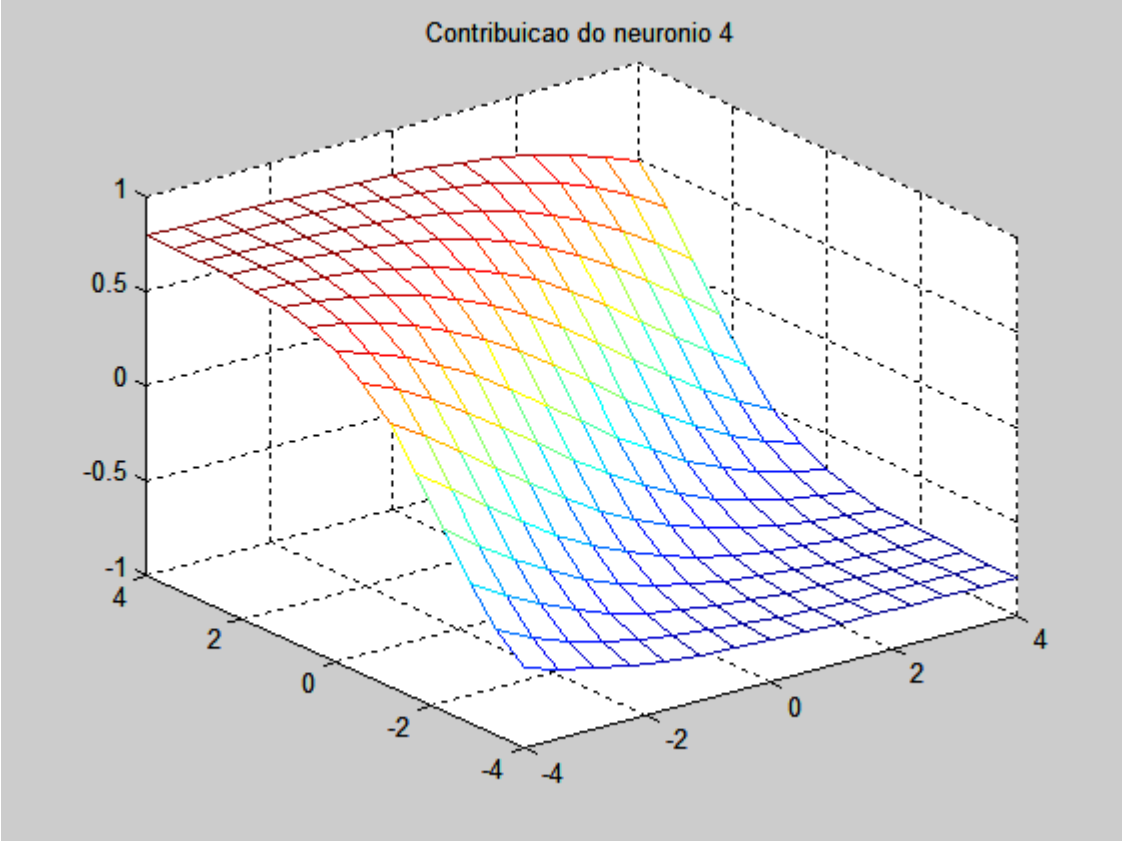


Figura 12 – Contribuição do neurônio 4, já multiplicada pelo peso do neurônio de saída.

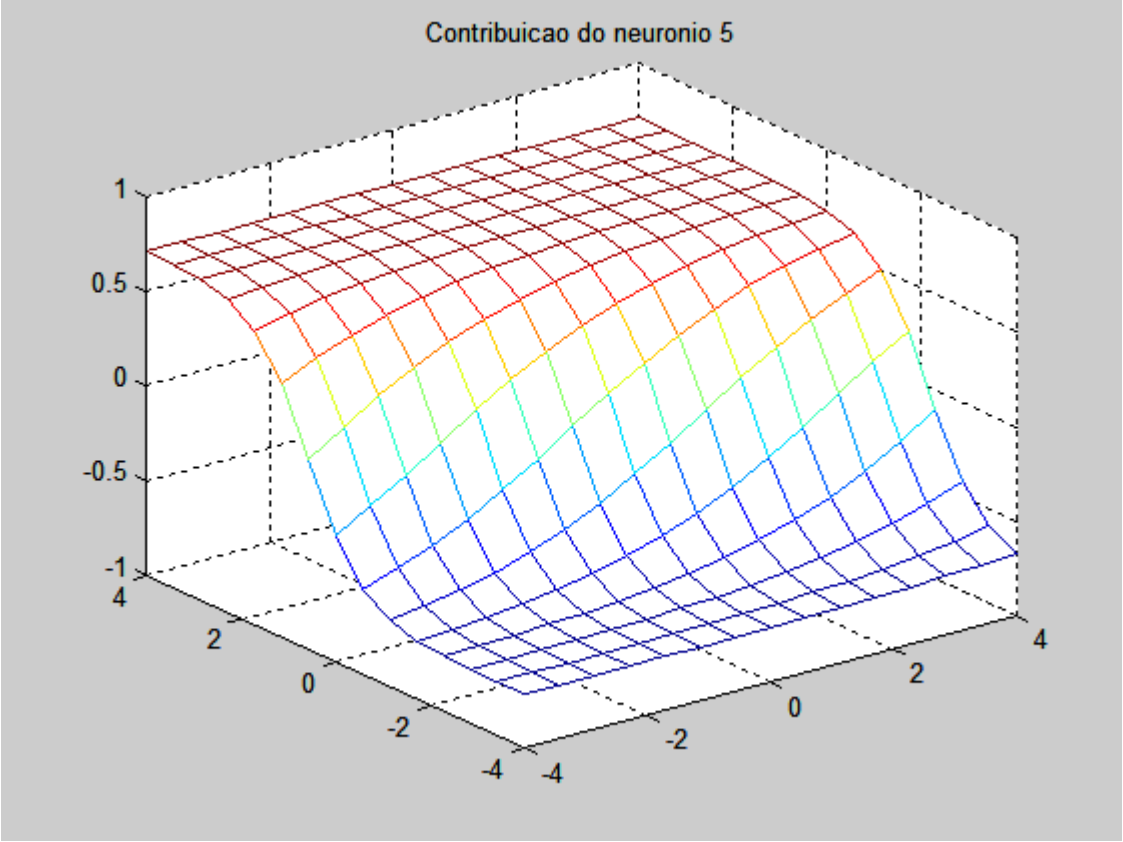
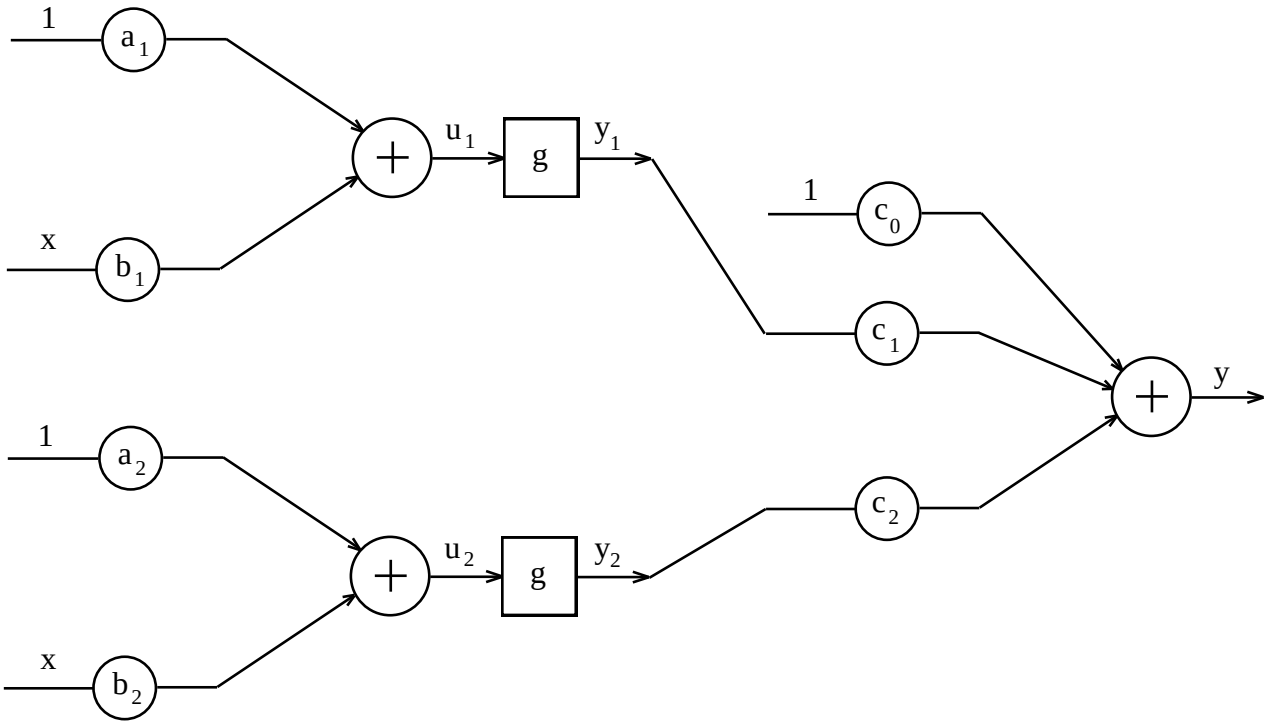


Figura 13 – Contribuição do neurônio 5, já multiplicada pelo peso do neurônio de saída.

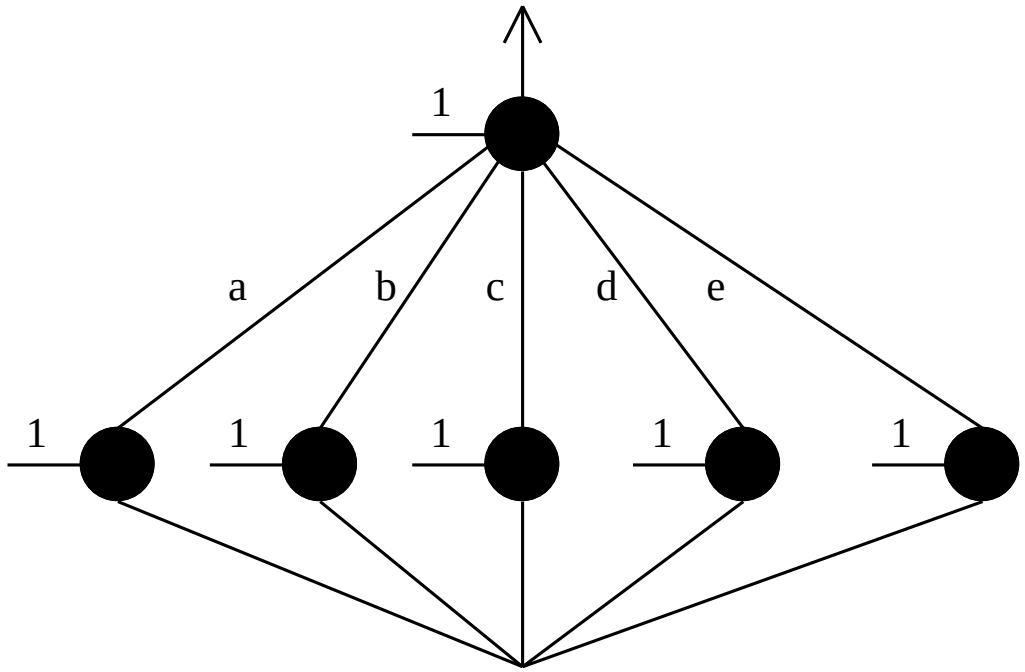
15. O papel dos pesos sinápticos

$$y = c_0 + \sum_{n=1}^{p} c_n g(b_n x + a_n)$$



$$y = c_0 + c_1 g(b_1 x + a_1) + c_2 g(b_2 x + a_2) \Rightarrow \begin{cases} \text{a : deslocamento no eixo x} \\ \text{b : inclinação da sigmóide} \\ \text{c : amplitude da sigmóide} \end{cases}$$

Exemplo: Forma "construtiva" de aproximação de um mapeamento não-linear empregando neurônios com função de ativação do tipo tangente hiperbólica. Exemplo considerando um único estímulo de entrada.



$$f(\mathbf{w}) = \underbrace{c_1 g(b_1 x + a_1)}_{a} + \underbrace{c_2 g(b_2 x + a_2)}_{b} + \underbrace{c_3 g(b_3 x + a_3)}_{c} + \underbrace{c_4 g(b_4 x + a_4)}_{d} + \underbrace{c_5 g(b_5 x + a_5)}_{e} + \underbrace{c_0}_{bias}$$

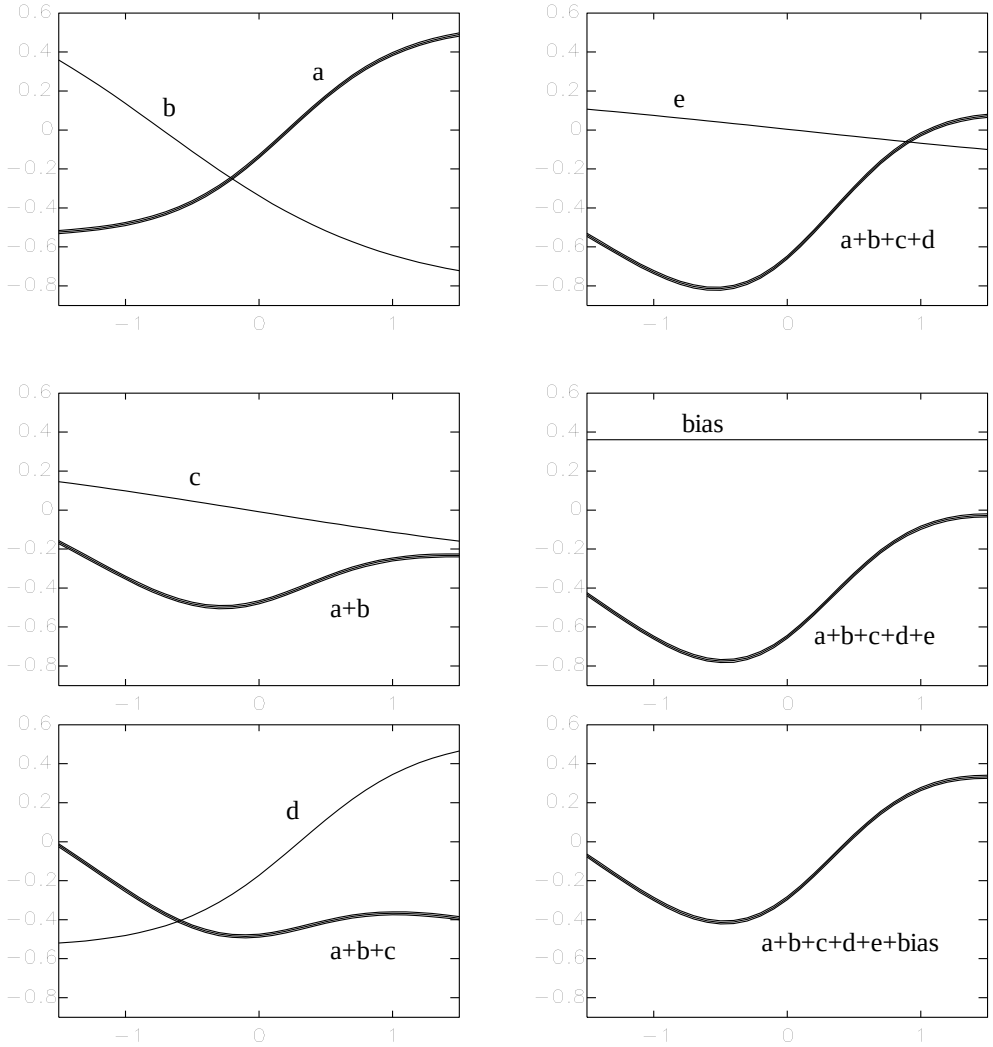


Figura 14 – Composição aditiva de ativações na reprodução de um mapeamento não-linear

O mapeamento a ser aproximado encontra-se na última figura à direita.

