Redes Neurais Recorrentes (Parte 2)

Índice Geral

1 Introdução

- Inspirada em conceitos de física estatística e dinâmica não-linear;
- Principais características: Unidades computacionais não-lineares
  Simetria nas conexões sinápticas
  Totalmente realimentada (exceto auto-realimentação)

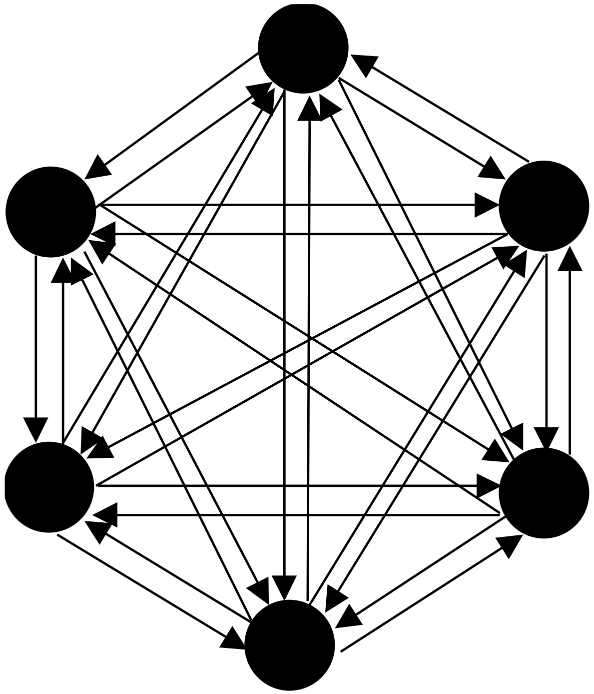


Figura 1 – Rede Neural de Hopfield: ênfase nas conexões

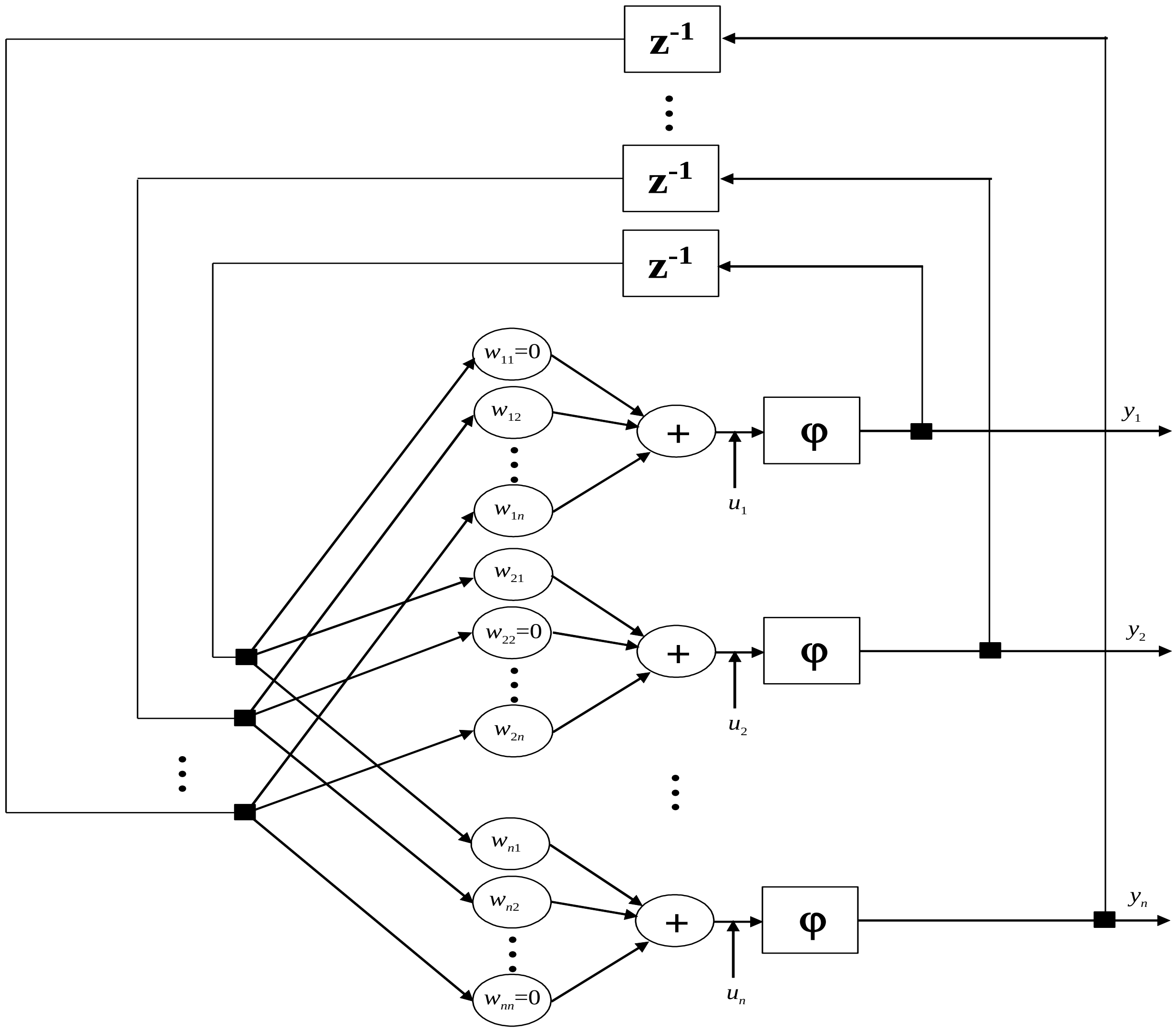
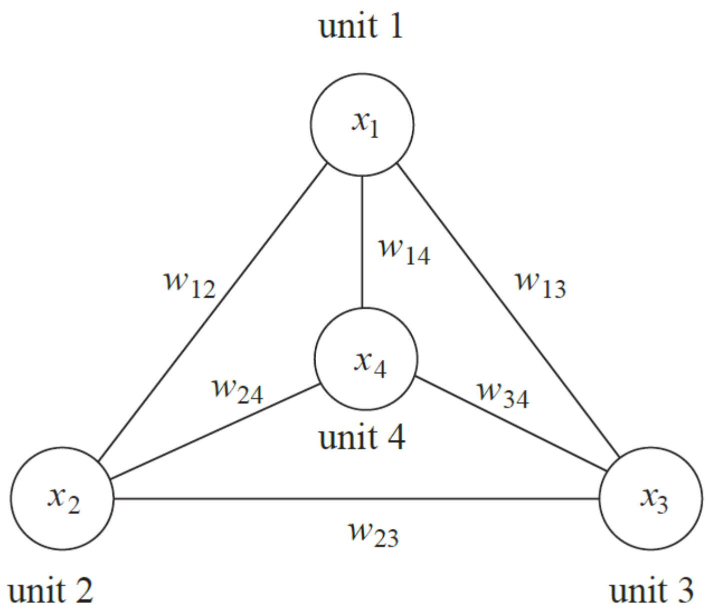