16. Superfície de erro

- Seja X uma região compacta do $\Re^m$ e seja g: $X \subset \Re^m \rightarrow \Re$ a função a ser aproximada (formulação para uma única saída, $r = 1$);
- O conjunto de dados de aproximação $\{(\mathbf{x}_l, s_l) \in \Re^m \times \Re\}_{l=1}^N$ é gerado considerando-se que os vetores de entrada $\mathbf{x}_l$ estão distribuídos na região compacta $X \subset \Re^m$ de acordo com uma função densidade de probabilidade fixa d_P: $X \subset \Re^m \rightarrow [0,1]$ e que os vetores de saída s_l são produzidos pelo mapeamento definido pela função g na forma:

$$s_l = g(\mathbf{x}_l) + \varepsilon_l, \; l = 1,...,N,$$

  onde $\varepsilon_l \in \Re$ é uma variável aleatória de média zero e variância fixa.
- A função g que associa a cada vetor de entrada $\mathbf{x} \in X$ uma saída escalar $s \in \Re$ pode ser aproximada com base no conjunto de dados de aproximação $\{(\mathbf{x}_l, s_l) \in \Re^m \times \Re\}_{l=1}^N$ por uma composição aditiva de funções de expansão ortogonal na forma:

$$\hat{s}_l = \hat{g}(\mathbf{x}_l, \theta) = \sum_{j=0}^{n} w_j f\left(\sum_{i=0}^{m} v_{ji} x_{li}\right) = \sum_{j=0}^{n} w_j f\left(\mathbf{v}_j^T \mathbf{x}_l\right)$$

onde θ é o vetor contendo todos os pesos da rede neural.

- Logo, o erro quadrático médio produzido na saída da rede neural, considerando as N amostras, assume a forma:

$$J(\theta) = \frac{1}{N}\sum_{l=1}^{N}(\hat{s}_l - s_l)^2 = \frac{1}{N}\sum_{l=1}^{N}(\hat{g}(\mathbf{x}_l, \theta) - s_l)^2 =$$

$$= \frac{1}{N}\sum_{l=1}^{N}\left(\sum_{j=0}^{n} w_j f\left(\sum_{i=0}^{m} v_{ji} x_{li}\right) - s_l\right)^2 = \frac{1}{N}\sum_{l=1}^{N}\left(\sum_{j=0}^{n} w_j f\left(\mathbf{v}_j^T \mathbf{x}_l\right) - s_l\right)^2$$

- Sendo P a dimensão do vetor θ, então tem-se que: $J : \Re^P \rightarrow \Re^1$.
- A superfície de erro definida por $J(\theta)$ reside no espaço $\Re^{P+1}$, sendo que se deve buscar em $\Re^P$ um ponto que minimiza $J(\theta)$, supondo que se queira minimizar o erro entre a saída produzida pelo rede neural e a saída desejada.

17. Aprendizado a partir de dados amostrados

- O aprendizado supervisionado visto como um problema de otimização não-linear
- A função-objetivo (critério de desempenho a ser otimizado) e os parâmetros ajustáveis:

$$\min_{\theta} J(\theta)$$

- Formalização matemática do que se quer otimizar + método de solução
- [Solução na forma fechada] × [Busca iterativa]
- Os dados de entrada/saída e a questão dos 3 mapeamentos envolvidos no processo:

    1. O mapeamento a ser aproximado (do qual se conhece apenas dados amostrados);
    2. O mapeamento resultante do processo de aproximação;
    3. O mapeamento entre cada vetor de pesos e o erro: superfície de erro.

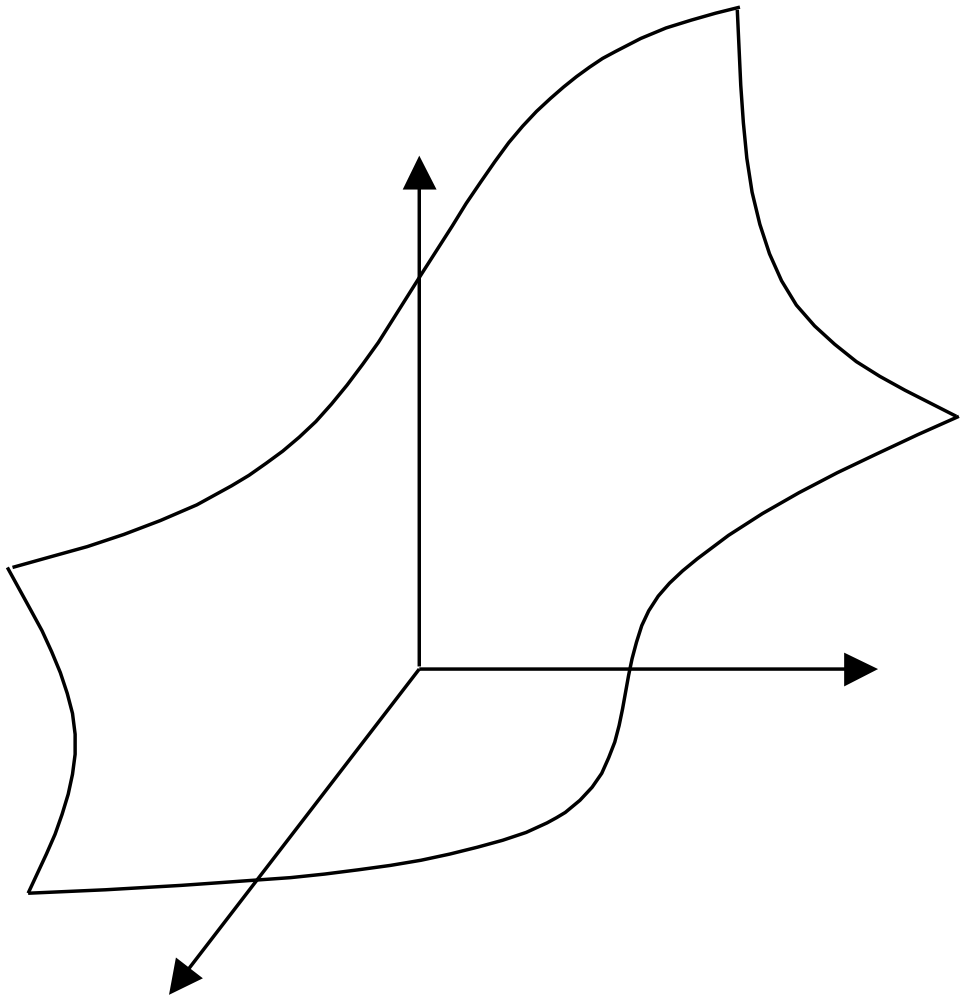


Figura 15– Mapeamento desconhecido a ser aproximado

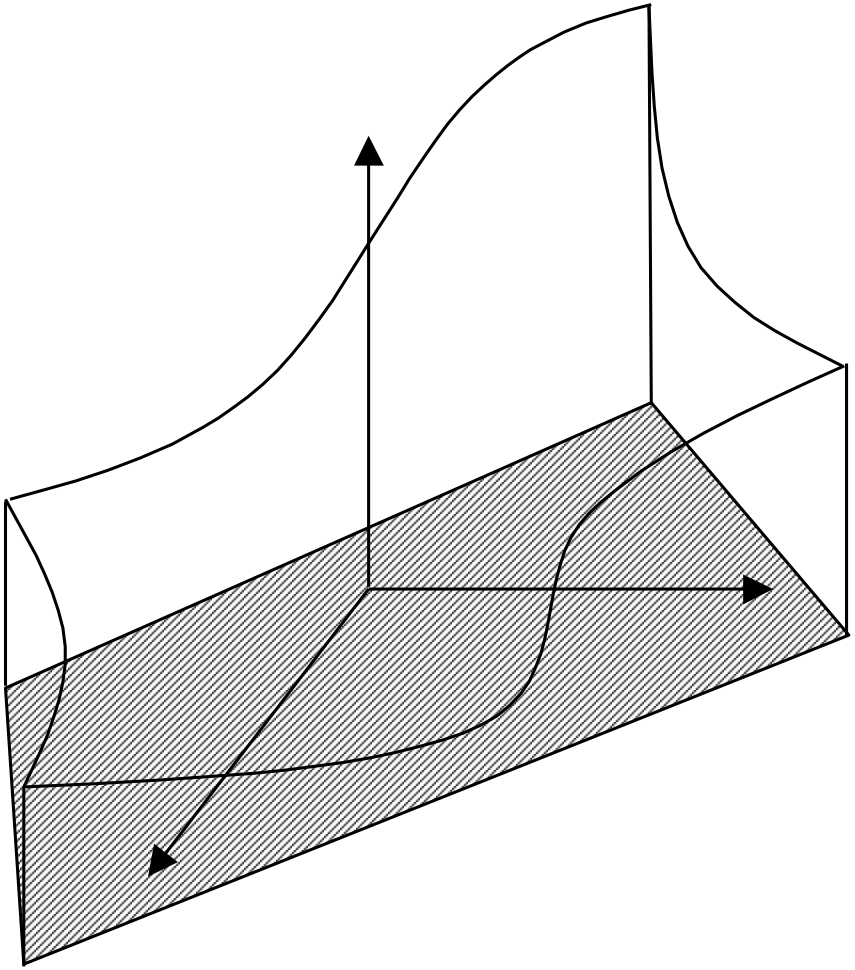


Figura 16 – Exemplo de região de operação. É uma região compacta (fechada e limitada).

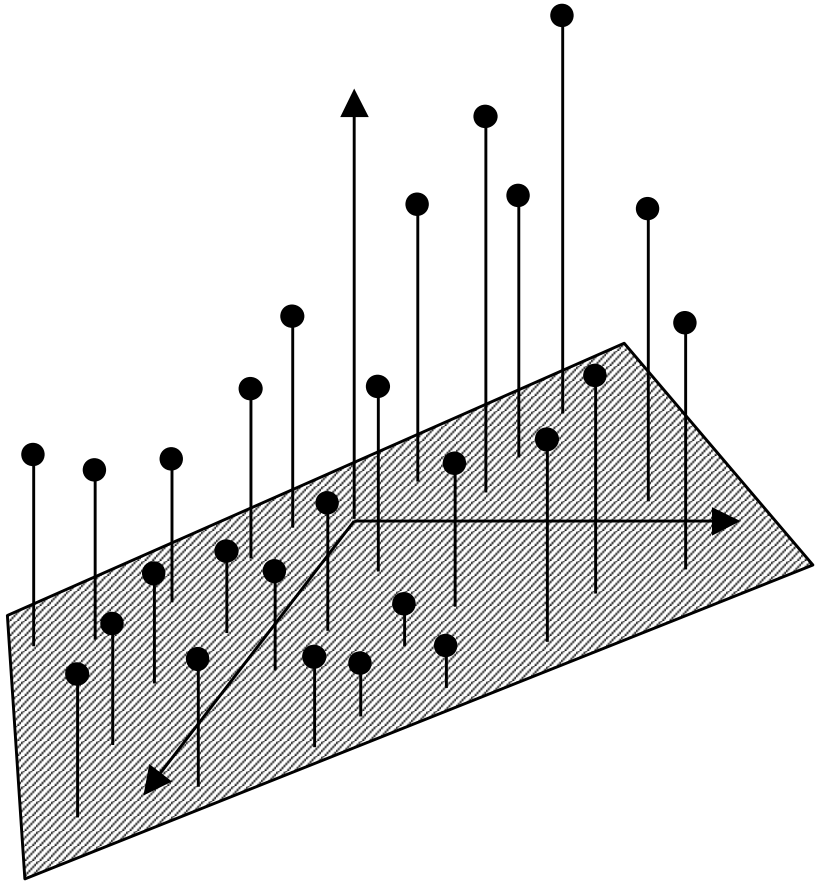


Figura 17 – Amostras expressando o comportamento da função para pontos específicos da região de operação. Essas amostras comporão os conjuntos de treinamento e validação (sendo que os dois conjuntos são independentes entre si)

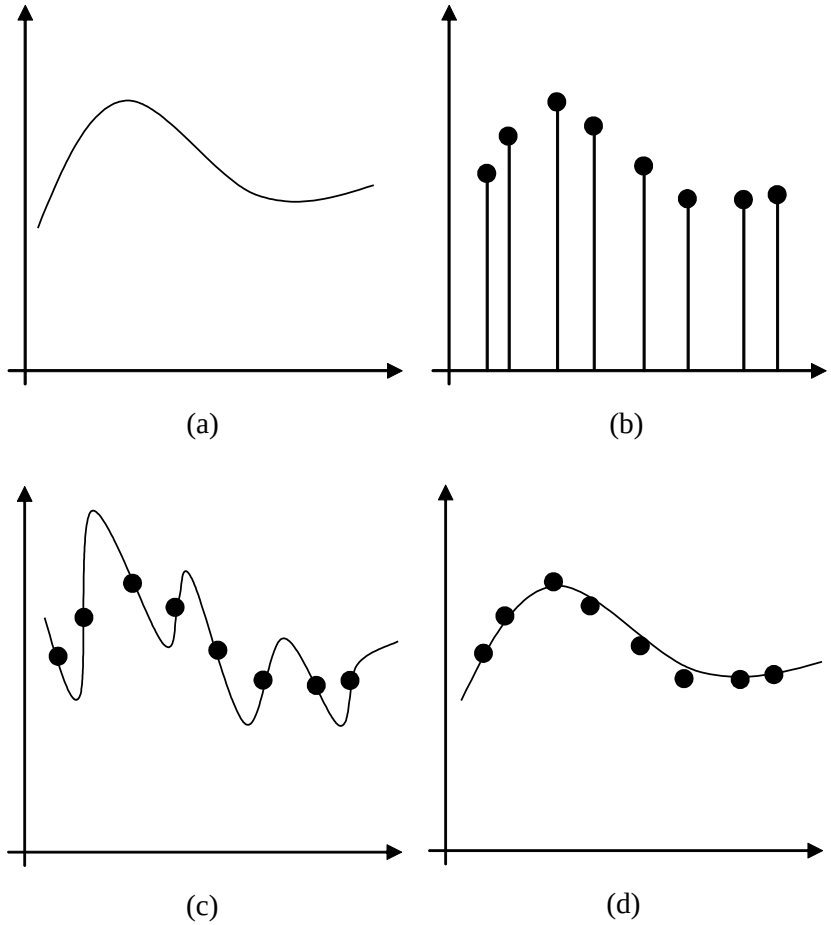


Figura 18 – (a) Função a ser aproximada (agora considerando apenas uma entrada); (b) Amostras disponíveis; (c) Resultado de um processo de aproximação com sobretreinamento; (d) Resultado de um processo de aproximação sem sobretreinamento.