Figura 2 – Rede Neural de Hopfield: ênfase no processamento dinâmico (caso discreto)

- A rede de Hopfield é uma rede neural recorrente cujo padrão de conexões sinápticas leva à existência de uma função de Lyapunov associada à dinâmica não-linear.
- Isso implica que a dinâmica é caracterizada apenas pela existência de múltiplos pontos de equilíbrio, de tal forma que o sistema dinâmico evolui de um estado inicial qualquer a um estado final que é um mínimo local da superfície de energia associada à função de Lyapunov.
- O estado final converge sempre para o mínimo da superfície de energia associado à bacia de atração em que se encontra o estado inicial, pois a função de Lyapunov decresce monotonicamente sob a ação da dinâmica do sistema e é limitada inferiormente.
- Existem as versões de tempo contínuo e estado contínuo e de tempo discreto e estado binário, sendo que já foi demonstrado que a rede neural de Hopfield é Turing-completa ou computacionalmente universal (HERKEN, 1995).

- Como um exemplo, pode-se considerar uma associação de uma rede de Hopfield com a porta lógica OU-exclusivo.
- Os 4 estados da tabela à esquerda podem ser feitos pontos de equilíbrio estáveis para um certo conjunto de pesos da rede neural à direita. Com isso, definindo os valores iniciais das unidades 1, 2 e 4 (a unidade 4 pode ser considerada uma unidade auxiliar), obtém-se o resultado do OU-exclusivo na unidade 3.

unit	1	2	3	4
state 1	−1	−1	−1	1
state 2	1	−1	1	1
state 3	−1	1	1	1
state 4	1	1	−1	−1



2 Rede neural de Hopfield

2.1 Modelos derivados da física estatística

- <u>Incorporação de um princípio físico fundamental</u>: armazenagem de informação em uma configuração dinamicamente estável, o que requer um tempo para se acomodar em uma condição de equilíbrio → **dinâmica de relaxação** → comportamento de estado estacionário.
- Cada padrão a ser armazenado fica localizado em um vale da superfície de energia. Como a dinâmica não-linear da rede é estabelecida de modo a minimizar a energia, os vales representam pontos de equilíbrio estável (cada qual com a sua base de atração).
- <u>Memória ↔ Ponto de equilíbrio estável</u>: embora outros pesquisadores já viessem buscando a implementação de tal conceito, HOPFIELD (1982) foi o primeiro a formulá-lo em termos precisos.