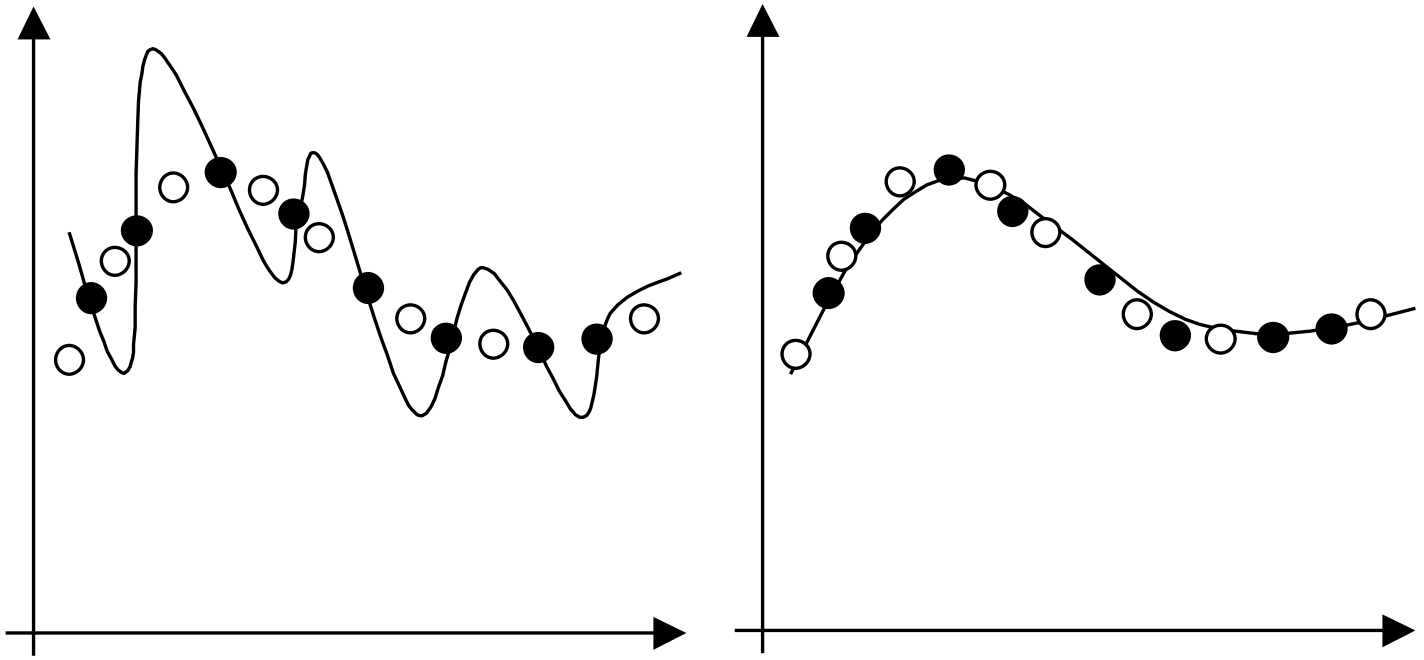


Figura 19 – Comparação de desempenho para dados de treinamento (pontos em preto) e validação (pontos em branco), de modo a medir a capacidade de generalização dos mapeamentos produzidos.

- O mapeamento da esquerda apresenta um erro de treinamento muito baixo, mas um erro de validação bastante elevado, quando comparado ao mapeamento da direita.

18. O problema do OU-exclusivo em MLP

- Considere os pontos (0,0),(0,1),(1,0) e (1,1) no plano $\Re^2$, conforme apresentado na Figura 20. O objetivo é determinar uma rede com duas entradas $\mathbf{x}_i \in \{0,1\}$ (i=1,2), e uma saída $\mathbf{y} \in \{0,1\}$ de maneira que: $\begin{cases} (\mathbf{x_1}, \mathbf{x_2}) = (0,0) \text{ ou } (1,1) \Rightarrow \mathbf{y} = 0 \\ (\mathbf{x_1}, \mathbf{x_2}) = (1,0) \text{ ou } (0,1) \Rightarrow \mathbf{y} = 1 \end{cases}$

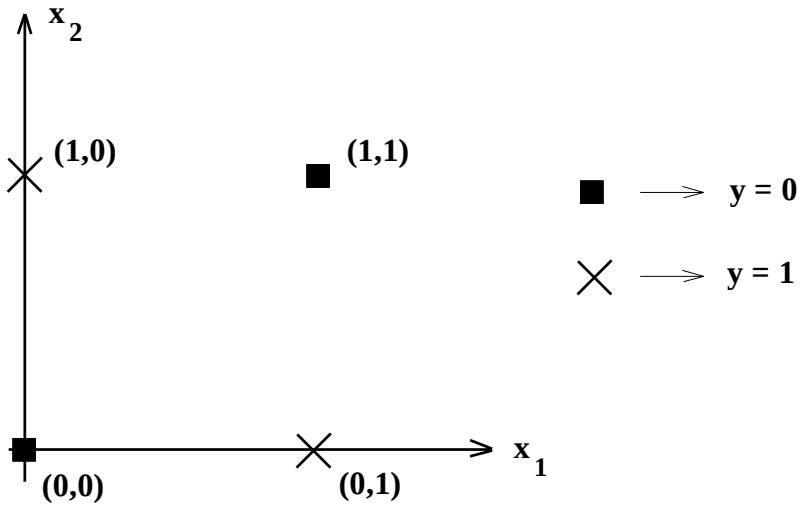


Figura 20 – O problema do OU-exclusivo

- Inicialmente, será analisado o comportamento de um neurônio tipo perceptron (veja Figura 21) no processo de solução do problema exposto acima. A saída **y** pode ser representada na forma:

$$\mathbf{y} = \mathbf{g}(\mathbf{w_1 x_1} + \mathbf{w_2 x_2} + \mathbf{w_0}) \quad \text{onde} \quad \begin{cases} g(\mathbf{u}) = 1 \ \text{se} \ \mathbf{u} \geq 0 \\ g(\mathbf{u}) = 0 \ \text{se} \ \mathbf{u} < 0 \end{cases}$$

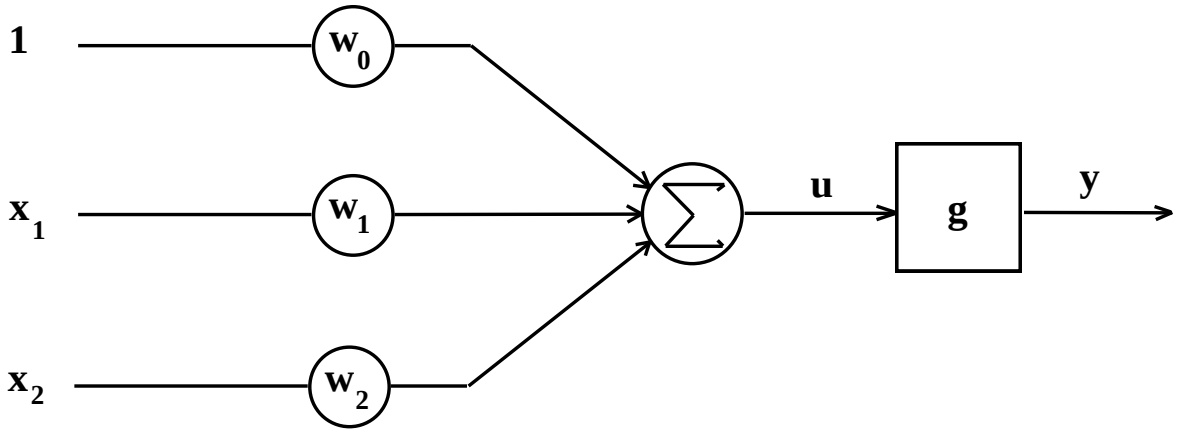


Figura 21 – Neurônio tipo perceptron, com duas entradas (mais a polarização)

- Para qualquer valor dos parâmetros $\mathbf{w_0}$, $\mathbf{w_1}$ e $\mathbf{w_2}$, a função **g**(**u**) separa o espaço de entradas em duas regiões, sendo que a curva de separação é uma linha reta.

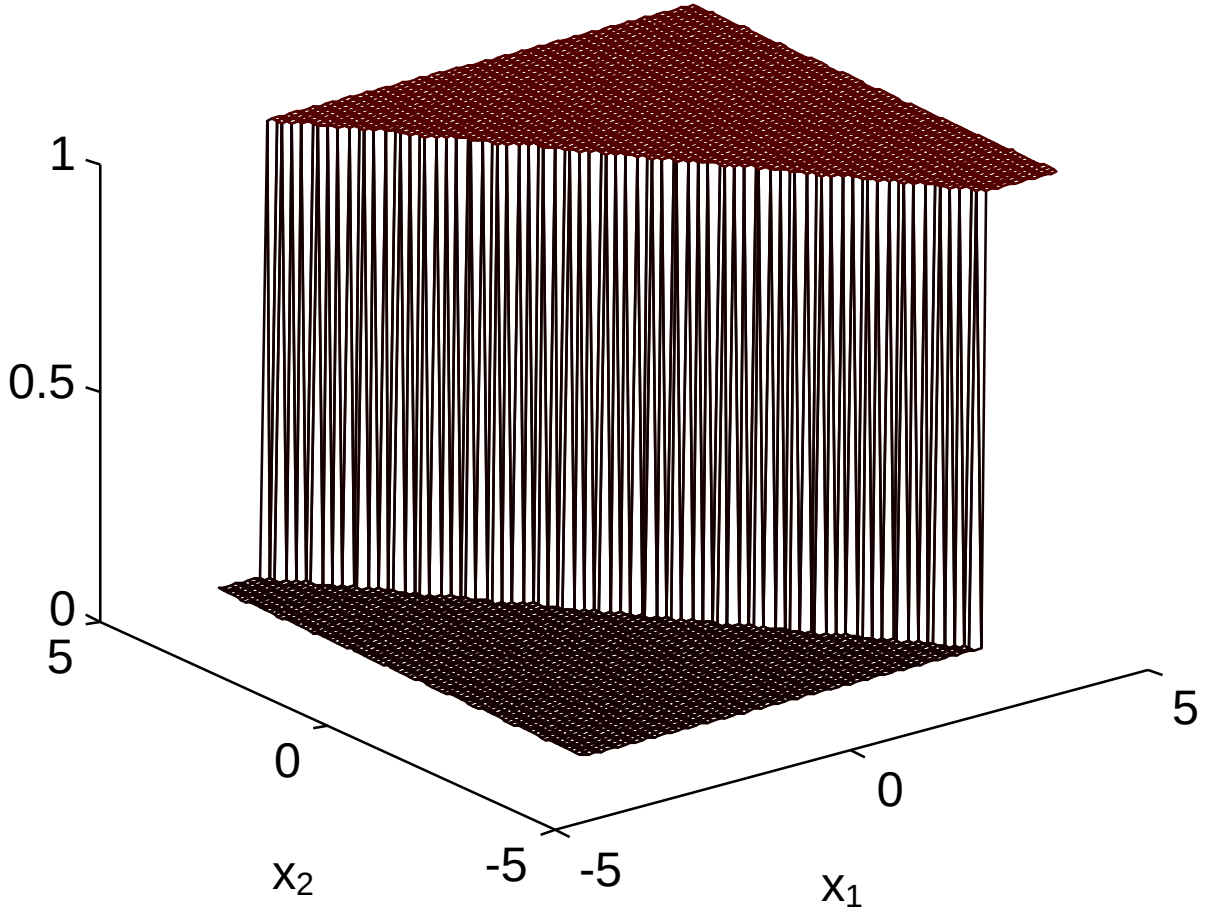


Figura 22 – Mapeamento de entrada-saída para o perceptron da Figura 21, com $\mathbf{w_0} = -6$, $\mathbf{w_1} = 4$ e $\mathbf{w_2} = 3$

- Aqui tomou-se a função g(·) como sendo a função sinal, pois as saídas são binárias.

- No problema do OU-exclusivo (Figura 20), pode-se constatar que não existe uma única linha reta divisória de forma que os pontos (0,0) e (1,1) se posicionem de um lado enquanto que (0,1) e (1,0) permaneçam do outro lado da linha.
- Logo, pode-se imediatamente concluir que um neurônio tipo perceptron não apresenta grau de liberdade suficiente para resolver o problema proposto, o que foi corretamente constatado por Minsky & Papert, em 1969.
- No entanto, esses autores também acreditavam que não havia razão para supor que redes multicamadas pudessem conduzir a uma solução para o problema proposto. Esta hipótese só foi definitivamente rejeitada com o desenvolvimento do algoritmo de retro-propagação (*back-propagation*), já nos anos 80, o qual permite o ajuste automático de pesos para redes neurais multicamadas, arquitetura necessária para a realização de mapeamentos não-lineares.
- Considere o problema de mapeamento de uma rede neural tipo perceptron, com uma camada intermediária (Figura 23), aplicada ao problema do OU-exclusivo.