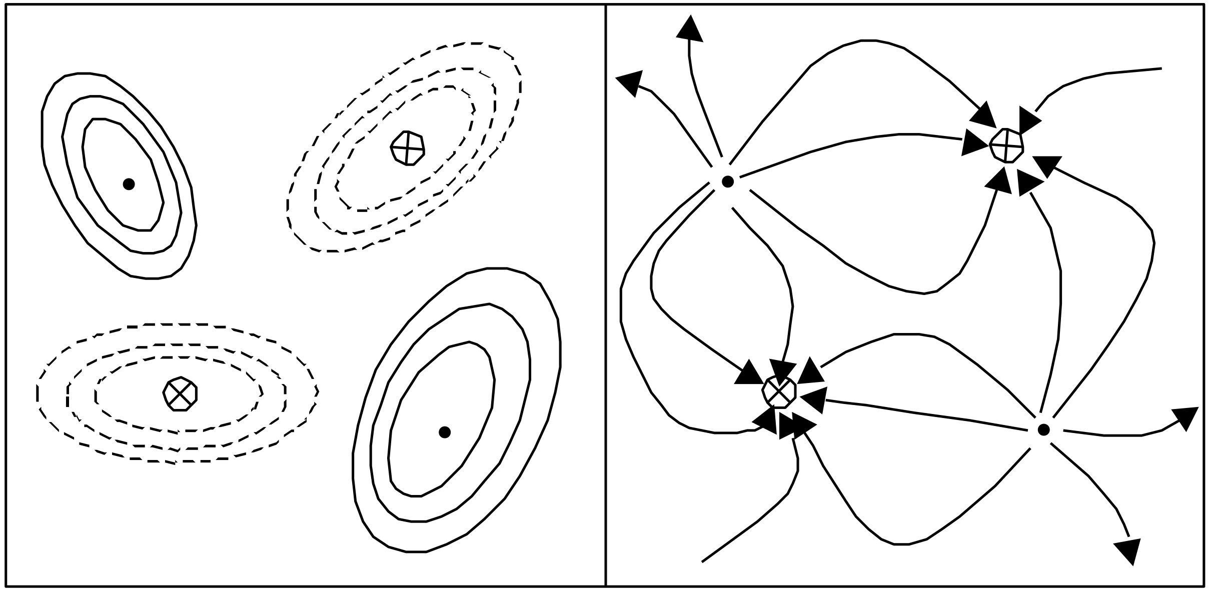


Figura 3 – Superfície de energia: pontos de equilíbrio e bases de atração.

- Considera-se, na Figura 3, um espaço de estados bidimensional. Logo, a superfície de energia reside no $\Re^3$, sendo que a energia é medida no eixo perpendicular ao plano apresentado. À esquerda, tem-se curvas de nível da superfície de energia, indicando a ocorrência de picos e vales. À direita, são apresentadas famílias de trajetórias que se iniciam próximas a pontos de equilíbrio instáveis.

- Este tipo de sistema dinâmico pode operar como:
  1) Memória associativa (endereçável por conteúdo);
  2) Dispositivo computacional para resolver problemas de otimização de natureza combinatória;
  3) Dispositivo computacional para resolver problemas de programação não-linear.

2.2 Especificações dinâmicas (espaço de estados contínuo e dinâmica contínua)

- Considere uma rede neural composta de N neurônios com acoplamento simétrico descrito por $w_{ji} = w_{ij}$ ($i,j = 1,\ldots,N$), onde w_{ji} é o peso sináptico que conecta a saída do neurônio i à entrada do neurônio j, e também com ausência de auto-realimentação, ou seja, $w_{jj} = 0$ ($j = 1,\ldots,N$). Observação: A simetria e a ausência de auto-realimentação não são necessárias em propostas mais recentes.

- Uma generalização, com $w_{ji} \neq w_{ij}$ e/ou $w_{jj} \neq 0$ $(i,j = 1,\ldots,N)$, pode resultar, na formulação de Hopfield, em um sistema dinâmico com outros comportamentos de estado estacionário, além dos pontos fixos (pontos de equilíbrio). Por exemplo: ciclos limites, quase periodicidade e caos.
- Sejam $u_j(t)$ o sinal de ativação interna do neurônio j e $y_j(t)$ o correspondente sinal de saída. Então tem-se que:

$$y_j(t) = \varphi_j\left(u_j(t)\right)$$

onde $\varphi_j(\cdot)$ é a não-linearidade sigmoidal do neurônio j. Neste desenvolvimento, estamos considerando u_j e y_j como variáveis de tempo contínuo.

- Considerando que o estado da rede (a saída $\mathbf{y}(t)$ também poderia ser escolhida como vetor de estados, já que a função de ativação é inversível) é dado por

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_N(t) \end{bmatrix},$$

a dinâmica da rede neural é descrita por um conjunto de equações diferenciais não-lineares acopladas (HOPFIELD, 1984; COHEN & GROSSBERG, 1983), na forma:

$$C_j \frac{du_j}{dt} = \sum_{\substack{i=1 \\ i \neq j}}^{N} w_{ji}\varphi_i(u_i) - \frac{u_j}{R_j} - \theta_j,\ j = 1,\ldots,N.$$

onde θ_j é um limiar aplicado ao neurônio j por uma fonte externa. O efeito capacitivo associado ao neurônio j, dado por C_j, determina a taxa finita de variação do sinal de ativação interna $u_j(t)$ em relação ao tempo t, que é uma propriedade intrínseca dos neurônios biológicos e também da implementação física dos neurônios artificiais.

- Para esta dinâmica, é possível definir uma função de energia, ou função de Lyapunov (HOPFIELD, 1984):

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j \neq i}}^{N} w_{ji} y_i y_j + \sum_{j=1}^{N}\frac{1}{R_j}\int_0^{y_j}\varphi_j^{-1}(z)dz + \sum_{j=1}^{N}\theta_j y_j$$

- Esta função de energia representa um caso particular de um teorema devido a COHEN & GROSSBERG (1983), e descreve totalmente a rede neural, ao considerar todos os pesos sinápticos e todas as variáveis de estado da rede.
- Além disso, supondo que θ_j varia lentamente com o tempo de computação, é possível definir o seguinte teorema:

Teorema 1: A função de energia E é uma função monotonicamente decrescente do estado da rede $\mathbf{u}(t)$.

Prova: Veja HOPFIELD (1984) e COHEN & GROSSBERG (1983).

- Isto implica que, dado um estado inicial $\mathbf{u}_0(t)$, a evolução do estado $\mathbf{u}(t)$ com o tempo vai seguir uma trajetória decrescente através da superfície de energia, até atingir um mínimo local. A partir deste ponto, o estado da rede fica constante, assim como a energia associada.

- Este é um resultado fundamental, pois garante que, mesmo com abundância de realimentações, os estados estacionários da rede neural recorrente descrita acima (um caso particular de sistema dinâmico não-linear) <u>correspondem sempre a pontos de equilíbrio</u>, não podendo assim apresentar oscilações permanentes do tipo ciclo limite, por exemplo, ou outros comportamentos estacionários mais complexos (como quase-periodicidade e caos).
- Os pontos de equilíbrio são denominados atratores, no sentido de que existe uma vizinhança (base de atração) sobre a qual estes pontos exercem uma influência dominante.
- Embora seja possível produzir redes recorrentes generalizadas, com $w_{ji} \neq w_{ij}$ e/ou $w_{jj} \neq 0$ ($i,j = 1,\ldots,N$), que apresentam uma dinâmica composta apenas por pontos de equilíbrio (CARPENTER *et al.*, 1987), geralmente se obtêm regimes estacionários caracterizados por comportamentos dinâmicos mais complexos (HOPFIELD & TANK, 1986), não desejados nesta aplicação específica.

- A Figura 4 a seguir ilustra a presença de dois ciclos limites no espaço de estados de um sistema dinâmico, sendo que se buscam conexões sinápticas para a rede de Hopfield que impedem a ocorrência deste tipo de comportamento. Repare que internamente a cada ciclo limite existe um ponto de equilíbrio instável.