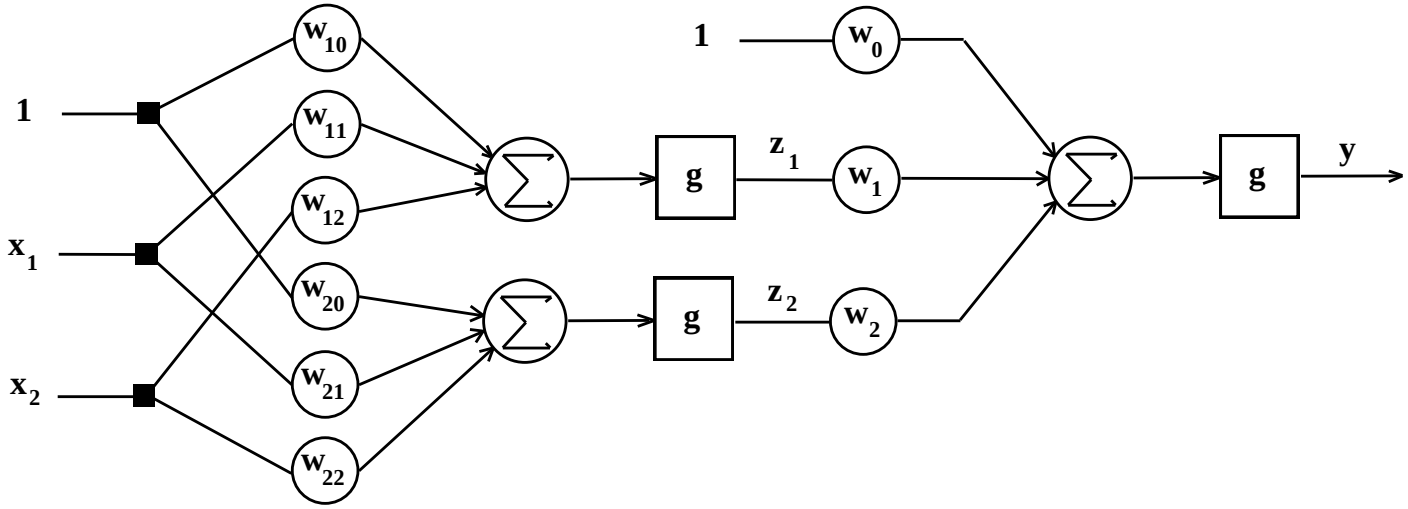


Figura 23 – Perceptron de três camadas (uma camada intermediária)

- A camada de entrada fornece um vetor de entrada ($\mathbf{x_1},\mathbf{x_2}$) para a camada intermediária, enquanto que a camada intermediária produz duas saídas $\mathbf{z_1}=\text{sgn}(\mathbf{w_{10}}+\mathbf{w_{11}x_1}+\mathbf{w_{12}x_2})$ e $\mathbf{z_2}=\text{sgn}(\mathbf{w_{20}}+\mathbf{w_{21}x_1}+\mathbf{w_{22}x_2})$. Na camada de saída, o sinal de saída da rede neural é dado por $\mathbf{y}=\text{sgn}(\mathbf{w_0}+\mathbf{w_1z_1}+\mathbf{w_2z_2})$.
- Surge uma questão: Existem parâmetros $\mathbf{w_{ij}}$ (i=1,2; j=0,1,2) e $\mathbf{w_k}$ (k = 0,1,2) tais que $\mathbf{y}$ = 0 para as entradas (0,0) e (1,1) e $\mathbf{y}$ = 1 para as entradas (1,0) e (0,1)?

- As saídas da primeira camada ($\mathbf{z_1}$ e $\mathbf{z_2}$) podem ser consideradas como variáveis intermediárias utilizadas na geração da saída **y**, sendo que o espaço gerado pelas variáveis $\mathbf{z_1}$ e $\mathbf{z_2}$ é chamado de espaço de características (*feature space*).
- Do que já foi visto a respeito de um neurônio tipo perceptron, sabe-se que existem pesos $\mathbf{w_{1j}}$ (j=0,1,2) tais que (veja curva de separação $\mathbf{L_1}$ na Figura 24(a)):

(0,1) produza $\mathbf{z_1} = 1$

(0,0),(1,0),(1,1) produza $\mathbf{z_1} = 0$.

- De forma similar, existem pesos $\mathbf{w_{2j}}$ (j=0,1,2) tais que (veja curva de separação $\mathbf{L_2}$ na Figura 24(a)):

(0,1),(0,0),(1,1) produza $\mathbf{z_2} = 1$

(1,0) produza $\mathbf{z_2} = 0$

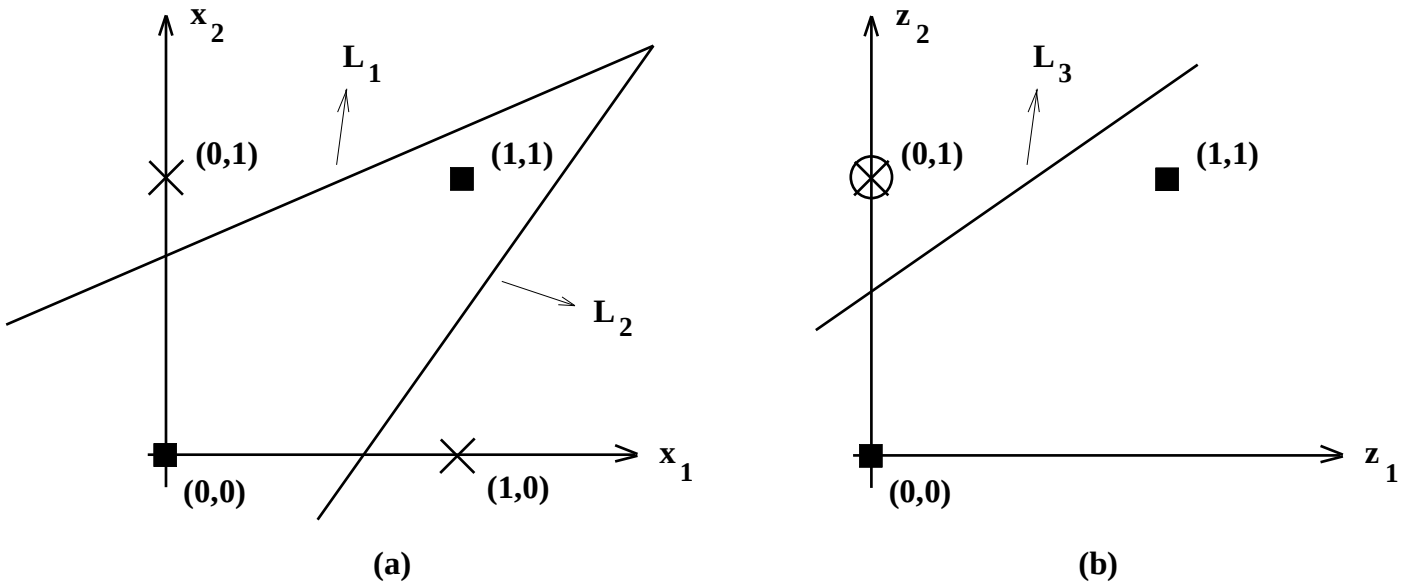


Figura 24 – Realização da função OU-exclusivo

- A discussão acima mostra que existem pesos $\mathbf{w_{ij}}$ (i=1,2; j=0,1,2) de maneira que a entrada (0,1) resulte em $\mathbf{z_1} = 1$, $\mathbf{z_2} = 1$, e a entrada (1,0) resulte em $\mathbf{z_1} = 0$, $\mathbf{z_2} = 0$, enquanto que (0,0) e (1,1) produzam $\mathbf{z_1} = 0$, $\mathbf{z_2} = 1$. Já que (0,0) e (1,1) podem ser separados linearmente de (0,1), como mostrado na Figura 24(b) pela curva de separação $\mathbf{L_3}$, pode-se concluir que a função booleana desejada pode ser obtida utilizando-se perceptrons em cascata, ou seja, 3 neurônios do tipo perceptron.

19. Otimização não-linear e capacidade de generalização

- Diversos tipos de parâmetros da rede neural poderiam ser submetidos a processos de ajuste durante o treinamento, como (*i*) pesos sinápticos; (*ii*) parâmetros da função de ativação de cada neurônio; (*iii*) número de neurônios na camada intermediária; (*iv*) número de camadas intermediárias.
- Iremos nos restringir aqui ao ajuste dos pesos sinápticos. Neste caso, o processo de treinamento supervisionado de redes neurais artificiais multicamadas é equivalente a um problema de otimização não-linear irrestrita, em que a superfície de erro é minimizada a partir do ajuste dos pesos sinápticos.
- Iremos nos restringir também a perceptrons com uma única camada intermediária, visto que com apenas uma camada intermediária a rede neural já apresenta **capacidade de aproximação universal** (CYBENKO, 1989; HORNIK *et al.*, 1989; HORNIK *et al.*, 1990; HORNIK *et al.*, 1994).

- Um problema comum a todos os modelos de aproximação de funções que possuem capacidade de aproximação universal, não apenas redes neurais artificiais do tipo MLP, é a necessidade de controlar adequadamente o seu grau de flexibilidade.
- Como o conjunto de amostras disponível para treinamento supervisionado é finito, infinitos mapeamentos podem produzir o mesmo desempenho de aproximação, independente do critério de desempenho adotado. Esses mapeamentos alternativos vão diferir justamente onde não há amostras disponíveis para diferenciá-los.
- Visando maximizar a **capacidade de generalização** do modelo de aproximação (no caso, uma rede neural MLP), ou seja, buscando encontrar o grau de flexibilidade adequado para o modelo de aproximação (dada a demanda da aplicação), um procedimento recomendado é dividir o conjunto de amostras disponível para treinamento em dois: um conjunto que será efetivamente empregado no ajuste dos pesos (conjunto de treinamento) e um conjunto que será empregado para definir o momento de interromper o treinamento (conjunto de validação).

- Deve-se assegurar que ambos os conjuntos sejam suficientemente representativos do mapeamento que se pretende aproximar. Assim, minimizar o erro junto ao conjunto de validação implica em maximizar a capacidade de generalização. Logo, espera-se que a rede neural que minimiza o erro junto ao conjunto de validação (não usado para o ajuste dos pesos) tenha o melhor desempenho possível junto a novas amostras.
- A figura alto/esquerda a seguir mostra um mapeamento unidimensional a ser aproximado (desconhecido pela rede neural) e amostras sujeitas a ruído de média zero (única informação disponível para o treinamento da rede neural). A figura alto/direita mostra o resultado da aproximação produzida por uma rede neural com muito poucos neurônios, a qual foi incapaz de realizar a aproximação (tem baixa flexibilidade).
- Já as figuras baixo/esquerda e baixo/direita mostram o resultado de uma mesma rede neural (com número suficiente de neurônios), mas à esquerda ocorreu sobre-treinamento, enquanto que à direita o treinamento foi interrompido quando minimizou-se o erro junto a dados de validação (não apresentados).

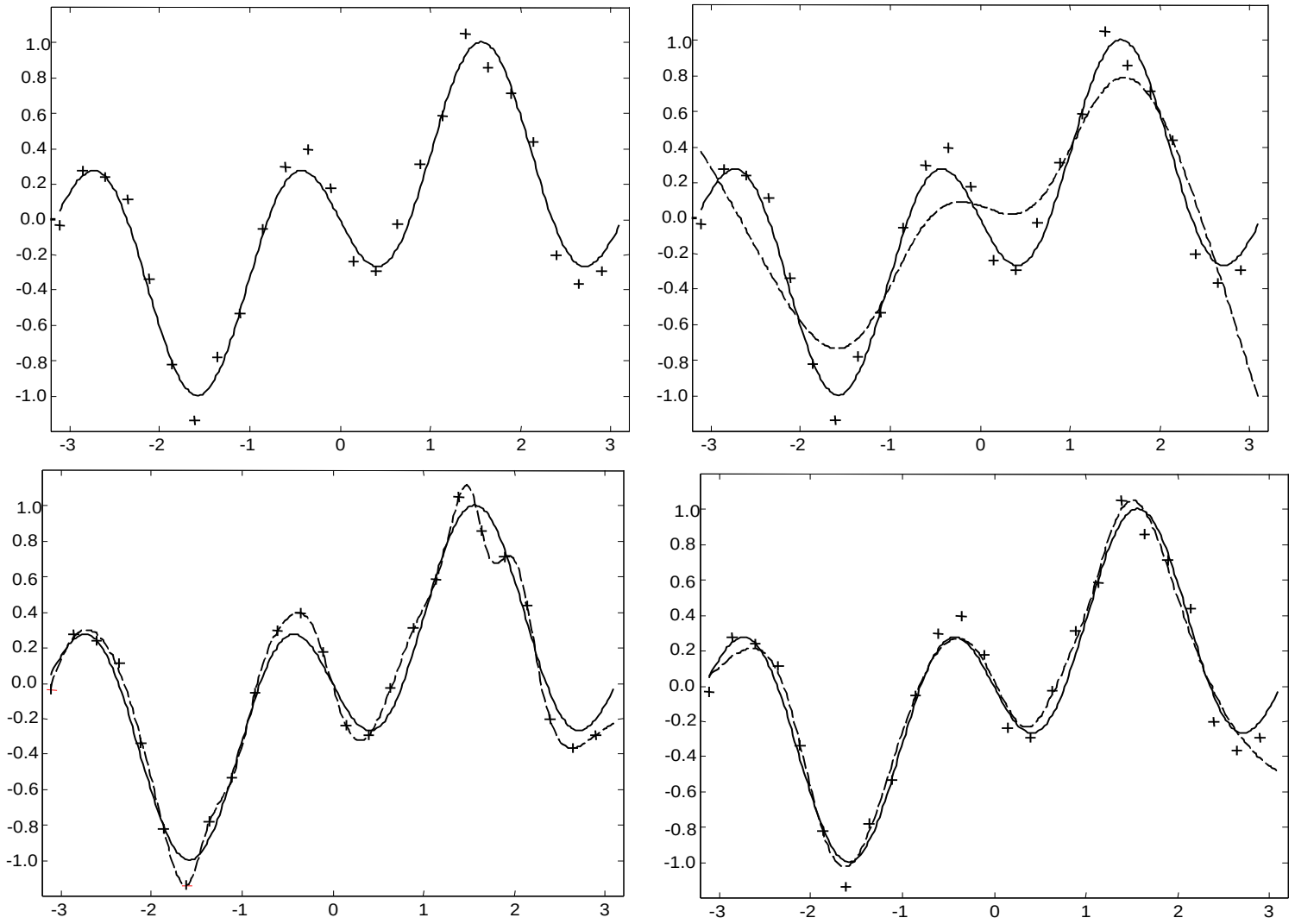


Figura 25 – Dados amostrados, função a ser aproximada e modelos de aproximação com diferentes capacidades de generalização

- Curvas típicas de erro de treinamento e validação são apresentadas a seguir.