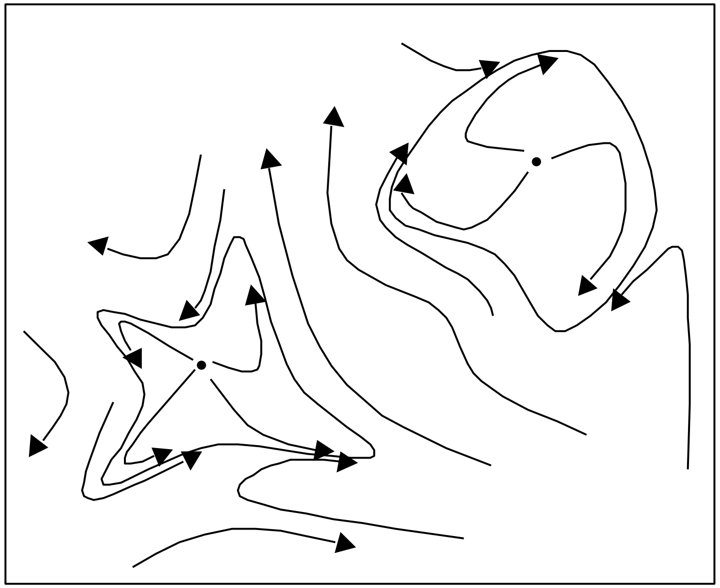


Figura 4 – Dinâmica de uma rede recorrente generalizada: presença de ciclos limites

2.3 Especificações dinâmicas (espaço de estados binário e dinâmica discreta)

- Um desenvolvimento equivalente leva à formulação para dinâmica discreta e neurônios que só podem assumir os estados binários –1 e +1. Novamente, valem $w_{ji} = w_{ij}$ ($i,j = 1,\ldots,N$), onde w_{ji} é o peso sináptico que conecta a saída do neurônio i à entrada do neurônio j, e $w_{jj} = 0$ ($j = 1,\ldots,N$).
- O modelo de rede neural de Hopfield utiliza como unidade de processamento básica o neurônio de MCCULLOCH & PITTS (1943).
- A ativação interna de cada neurônio j ($j = 1,\ldots,N$) é dada por

$$u_j(t) = \sum_{\substack{i=1 \\ i \neq j}}^{N} w_{ji} y_i(t) - \theta_j ,$$

onde θ_j é um limiar fixo aplicado externamente ao neurônio j.

- Tomando, agora, a saída y_j como o estado do neurônio j ($j = 1,\ldots,N$), este será modificado de acordo com a função de ativação dada pela função sinal:

$$y_j(t+1) = \text{sgn}[u_j(t)] = \begin{cases} +1 & \text{se}\, u_j(t) > 0 \\ -1 & \text{se}\, u_j(t) < 0 \end{cases}$$

- Por convenção, se $u_j = 0$, o estado y_j do neurônio permanece em seu valor anterior, seja ele –1 ou +1.
- Para que exista uma função de Lyapunov, é necessário impor que a atualização dos estados dos N neurônios seja feita de forma sequencial e em ordem aleatória. Não deve ocorrer a atualização síncrona dos estados.
- A função de Lyapunov aqui assume a forma:

$$E(t) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} w_{ji} y_i(t) y_j(t) + \sum_{j=1}^{N} \theta_j y_j(t).$$

- Assim, um ponto fixo estável $\bar{y}_\mu$ da rede de Hopfield é um estado de convergência a partir de uma condição inicial que pertence à sua base de atração.

- A partir deste ponto, e considerando espaço de estados binário e dinâmica discreta, vamos apresentar as duas fases de implementação da rede de Hopfield como memória associativa (endereçável por conteúdo). A primeira fase corresponde à síntese da dinâmica não-linear com base na definição dos pesos da rede de Hopfield (<u>memorização</u>). A segunda fase corresponde à restauração de uma dentre as memórias armazenadas (<u>estados de equilíbrio</u>), a partir de um padrão de entrada (<u>estado inicial</u>).

2.4 Fase 1: Armazenagem de padrões (memórias fundamentais)

- Suponha que se queira armazenar um conjunto de p padrões dados por vetores N-dimensionais (<u>palavras binárias</u>), denotados por $\{\xi_\mu \mid \mu=1,\ldots,p\}$.
- Para tanto, basta definir os pesos pela aplicação da regra de Hebb generalizada, ou regra do produto externo. Seja $\xi_{\mu i}$ o i-ésimo elemento do vetor ξ_μ, então o peso sináptico conectando o neurônio i ao neurônio j é definido por:

$$w_{ji} = \frac{1}{N}\sum_{\mu=1}^{p}\xi_{\mu j}\xi_{\mu i}\,,$$

sendo que novamente toma-se $w_{jj} = 0$ ($j = 1,\ldots,N$).