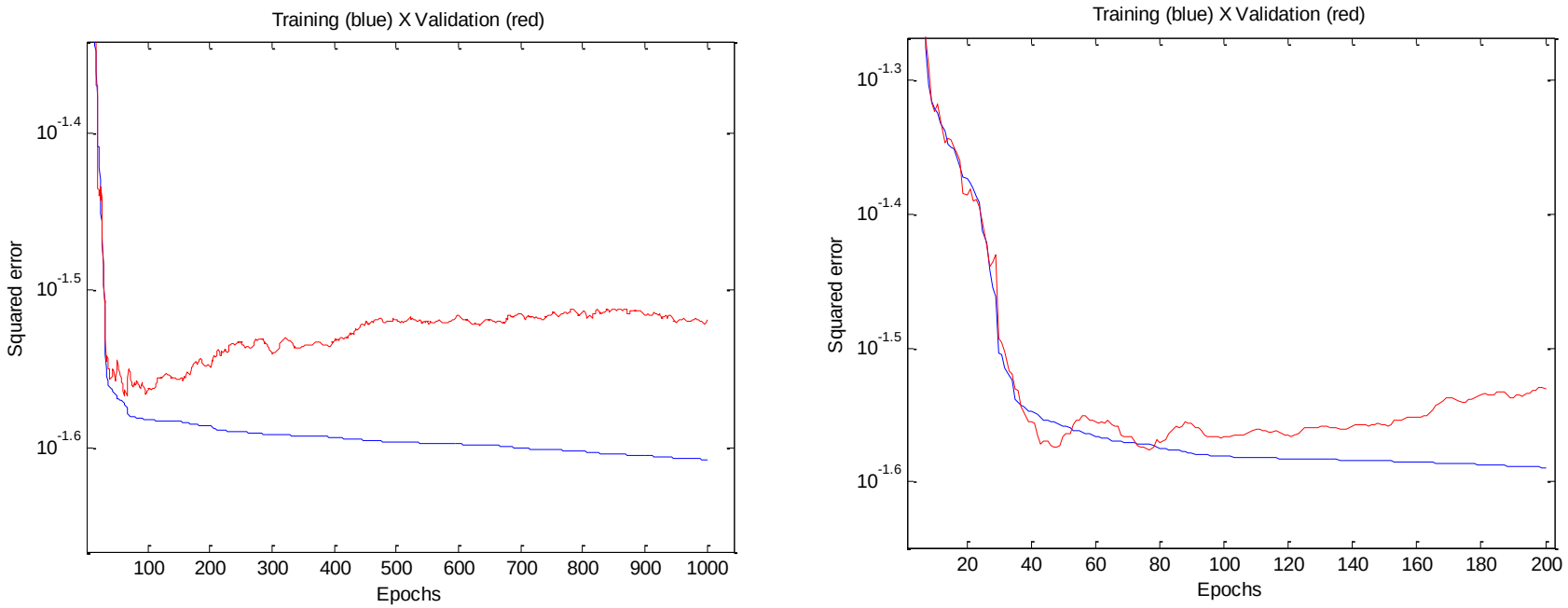


Figura 26 – Ilustrações típicas da evolução, ao longo das épocas de treinamento, dos erros de treinamento e de validação em treinamento supervisionado de redes MLP.

- No entanto, nem sempre o erro de validação apresenta este comportamento, e cada caso deve ser analisado isoladamente.
- Este mecanismo de regularização do treinamento de redes neurais é denominado *holdout*, por manter de fora do treinamento um subconjunto das amostras.

- Como a curva do erro de validação oscila bastante e esboça um comportamento pouco previsível, não é indicado desenvolver detectores automáticos de mínimos e encerrar o treinamento ali. O mais indicado é sobretreinar a rede e armazenar os pesos associados ao mínimo do erro de validação.
- Não existe um consenso sobre como fazer o melhor particionamento do conjunto de dados, ou seja, sobre como dividi-lo de forma que possamos encontrar uma rede com a melhor capacidade de generalização em todos os casos. Uma sugestão de partida pode ser 80% das amostras para treinamento e 20% para validação.
- Quando as amostras correspondem a dados rotulados em problemas de classificação de padrões, procure respeitar a distribuição junto a cada classe.
- Quando a quantidade de dados é elevada, é comum também realizar uma terceira partição, chamada de conjunto de teste. Neste caso, o objetivo do treinamento continua sendo a minimização do erro junto ao conjunto de validação, mas o desempenho final da rede neural é medido junto ao conjunto de teste.

19.1 Validação cruzada com *k* pastas

- A técnica *holdout* que acabou de ser descrita tem uma desvantagem importante: alguns dados disponíveis sempre serão usados no ajuste dos pesos e outros nunca serão usados para este fim, pois compõem o conjunto de validação.
- Uma técnica mais elaborada é dividir o conjunto de amostras disponíveis para treinamento em k pastas e realizar k treinamentos, cada um considerando k–1 pastas para o ajuste de pesos (equivale ao conjunto de treinamento da seção anterior) e 1 pasta para a validação (equivale ao conjunto de validação da seção anterior).
- Com este procedimento, toda amostra disponível vai aparecer k–1 vezes no conjunto de treinamento e 1 vez no conjunto de validação.
- Repare que os k conjuntos de treinamento terão composição distinta, assim como os k conjuntos de validação. O desempenho da rede neural é geralmente tomado como a média do desempenho das k redes neurais obtidas.

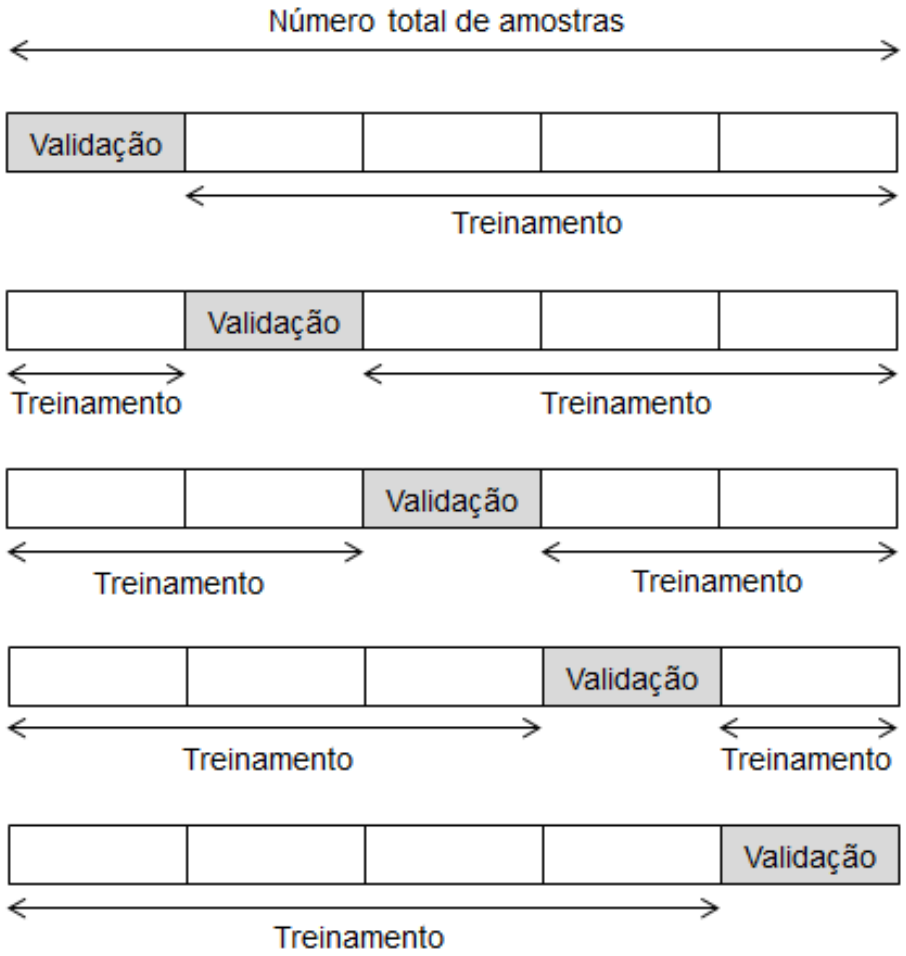


Figura 27 – Esquema de divisão das amostras disponíveis para treinamento na abordagem de validação cruzada com *k*-pastas (*k-fold crossvalidation*), com *k*=5.

19.2 Rede neural MLP como um modelo instável

- Como todos os modelos que apresentam capacidade de aproximação universal, a rede neural MLP é considerada um modelo instável, no sentido de que a proposta de mapeamento de entrada-saída pode sofrer mudanças não-esperadas com a simples inclusão de uma amostra adicional junto ao conjunto de dados de treinamento.
- Essa instabilidade simplesmente é o reflexo do grau de flexibilidade e do poder de aproximação e não necessariamente se apresenta como uma desvantagem em aplicações práticas. De fato, esta propriedade será fundamental para o sucesso da abordagem de comitê de máquinas denominada *ensemble*, a ser tratada mais adiante.
- As técnicas de regularização *holdout* e *k-fold crossvalidation* podem ser interpretadas, então, como uma forma de "robustecer" o resultado do processo de aproximação do mapeamento de entrada-saída a partir de modelos instáveis.

19.3 Gradiente, hessiana e algoritmos de otimização

- Considere uma função contínua e diferenciável até 2a. ordem em todos os pontos do domínio de interesse, tal que: f: $\Re^n \rightarrow \Re$ e $\mathbf{x} \in \Re^n$
- Expansão em série de Taylor em torno do ponto $\mathbf{x}^* \in \Re^n$:

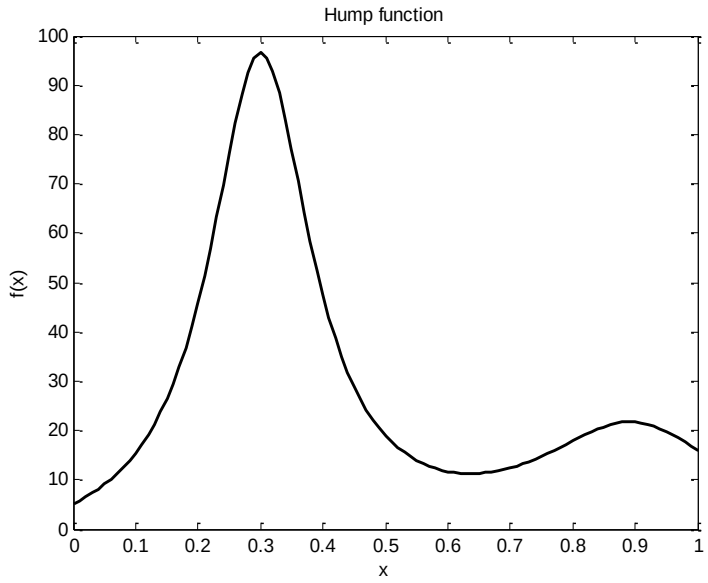
$$f(\mathbf{x}) = f(\mathbf{x}^*) + \nabla f(\mathbf{x}^*)^T (\mathbf{x} - \mathbf{x}^*) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T \nabla^2 f(\mathbf{x}^*)(\mathbf{x} - \mathbf{x}^*) + O(3)$$

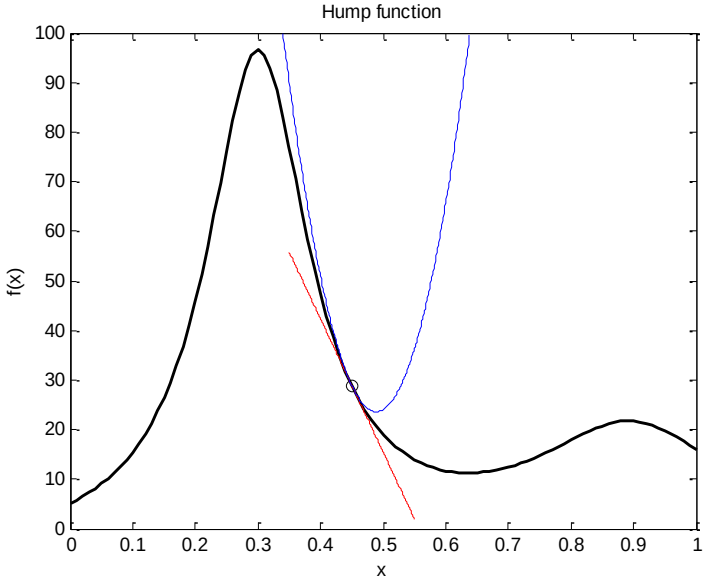
$$\nabla f(x^*) = \begin{bmatrix} \frac{\partial f(\mathbf{x}^*)}{\partial x_1} \\ \frac{\partial f(\mathbf{x}^*)}{\partial x_2} \\ \vdots \\ \frac{\partial f(\mathbf{x}^*)}{\partial x_n} \end{bmatrix} \qquad \nabla^2 f(\mathbf{x}^*) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_2^2} & & \vdots \\ & & \ddots & \\ \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_n \partial x_1} & \cdots & & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_n^2} \end{bmatrix}$$

Vetor gradiente **Matriz hessiana**

- O algoritmo de retropopagação do erro (do inglês *backpropagation*) é empregado para obter o vetor gradiente, onde cada elemento do vetor gradiente está associado a um peso da rede neural e indica o quanto a saída é influenciada por uma variação incremental neste peso sináptico.

- Função $y = f(x) = \frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6$, conhecida como *hump function*, e suas aproximações de 1a. e 2a. ordem no ponto $x = 0.45$.





- Existem várias técnicas para obter exatamente ou aproximadamente a informação de 2a. ordem junto à superfície de erro produzida por redes neurais MLP (BATTITI, 1992; BISHOP, 1992).