- Seja **W** a matriz $N \times N$ de pesos sinápticos, onde w_{ji} é o elemento da j-ésima linha e i-ésima coluna. Então, é possível expressar a regra do produto externo na forma:

$$\mathbf{W} = \frac{1}{N}\sum_{\mu=1}^{p}\xi_{\mu}\xi_{\mu}^{T} - \frac{p}{N}\mathbf{I}.$$

- Observe que $\mathbf{W} = \mathbf{W}^T$, ou seja, $w_{ji} = w_{ij}$ ($i,j = 1,\ldots,N$).

2.5 Fase 2: Recuperação dos padrões (estados de equilíbrio estáveis)

- Durante a fase de recuperação dos padrões armazenados (memórias), um vetor N-dimensional $\mathbf{y}(0)$ é tomado como o estado inicial (condição inicial) da rede de Hopfield. Obviamente, os elementos de **y** assumem valores –1 ou +1.

- Geralmente, este vetor **y** vai representar uma versão incompleta ou ruidosa da memória fundamental que foi armazenada na rede.
- O processo de recuperação da memória armazenada obedece a uma regra dinâmica denominada <u>ajuste assíncrono</u>. Um único neurônio j da rede é escolhido aleatoriamente para ter sua saída y_j (que agora está associada ao estado da rede) recalculada em função do valor de u_j.
- Assim, o ajuste do estado da rede, de uma iteração para outra, é determinístico, mas a escolha do neurônio cujo estado será atualizado é aleatória.
- Este ajuste assíncrono prossegue até que não haja mais mudanças de estado a processar, ou seja, até que a rede atinja um ponto de equilíbrio caracterizado por:

$$\bar{y}_j = \operatorname{sgn}\left(\sum_{i=1}^{N} w_{ji}\bar{y}_i - \theta_j\right),\ j = 1,\ldots,N,$$

ou em notação matricial:

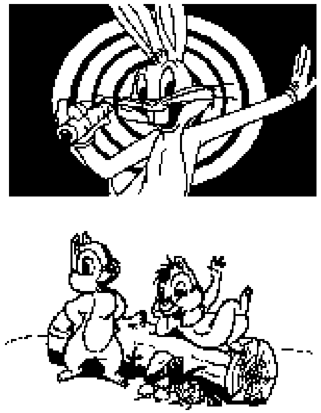
$$\bar{\mathbf{y}} = \operatorname{sgn}(\mathbf{W}\bar{\mathbf{y}} - \theta).$$

- A rede de Hopfield sempre vai convergir para um ponto de equilíbrio se o ajuste for assíncrono. Se o ajuste fosse síncrono, ou seja, todos os neurônios tendo seus estados recalculados em paralelo, ciclos limites de até 2 pontos poderiam ser produzidos (BRUCK, 1990).

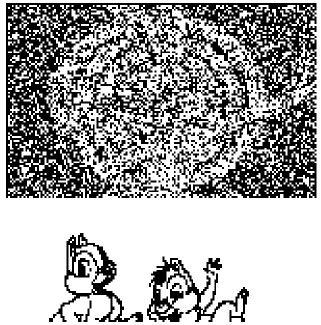
3 Princípio de operação da memória associativa

- A rede neural de Hopfield pode ser vista como uma memória associativa não-linear, ou uma memória endereçável por conteúdo, cuja principal função é restaurar um padrão binário armazenado (item de memória), em resposta à apresentação de uma versão incompleta (papel restaurador) ou ruidosa (papel de corretor de erro) deste padrão.
- É explorado aqui o controle da dinâmica por uma função de Lyapunov, mas é necessário que cada memória corresponda a um ponto de equilíbrio estável da dinâmica.

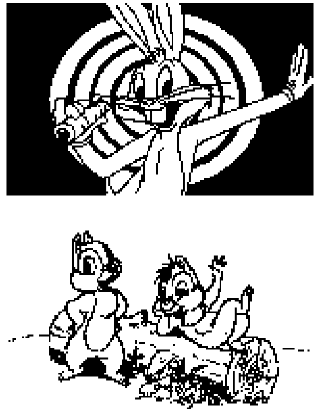
- Logo, deve-se obter um sistema dinâmico com tantos pontos de equilíbrio quanto forem as memórias e cuja localização no espaço de estados deve estar diretamente associada a essas memórias.



<u>memórias</u>



<u>entradas</u>

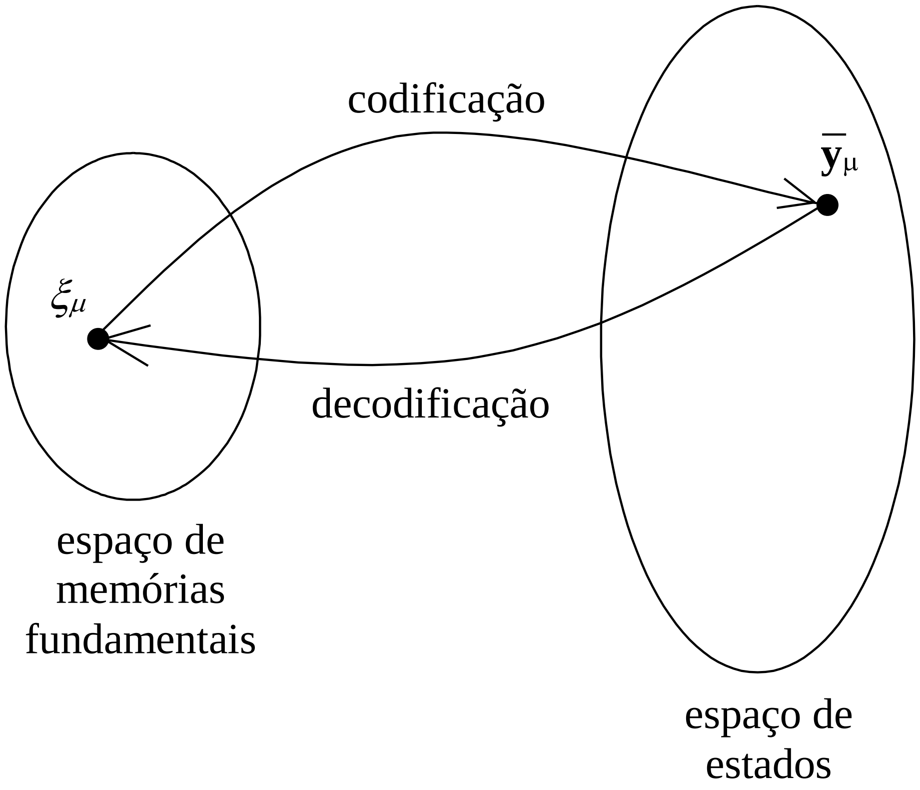


<u>padrões restaurados</u>

- Como memorizar?
- Como restaurar?

- Portanto, a recuperação do padrão armazenado na memória se dá a partir de um subconjunto das informações contidas no padrão (conteúdo parcial).

- A essência da memória endereçável por conteúdo é mapear uma memória fundamental ξ_μ em um ponto fixo estável $\bar{\mathbf{y}}_\mu$ do sistema dinâmico representado pela rede recorrente.



- Em outras palavras, a rede neural de Hopfield é um sistema dinâmico não-linear cujo espaço de estados contém um conjunto de pontos fixos estáveis que representam as memórias fundamentais do sistema.
- Percebe-se que a rede neural de Hopfield corresponde a um sistema dinâmico não-linear autônomo, pois não tem entrada externa.
- Foi visto na Parte 1 deste Tópico 5 que, em regime (após vencido o transitório ou transiente), sistemas dinâmicos não-lineares autônomos, sejam de tempo discreto ou contínuo, podem apresentar quatro comportamentos possíveis: pontos de equilíbrio, ciclos limites (soluções periódicas), soluções quase-periódicas e caos. Um mesmo sistema dinâmico pode apresentar múltiplos casos desses quatro comportamentos, dependendo da condição inicial (estado inicial do sistema dinâmico).
- Os pesos da rede neural de Hopfield não são definidos via algoritmos iterativos de treinamento, e sim via técnicas de síntese de dinâmicas não-lineares.

4 Regra de Hebb

- A regra de aprendizado de Hebb é a mais antiga e mais famosa regra de aprendizado, podendo também ser apresentada em duas partes, na forma:

  1. Se os dois neurônios localizados um em cada lado de uma conexão sináptica são ativados simultaneamente (de modo síncrono), então a intensidade da conexão é aumentada.

  2. Se os dois neurônios localizados um em cada lado de uma conexão sináptica são ativados de modo assíncrono, então a intensidade da conexão é reduzida.

- A segunda parte da regra de Hebb não fazia parte de sua versão original, tendo sido introduzida posteriormente.
- A regra de Hebb pode ser interpretada como um mecanismo (interativo, local e dependente do tempo) de aumentar a eficiência sináptica em função da correlação existente entre as atividades pré- e pós-sináptica.