- O processo de otimização não-linear envolvido no ajuste de pesos de uma rede neural vai realizar aproximações locais de primeira ordem ou de primeira e segunda ordem junto à superfície de erro e realizar ajustes incrementais e recursivos na forma:

$$\theta_{k+1} = \theta_k + \text{passo}_k * \text{direção}_k$$

- Parte-se de uma condição inicial θ_0 e aplica-se iterativamente a fórmula acima, sendo que a direção depende da informação local de primeira e segunda ordem. Cada proposta de algoritmo de otimização vai diferir na forma de computar o *passo* e a *direção* de ajuste, a cada iteração *k*.

- A figura a seguir apresenta uma classificação dos principais algoritmos empregados para o treinamento supervisionado de redes neurais artificiais.

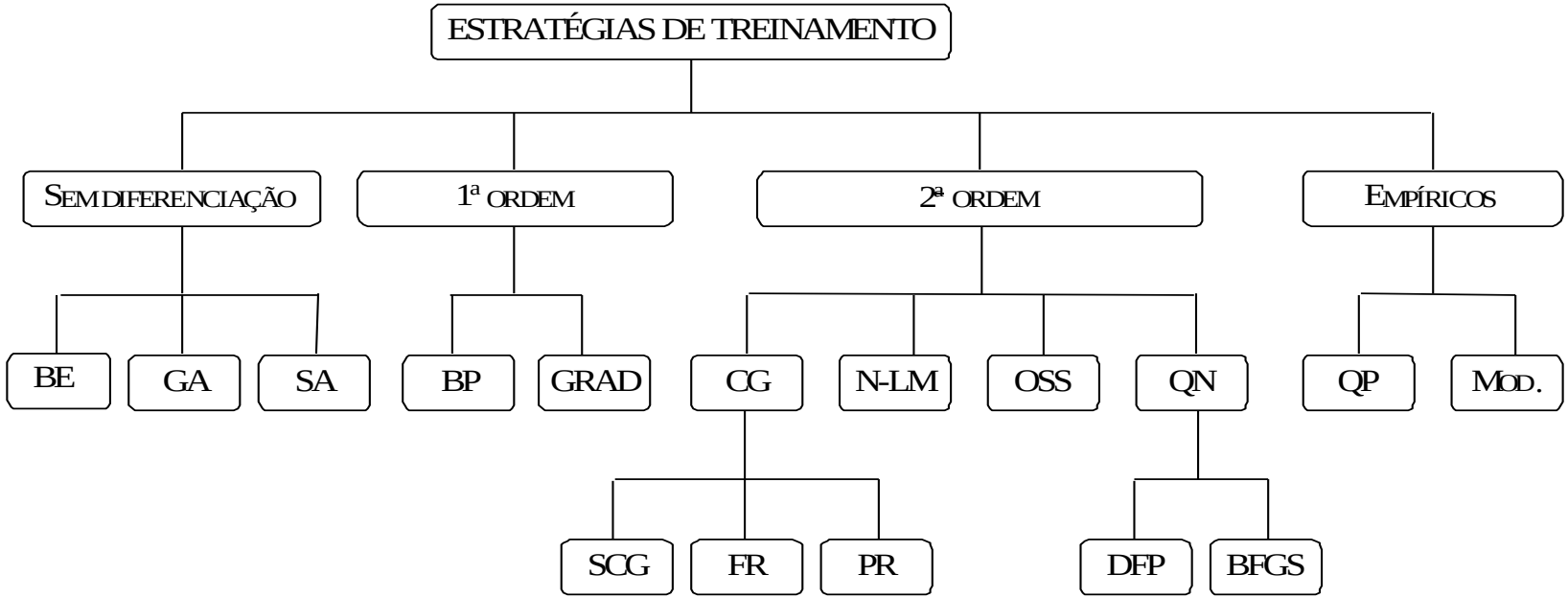


Figura 28 – Taxonomia de algoritmos de otimização para treinamento supervisionado de redes neurais MLP.

- Aquele que é utilizado no toolbox fornecido pelo professor é o gradiente conjungado escalonado (do inglês *Scaled Conjugate Gradient* – SCG). Uma vantagem deste algoritmo é que ele apresenta um custo computacional (memória e processamento por iteração) linear com o número de pesos e não quadrático, como a maioria dos algoritmos de 2a. ordem.

19.4 Mínimos locais

- Como o processo de ajuste é iterativo e baseado apenas em informações locais, os algoritmos de otimização geralmente convergem para o mínimo local mais próximo de onde se encontra a busca, que pode representar uma solução inadequada (com nível de erro acima do aceitável).

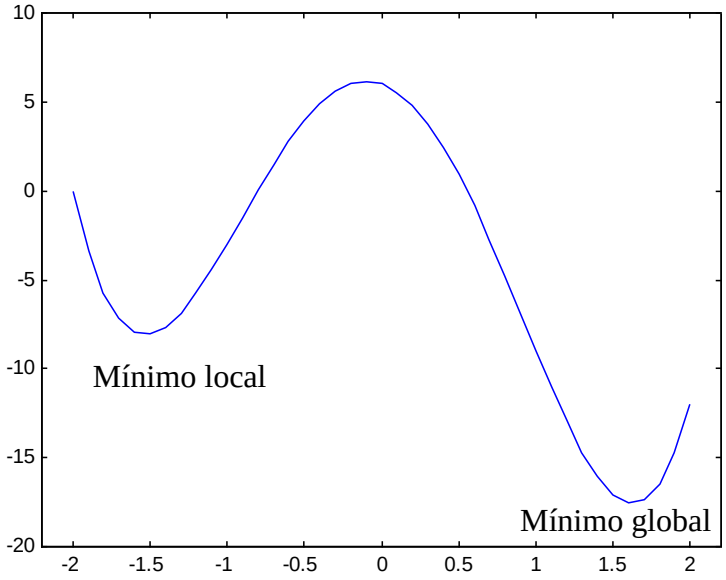


Figura 29 – Exemplo ilustrativo de mínimos local e global (considerando uma única variável).

- Note que algoritmos de 2a. ordem tendem a convergir mais rápido para os mínimos locais, mas não se pode afirmar que eles convergem para mínimos de melhor qualidade que aqueles produzidos pelos algoritmos de primeira ordem.
- Um conceito relevante aqui é o de **bacia de atração**. Todo mínimo local tem uma bacia de atração e o mínimo local a ser fornecido pelo algoritmo de otimização tende a ser aquele em cuja bacia de atração encontra-se a condição inicial (ponto de partida da busca iterativa).
- Isso é bem provável no caso de buscas iterativas que dão passos sempre minimizantes, mas podem ocorrer passos capazes de deslocar a busca para bacias de atração vizinhas, associadas a outros mínimos locais.
- A trajetória da condição inicial até o ótimo local resultante será certamente distinta para algoritmos de 1a. e 2a. ordem, pois eles diferem a cada iteração da busca, mas o resultado final tende a ser o mesmo, a menos que haja deslocamentos casuais para outras bacias de atração vizinhas.

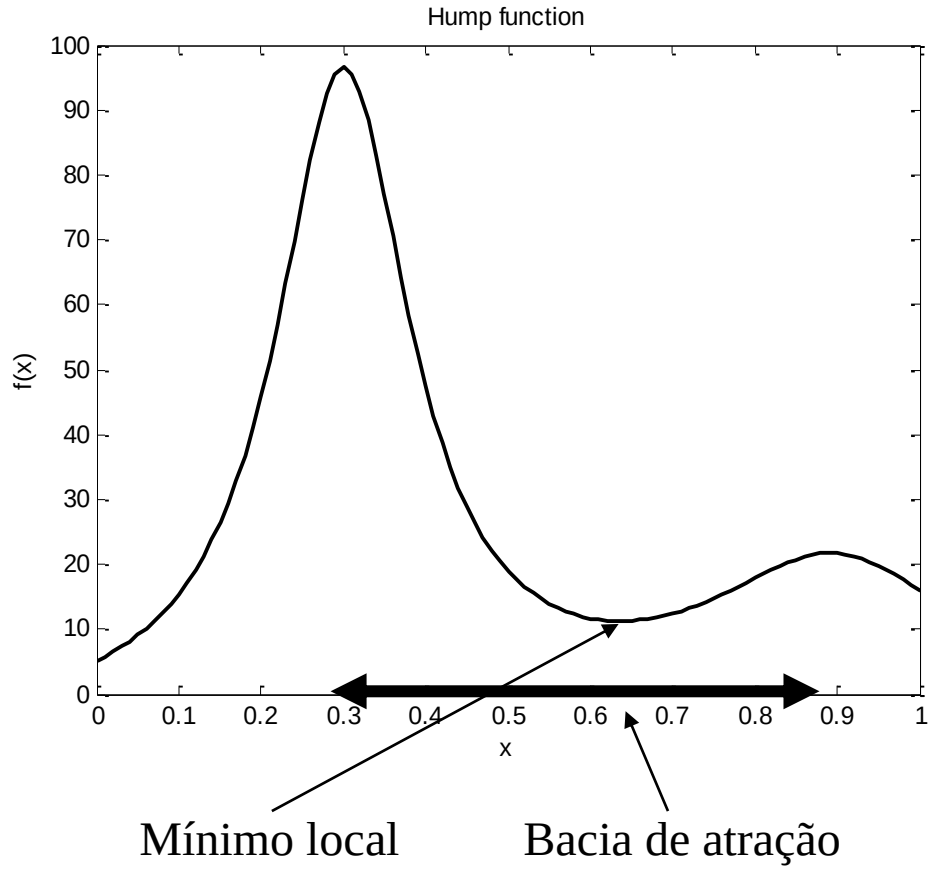


Figura 30 – Exemplo de bacia de atração de um mínimo local

19.5 Condição inicial para os pesos da rede neural

- Embora existam técnicas mais elaboradas, os pesos da rede neural podem ser inicializados com valores pequenos e aleatoriamente distribuídos em torno de zero.
- Esta inicialização promove as seguintes propriedades da rede neural inicial:
    - O mapeamento realizado pela MLP tende a se aproximar de um hiperplano, não apresentando, assim, nenhuma tendência definida, em termos de comportamento não-linear;
    - A ativação de todos os neurônios vai se encontrar fora da região de saturação, facilitando o processo de ajuste de pesos, a ser iniciado.
- Em termos práticos, pode-se afirmar que, com este procedimento de inicialização, o mapeamento produzido pela rede neural começa sem nenhuma contorção expressiva, mas com máxima capacidade de se contorcer de acordo com a demanda da aplicação.

19.6 Critério de parada

- O critério de parada mais empregado é o número máximo de épocas de treinamento, onde uma época é dada pela apresentação de todas as amostras do conjunto de treinamento à rede neural.
- Mas existem outros critérios que podem ser considerados conjuntamente ou em substituição ao número de épocas:
    - Módulo do vetor gradiente abaixo de um certo limiar;
    - Progresso do erro de treinamento abaixo de um certo limiar;
    - Grau de ajuste dos pesos sinápticos abaixo de um certo limiar.
- Esses limiares mencionados nos três itens acima devem ser definidos pelo usuário, havendo a necessidade de realização de alguns testes junto a cada aplicação pretendida.
- No toolbox fornecido pelo professor, empregam-se o número máximo de épocas conjuntamente com o módulo do vetor gradiente.

20. Processo Iterativo para MLP – Método Padrão-a-Padrão

Obs. 1: Está sendo considerada a aplicação de um método de 1a. ordem para ajuste dos pesos.

Obs. 2: Este método e variações dele são conhecidas como *stochastic gradient descent* (DUCHI et al., 2011).

- Defina uma condição inicial para o vetor de pesos $\mathbf{w}$ e escolha um passo α pequeno;
- Faça $k = 0$ e calcule $J(\mathbf{w}(k))$;
- Enquanto o critério de parada não for atendido, faça:
    - ♦ Ordene aleatoriamente os padrões de entrada-saída;
    - ♦ Para l variando de 1 até N, faça:
        Apresente o padrão l de entrada à rede;
        Calcule $J_l(\mathbf{w}(k))$ e $\nabla J_l(\mathbf{w}(k))$;
        $\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha \nabla J_l(\mathbf{w}(k))$;
    - ♦ $k = k + 1$;
    - ♦ Calcule $J(\mathbf{w}(k))$;

21. Processo Iterativo para MLP – Método em Lote ou Batelada

Obs. 1: Está sendo considerada a aplicação de um método de 1a. ordem para ajuste dos pesos.

Obs. 2: Versões intermediárias entre padrão-a-padrão e batelada têm sido bastante empregadas, sendo denominadas de *mini-batch* (https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size/).

- Defina uma condição inicial para o vetor de pesos **w** e escolha um passo α pequeno;
- Faça $k = 0$ e calcule $J(\mathbf{w}(k))$;
- Enquanto o critério de parada não for atendido, faça:
  - ♦ Para l variando de 1 até N, faça:
    Apresente o padrão l de entrada à rede;
    Calcule $J_l(\mathbf{w}(k))$ e $\nabla J_l(\mathbf{w}(k))$;
  - ♦ $\mathbf{w}(k+1) = \mathbf{w}(k) - \frac{\alpha}{N}\sum_{l=1}^{N}\nabla J_l(\mathbf{w}(k))$;
  - ♦ $k = k + 1$;
  - ♦ Calcule $J(\mathbf{w}(k))$;

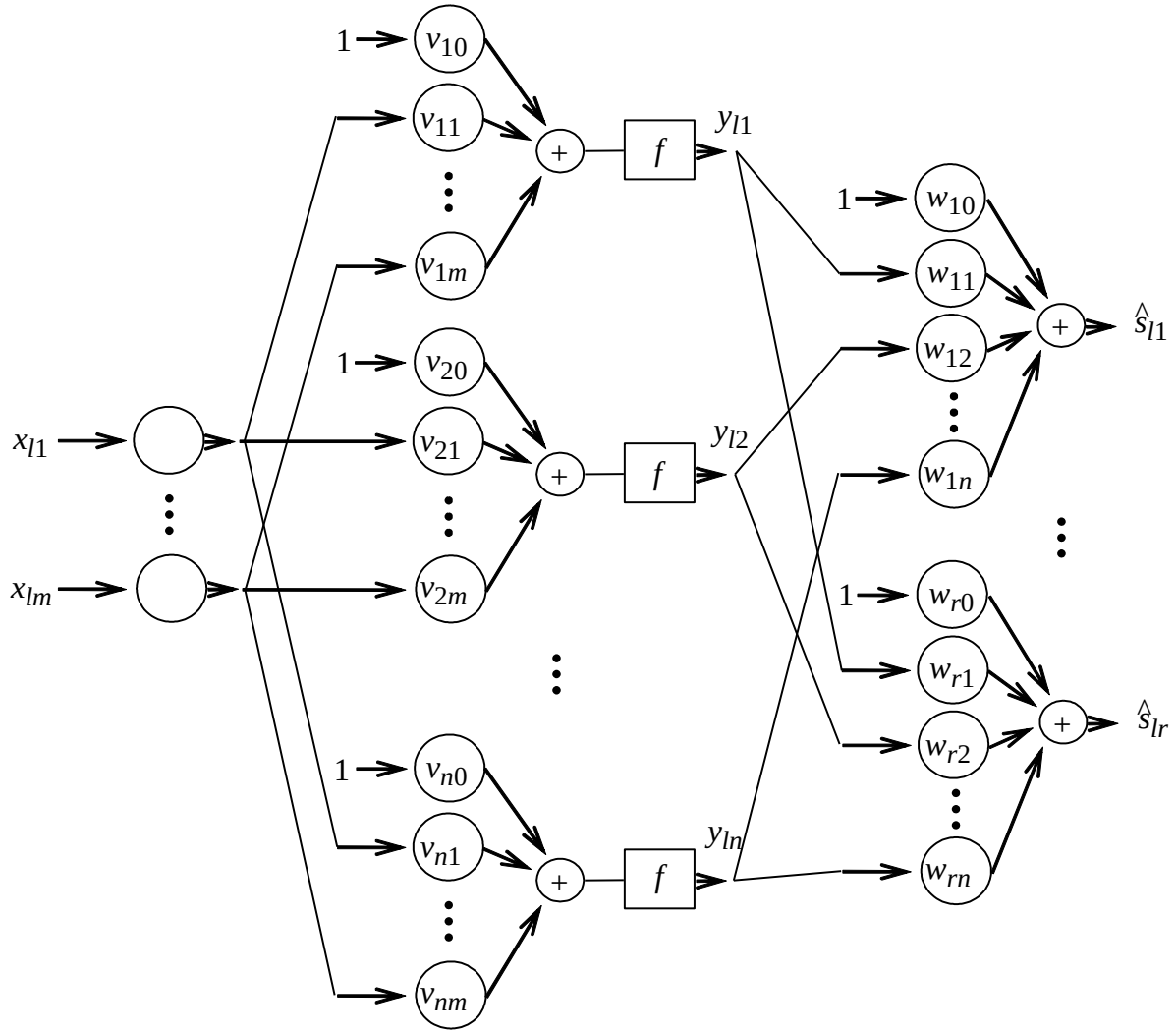
22. Vetor gradiente para redes MLP

- Nesta seção, são apresentados todos os passos até a obtenção das derivadas parciais da função $J(\theta)$ em relação aos pesos v_{ji} e w_{kj}, com *i*, *j* e *k* quaisquer, na rede neural MLP apresentada abaixo, a qual tem múltiplas saídas, num total de *r* saídas.
- O vetor θ contém todos os pesos sinápticos da rede neural, ordenados de forma arbitrária, mas fixa.
- A função-objetivo a ser minimizada é dada por:

$$J(\theta)=\frac{1}{2}\sum_{l=1}^{N}\sum_{k=1}^{r}\left(s_{lk}-\hat{s}_{lk}\right)^2=\frac{1}{2}\sum_{l=1}^{N}\sum_{k=1}^{r}\left\{s_{lk}-\left[\sum_{j=1}^{n}w_{kj}\,f\left(\sum_{i=0}^{m}v_{ji}x_{li}\right)+w_{k0}\right]\right\}^2$$

- Vamos obter $\frac{\partial J}{\partial v_{ji}}$, com j=1,…,n e i=0,…,m, e obter $\frac{\partial J}{\partial w_{kj}}$, com k=1,…,r e j=0,…,n.

  Está sendo considerado que x_{l0} = 1, l=1,...,N.


1
v_{10}
v_{11}
v_{1m}
y_{l1}
f
+
1
w_{10}
w_{11}
w_{12}
w_{1n}
$\hat{s}_{l1}$
x_{l1}
x_{lm}
1
v_{20}
v_{21}
v_{2m}
y_{l2}
1
w_{r0}
w_{r1}
w_{r2}
w_{rn}
$\hat{s}_{lr}$
1
v_{n0}
v_{n1}
v_{nm}
y_{ln}

Resolução:

Para evitar ambiguidade de sub-índices, os índices *i*, *j* e *k* de $J(\theta)$ serão renomeados, respectivamente, para *I*, *J* e *K*, produzindo:

$$J(\theta)=\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\left(s_{lK}-\hat{s}_{lK}\right)^2=\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\left\{s_{lK}-\left[\sum_{J=1}^{n}w_{KJ}\,f\left(\sum_{I=0}^{m}v_{JI}x_{lI}\right)+w_{K0}\right]\right\}^2$$

Três variáveis auxiliares serão consideradas:

$$e_{lK}=s_{lK}-\hat{s}_{lK}$$

$$u_{lJ}=\sum_{I=0}^{m}v_{JI}x_{lI}$$

$$y_{lJ}=f\left(u_{lJ}\right)$$

As propriedades de operadores de diferenciação que iremos aplicar são a regra da cadeia e equivalência entre a derivada da soma e a soma das derivadas.

Parte 1 da Solução:

$$\frac{\partial J(\theta)}{\partial v_{ji}} = \frac{\partial}{\partial v_{ji}}\left[\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\left(s_{lK}-\hat{s}_{lK}\right)^2\right] = \frac{\partial}{\partial v_{ji}}\left[\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}e_{lK}^2\right] = \frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\frac{\partial\left(e_{lK}^2\right)}{\partial v_{ji}} =$$

$$= \frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\frac{\partial\left(e_{lK}^2\right)}{\partial e_{lK}}\frac{\partial\left(e_{lK}\right)}{\partial v_{ji}} = \sum_{l=1}^{N}\sum_{K=1}^{r}e_{lK}\frac{\partial\left(e_{lK}\right)}{\partial \hat{s}_{lK}}\frac{\partial\left(\hat{s}_{lK}\right)}{\partial v_{ji}} = -\sum_{l=1}^{N}\sum_{K=1}^{r}e_{lK}\frac{\partial\left(\hat{s}_{lK}\right)}{\partial v_{ji}}$$

Sabendo que $\hat{s}_{lK} = \sum_{J=1}^{n} w_{KJ}\, f\left(\sum_{I=0}^{m} v_{JI}\, x_{lI}\right) + w_{K0}$, então tem-se:

$$\frac{\partial\left(\hat{s}_{lK}\right)}{\partial v_{ji}} = \frac{\partial}{\partial v_{ji}}\left[\sum_{J=1}^{n} w_{KJ}\, f\left(\sum_{I=0}^{m} v_{JI}\, x_{lI}\right) + w_{K0}\right] = \sum_{J=1}^{n}\frac{\partial}{\partial v_{ji}}\left[w_{KJ}\, f\left(\sum_{I=0}^{m} v_{JI}\, x_{lI}\right) + w_{K0}\right] =$$

$$= \frac{\partial}{\partial v_{ji}}\left[w_{Kj}\, f\left(\sum_{I=0}^{m} v_{jI}\, x_{lI}\right)\right] = \frac{\partial}{\partial y_{lj}}\left[w_{Kj}\, f\left(\sum_{I=0}^{m} v_{jI}\, x_{lI}\right)\right]\frac{\partial y_{lj}}{\partial v_{ji}} = w_{Kj}\frac{\partial}{\partial v_{ji}}\left[f\left(u_{lj}\right)\right] =$$

$$= w_{Kj}\frac{\partial}{\partial u_{lj}}\left[f\left(u_{lj}\right)\right]\frac{\partial u_{lj}}{\partial v_{ji}} = w_{Kj}\, f'\left(u_{lj}\right)\frac{\partial}{\partial v_{ji}}\left(\sum_{I=0}^{m} v_{jI}\, x_{lI}\right) = w_{Kj}\, f'\left(u_{lj}\right)\sum_{I=0}^{m}\frac{\partial}{\partial v_{ji}}\left(v_{jI}\, x_{lI}\right) =$$

$$= w_{Kj}\, f'\left(u_{lj}\right)\frac{\partial}{\partial v_{ji}}\left(v_{ji}x_{li}\right) = w_{Kj}\, f'\left(u_{lj}\right)x_{li}$$

$$\frac{\partial J(\theta)}{\partial v_{ji}} = -\sum_{l=1}^{N}\sum_{K=1}^{r} e_{lK} w_{Kj}\, f'\left(u_{lj}\right)x_{li} \Rightarrow \boxed{\frac{\partial J(\theta)}{\partial v_{ji}} = -\sum_{l=1}^{N}\sum_{K=1}^{r}\left(s_{lK} - \hat{s}_{lK}\right)w_{Kj}\, f'\left(u_{lj}\right)x_{li}}$$

Parte 2 da Solução:

$$\frac{\partial J(\theta)}{\partial w_{kj}} = \frac{\partial}{\partial w_{kj}}\left[\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\left(s_{lK} - \hat{s}_{lK}\right)^2\right] = \frac{\partial}{\partial w_{kj}}\left[\frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r} e_{lK}^2\right] = \frac{1}{2}\sum_{l=1}^{N}\sum_{K=1}^{r}\frac{\partial\left(e_{lK}^2\right)}{\partial w_{kj}} =$$

$$= \frac{1}{2}\sum_{l=1}^{N}\frac{\partial\left(e_{lk}^2\right)}{\partial w_{kj}} = \frac{1}{2}\sum_{l=1}^{N}\frac{\partial\left(e_{lk}^2\right)}{\partial e_{lk}}\frac{\partial\left(e_{lk}\right)}{\partial w_{kj}} = \sum_{l=1}^{N} e_{lk}\frac{\partial\left(e_{lk}\right)}{\partial w_{kj}} = \sum_{l=1}^{N} e_{lk}\frac{\partial\left(e_{lk}\right)}{\partial \hat{s}_{lk}}\frac{\partial\left(\hat{s}_{lk}\right)}{\partial w_{kj}} = -\sum_{l=1}^{N} e_{lk}\frac{\partial\left(\hat{s}_{lk}\right)}{\partial w_{kj}}$$

Sabendo que $\hat{s}_{lk} = \sum_{J=1}^{n} w_{kJ}\, f\left(\sum_{I=0}^{m} v_{JI}x_{lI}\right) + w_{k0}$, então tem-se:

$$\frac{\partial\left(\hat{s}_{lk}\right)}{\partial w_{kj}} = \frac{\partial}{\partial w_{kj}}\left[\sum_{J=1}^{n} w_{kJ}\, f\left(\sum_{I=0}^{m} v_{JI}x_{lI}\right) + w_{k0}\right] = \sum_{J=1}^{n}\frac{\partial}{\partial w_{kj}}\left[w_{kJ}\, f\left(\sum_{I=0}^{m} v_{JI}x_{lI}\right) + w_{k0}\right] =$$

$$= \begin{cases} \dfrac{\partial}{\partial w_{kj}} \left[w_{kj} f\left(\sum_{I=0}^{m} v_{jI} x_{lI} \right) \right] & \text{se } j \neq 0 \\ \dfrac{\partial}{\partial w_{kj}} \left[w_{k0} \right] & \text{se } j = 0 \end{cases} = \begin{cases} f\left(\sum_{I=0}^{m} v_{jI} x_{lI} \right) & \text{se } j \neq 0 \\ 1 \text{ se } j = 0 \end{cases}$$

$$\frac{\partial J(\theta)}{\partial w_{kj}} = -\sum_{l=1}^{N} e_{lk} \frac{\partial(\hat{s}_{lk})}{\partial w_{kj}} = \begin{cases} -\sum_{l=1}^{N} e_{lk} f\left(\sum_{I=0}^{m} v_{jI} x_{lI} \right) & \text{se } j \neq 0 \\ -\sum_{l=1}^{N} e_{lk} & \text{se } j = 0 \end{cases}$$

$$\boxed{\frac{\partial J(\theta)}{\partial w_{kj}} = \begin{cases} -\sum_{l=1}^{N} (s_{lk} - \hat{s}_{lk}) f\left(\sum_{I=0}^{m} v_{jI} x_{lI} \right) & \text{se } j \neq 0 \\ -\sum_{l=1}^{N} (s_{lk} - \hat{s}_{lk}) & \text{se } j = 0 \end{cases}}$$

23. Outros métodos de otimização não-linear empregados no treinamento de RNAs

- Os desafios associados ao treinamento de redes neurais profundas (*deep learning*) levaram à proposição de algoritmos supostamente mais competentes para o ajuste de pesos, no sentido de promoverem, em média, progressos mais expressivos na exploração da superfície de erro para a mesma carga de recursos computacionais empregada durante o treinamento da rede neural (RUDER, 2017).
- Alguns desses avanços são truques computacionais de natureza empírica, como os algoritmos adaptativos que não recorrem diretamente a informações de 2a. ordem, enquanto outros são melhor suportados por fundamentos conceituais, como a técnica de *dropout* (a ser abordada mais adiante neste curso) usada para conter o sobreajuste (*overfitting*) no processo de treinamento.
- São apresentados, a seguir, apenas noções básicas de parte desses progressos recentes em otimização não-linear voltada para o ajuste de pesos de RNAs.

- Evidentemente, existem iniciativas voltadas para outras estratégias de aprendizado de máquina, além do treinamento de RNAs (BOTTOU *et al.*, 2018).

23.1 Mini-batch gradient descent

- Trata-se de uma técnica intermediária entre os métodos padrão-a-padrão (*stochastic gradient descent* – SGD) e batelada (*batch*). Mais detalhes podem ser obtidos em: [https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size/].
- Definem-se partições ou lotes (subconjuntos) de dados de treinamento, sendo que o treinamento para cada lote é em batelada. A evolução do erro não oscila tanto quanto no caso padrão-a-padrão, mas possivelmente oscila o suficiente para escapar de mínimos locais ruins (LI *et al.*, 2014).
- Além disso, há um favorecimento das implementações em computação paralela e uma redução no uso de memória, já que não é necessário manter simultaneamente todos os dados de treinamento em memória.

23.2 Algoritmos adaptativos

- Os algoritmos adaptativos foram motivados basicamente pela limitação associada ao uso de uma mesma taxa de aprendizado para todos os padrões de entrada ou então pela definição prévia de uma política de adaptação dessa taxa de aprendizado. Em ambos os casos, é desafiador definir adequadamente e a priori essa única taxa de aprendizado ou então essa política de ajuste da taxa.
- Deve-se levar em conta que uma taxa de aprendizado muito pequena produz uma convergência muito lenta do treinamento, enquanto que uma taxa de aprendizado muito elevada pode impedir a convergência do processo de ajuste, seja por apresentar oscilações em torno de um ótimo local, seja por divergir.
- Além disso, usar uma mesma taxa de aprendizado para todos os pesos ajustáveis, seja ela fixa ou adaptativa, pode não ser a melhor solução em aplicações práticas caracterizadas, por exemplo, pela existência de dados esparsos, em que o ajuste dos pesos se dá em frequências distintas. Neste caso, pode não ser uma boa estratégia

ajustar os pesos com a mesma intensidade a cada iteração. Tende a ser mais produtivo, assim, promover ajustes mais intensos para pesos que sofrem ajustes mais raros ou menos intensos num histórico de ajustes recentes.

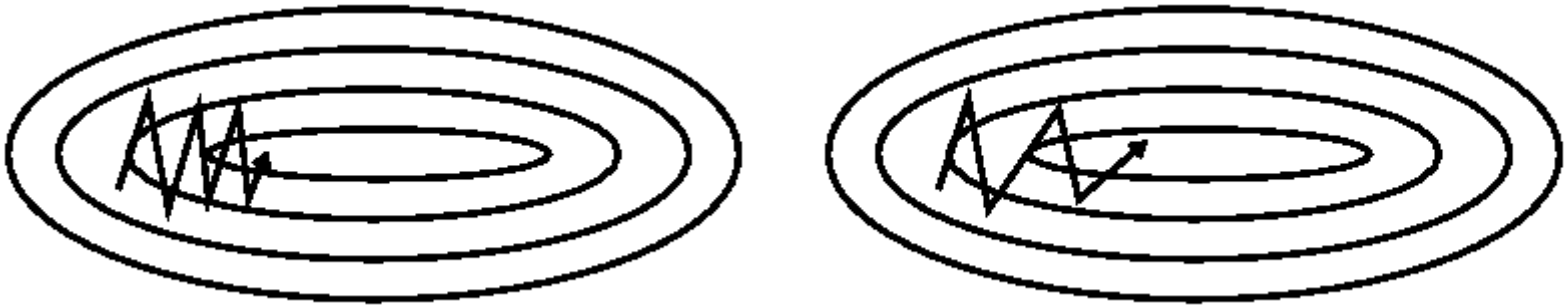
- Outro aspecto-chave em *deep learning* é a existência de uma quantidade muito grande de mínimos locais de baixa qualidade, de pontos de sela e de plateaus. Escapar dessas "armadilhas" geralmente requer passos de treinamento adaptativos.

23.3 Algoritmo adaptativo 1: SGD + momentum

- Sua principal missão é reduzir oscilações durante a exploração da superfície de erro, mas também opera como acelerador sempre que gradientes subsequentes preservarem direções de busca (QIAN, 1999).

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha \nabla J_l\left(\mathbf{w}(k)\right) + \gamma\left[\mathbf{w}(k) - \mathbf{w}(k-1)\right]$$

- Um valor típico geralmente adotado para γ é 0,9. De qualquer modo, deve-se considerar o fato de que o processo de ajuste ganhou uma ordem na dinâmica e um hiperparâmetro a mais, a ser devidamente definido.



Sem termo de momentum Com termo de momentum

Figura 31 – Contribuição típica do termo de momentum na redução de oscilações. Fonte: [https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d]

23.4 Algoritmo adaptativo 2: *Nesterov accelerated gradient* (NAG)

- Numa tentativa de acelerar a busca, é possível calcular o vetor gradiente não em $\mathbf{w}(k)$, mas em $\mathbf{w}(k)+\gamma\left[\mathbf{w}(k)-\mathbf{w}(k-1)\right]$, que é uma aproximação do novo valor do vetor de pesos da rede neural. Com isso, a regra de ajuste assume a forma:

$$\mathbf{w}(k+1)=\mathbf{w}(k)-\alpha\nabla J_l\left(\mathbf{w}(k)+\gamma\left[\mathbf{w}(k)-\mathbf{w}(k-1)\right]\right)+\gamma\left[\mathbf{w}(k)-\mathbf{w}(k-1)\right]$$

- Trata-se de uma tentativa de antecipar o comportamento da superfície de erro.

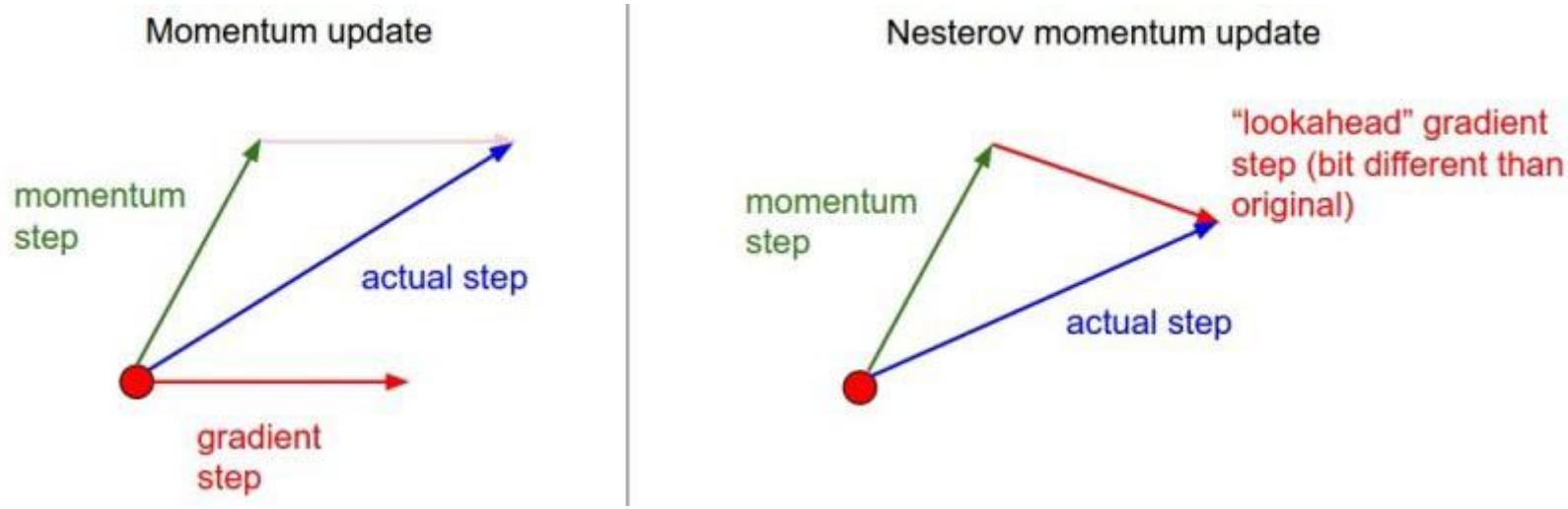


Figura 32 – Motivação geométrica da diferença entre SGD+momentum e NAG. Fonte: [https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d]

23.5 Estado-da-arte em algoritmos adaptativos

- Antes de listar as propostas mais avançadas em algoritmos adaptativos, é importante salientar que há dois aspectos a serem considerados ao medir a competência de um algoritmo de otimização não-linear no treinamento de redes neurais artificiais: (1) eficiência computacional; e (2) capacidade de produzir redes neurais que generalizam bem.

- De fato, os algoritmos adaptativos de uso mais frequente na literatura tendem a produzir ganhos de eficiência computacional, mas podem levar a redes neurais que generalizam pior que aquelas produzidas por *mini-batch* SGD (*stochastic gradient descent*), conforme investigado de forma empírica e também com evidências teóricas no trabalho de WILSON *et al*. (2018). A principal conclusão de WILSON *et al*. (2018) é que métodos adaptativos e não-adaptativos tendem a obter soluções de otimização diferentes e apresentar capacidades de generalização bem distintas.
- Como as propostas de algoritmos adaptativos são bem recentes, assim como essas críticas ao seu comportamento em aplicações práticas, associadas ao treinamento de redes neurais artificiais profundas ou não, cabem estudos mais detalhados visando chegar a regras de uso mais objetivas e efetivas. Havendo recursos computacionais disponíveis, por hora, a recomendação é treinar a rede neural com *mini-batch SCG* e também com ADAM, comparando o desempenho das redes neurais produzidas em cada caso, em termos de capacidade de generalização.

- O trabalho de RUDER (2017) apresenta uma formalização didática acerca das principais propostas em algoritmos adaptativos. Aqui, iremos apenas lista-las e citar referências relevantes em cada caso:
  - ✓ Adagrad (DUCHI *et al.*, 2011)
  - ✓ Adadelta (ZEILER, 2012)
  - ✓ ADAM (*Adaptive Moment Estimation*) (KINGMA & BA, 2015)
- O ADAM, em particular, calcula as taxas individuais de aprendizagem adaptativa para diferentes pesos da RNA a partir das estimativas de primeiro e segundo momentos dos gradientes consecutivos.
- Embora métodos de segunda ordem tradicionais, como o método de Newton, não sejam factíveis quando o número de parâmetros ajustáveis é elevado, existem iniciativas que viabilizam o emprego da informação de segunda ordem sem empregar a matriz hessiana, sendo técnicas bastante promissoras em *deep learning* (MARTENS, 2010), mas ainda pouco exploradas.

24. Referências

ARBIB, M.A. (ed.) "The Handbook of Brain Theory and Neural Networks", The MIT Press, 1998.

BATTITI, R. "First- and Second-Order Methods for Learning: Between Steepest Descent and Newton's Method", *Neural Computation*, vol. 4, no. 2, pp. 141-166, 1992.

BISHOP, C. "Exact Calculation of the Hessian Matrix for the Multilayer Perceptron", *Neural Comp.*, vol. 4, no. 4, pp. 494-501, 1992.

BOTTOU, L., CURTIS, F.E., NOCEDAL, J. "Optimization Methods for Large-Scale Machine Learning", *arXiv*:1606.04838v3, 2018.

CYBENKO, G. "Approximation by superposition of sigmoidal functions", *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303-314, 1989.

DAYAN, P., ABBOT, L.F. "Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems", The MIT Press, 2001.

DUCHI, J., HAZAN, E., SINGER, Y. "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

EDELMAN, G.M. "Neural Darwinism: The Theory of Neuronal Group Selection", Basic Books, 1988.

GARDNER, H. "Frames of Mind: The Theory of Multiple Intelligences", BasicBooks, 1983.

HAYKIN, S. "Neural Networks: A Comprehensive Foundation", 2nd edition, Prentice-Hall, 1999.

HAYKIN, S. "Neural Networks and Learning Machines", 3rd edition, Prentice Hall, 2008.

HEBB, D. O. "The Organization of Behavior", Wiley, 1949.

HINTON, G.E. "Connectionist learning procedures", *Artificial Intelligence*, 40: 185-234, 1989.

HINTON, G. E. & SEJNOWSKI, T.J. "Learning and relearning in Boltzmann machines", *in* D. E. Rumelhart, J. L. McClelland & The PDP Research Group (eds.) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, MIT Press, vol. 1, pp. 282-317, 1986.

HOPFIELD, J.J. "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Sciences of the U.S.A.*, vol. 79, pp. 2554-2558, 1982.

HORNIK, K., STINCHCOMBE, M., WHITE, H. "Multi-layer feedforward networks are universal approximators", *Neural Networks*, vol. 2, no. 5, pp. 359-366, 1989.

HORNIK, K., STINCHCOMBE, M., WHITE, H. "Universal approximation of an unknown function and its derivatives using multilayer feedforward networks", *Neural Networks*, vol. 3, no. 5, pp. 551-560, 1990.

HORNIK, K., STINCHCOMBE, M., WHITE, H., AUER, P. "Degree of Approximation Results for Feedforward Networks Approximating Unknown Mappings and Their Derivatives", *Neural Computation*, vol. 6, no. 6, pp. 1262-1275, 1994.

LI, M., ZHANG, T., CHEN, Y., SMOLA, A.J. "Efficient Mini-batch Training for Stochastic Optimization", *Proceedings of the 20th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining* (KDD'2014), pp. 661-670, 2014.

KINGMA, D.P., BA, J.L. "Adam: A Method for Stochastic Optimization", *International Conference on Learning Representations*, pp. 1-13, 2015.

MARR, D. "A theory for cerebral neocortex", *Proceedings of the Royal Society of London, Series B*, 176: 161-234, 1970.

MARTENS, J. "Deep learning via Hessian-free optimization", *Proceedings of the 27th International Conference on Machine Learning* (ICML'2010), pp. 735-742, 2010.

MCCULLOCH, W.S. & PITTS, W. "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115-133, 1943.

MINSKY, M.L. "The Society of Mind", Simon & Schuster, 1988.

MINSKY, M.L. & PAPERT, S.A. "Perceptrons: Introduction to Computational Geometry", Expanded edition, The MIT Press, 1988 (1st edition: 1969).

NERRAND, O., ROUSSEL-RAGOT, P., PERSONNAZ, L., DREYFUS, G. "Neural Networks and Nonlinear Adaptive Filtering: Unifying Concepts and New Algorithms". *Neural Computation*, vol. 5, no. 2, pp. 165-199, 1993.

QIAN, N. “On the momentum term in gradient descent learning algorithms”, *Neural Networks*, vol. 12, pp. 145-151, 1999.

RAMÒN Y CAJAL, S. “Histologie du système nerveux de l'homme et des vertébré”, A. Maloine, Paris, 1909.

RUDER, S. “An overview of gradient descent optimization algorithms”, *arXiv*: 1609.04747v2, 14 pages, 2017.

RUMELHART, D.E. & MCCLELLAND, J.L. “Parallel Distributed Processing: Explorations in the Microstructure of Cognition”, vols. 1 & 2, The MIT Press, 1986.

SCHALKOFF, R.J. “Artificial Neural Networks”, The McGraw-Hill Companies, 1997.

WIENER, N. “Cybernetics”, The MIT Press, 1948.

WILSON, A.C., ROELOFS, R., STERN, M., SREBRO, N., RECHT, B. “The Marginal Value of Adaptive Gradient Methods in Machine Learning”, *arXiv*: 1705.08292v2, 2018.

WILSON, R.A., KEIL, F.C. (eds.) “The MIT Encyclopedia of the Cognitive Sciences”, The MIT Press, 2001.

ZEILER, M.D. “ADADELTA: An Adaptive Learning Rate Method”, *arXiv*: 1212.5701, 2012.