5 Recapitulação

- Não-linearidade é condição necessária para produzir múltiplos atratores no espaço de estados de sistemas dinâmicos.
- Hopfield resolveu (parcialmente) o seguinte problema: *Dado um conjunto de estados específicos que devem estar associados a memórias fundamentais, como gerar um sistema dinâmico não-linear que apresente pontos de equilíbrio estável justamente nestes estados específicos?*
- Se este sistema dinâmico não-linear puder ser sintetizado, então vai existir uma superfície de energia com mínimos locais nos referidos estados específicos, sendo que a dinâmica do sistema vai atuar no sentido de conduzir o estado inicial do sistema a um dos mínimos locais da superfície de energia (particularmente àquele em cuja bacia de atração se encontra a condição inicial).
- Em uma dinâmica de relaxação, não é possível transitar entre bacias de atração, pois a energia nunca pode aumentar no decorrer do tempo.

6 A emergência de memória associativa

- Para a rede neural de Hopfield considerada, com $w_{ji} = w_{ij}$, $w_{jj} = 0$, caso discreto com $\theta_j = 0$ ($j = 1,\ldots,N$), ou caso contínuo com $R_j \to \infty$, $\theta_j = 0$ ($i,j = 1,\ldots,N$) e inclinação da função tangente hiperbólica tendendo a infinito, a função de energia pode ser definida na forma:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ji}y_i y_j = -\frac{1}{2}\mathbf{y}^T\mathbf{W}\mathbf{y}$$

onde $y_j = \text{sgn}(u_j) = \text{sgn}\left(\sum_{\substack{i=1\\ i\neq j}}^{N} w_{ji}y_i\right)$ (j=1,…,N).

- A mudança na função de energia ΔE, devido a uma mudança Δy_j (o novo valor menos o valor anterior) no estado do neurônio j, é dada por:

$$\Delta E = -\Delta y_j \sum_{\substack{i=1\\ i\neq j}}^{N} w_{ji}y_i = -\Delta y_j u_j$$

garantindo que a dinâmica da rede vai promover o decrescimento monotônico da função de energia no tempo. Repare que, se $\Delta y_j > 0$, então $u_j = +1$, e se $\Delta y_j < 0$, então $u_j = -1$.

<u>Exemplo para *N* = 3:</u>

- $E = -\dfrac{1}{2}\sum_{i=1}^{3}\sum_{\substack{j=1\\ j\neq i}}^{3} w_{ji} y_i y_j$
- $E = -\dfrac{1}{2}\left[w_{21}y_1y_2 + w_{31}y_1y_3 + w_{12}y_2y_1 + w_{32}y_2y_3 + w_{13}y_3y_1 + w_{23}y_3y_2\right]$
- Supondo que $\Delta y_2 \neq 0$:

$$\Delta E = -\frac{1}{2}\left[w_{21}y_1\Delta y_2 + w_{12}\Delta y_2 y_1 + w_{32}\Delta y_2 y_3 + w_{23}y_3\Delta y_2\right]$$

- Que produz:

$$\Delta E = -\left[w_{21}y_1\Delta y_2 + w_{23}y_3\Delta y_2\right] = -\Delta y_2\left[w_{21}y_1 + w_{23}y_3\right] \Rightarrow \Delta E = -\Delta y_2 \sum_{\substack{i=1\\ i\neq 2}}^{3} w_{ji} y_i$$

- Quando um estado de equilíbrio estável (ponto de mínimo local da função de energia) é atingido, não há como reduzir ainda mais a energia, fazendo com que o estado da rede fique <u>invariante frente à dinâmica</u>.
- Assim, para garantir a emergência de memória associativa, duas condições devem ser satisfeitas:

1. As memórias fundamentais devem ser armazenadas como estados estáveis da rede (pontos de equilíbrio);
2. Estes estados estáveis correspondentes às memórias fundamentais devem ter uma base de atração de dimensão não-nula.

- Existe uma associação direta entre a extensão da base de atração e a descorrelação das memórias, no sentido de que memórias muito correlacionadas tendem a estar próximas no espaço de estados, o que conduz a bacias de atração de tamanho menor para cada uma delas.

7 Atratores espúrios

- Quando a rede neural de Hopfield armazena K memórias fundamentais através do ajuste de seus pesos pela regra de Hebb generalizada, os estados estáveis presentes na superfície de energia não vão se restringir aos estados associados às memórias fundamentais armazenadas. Todos os estados estáveis não associados às memórias fundamentais armazenadas são denominados atratores espúrios.
- Os atratores espúrios existem em virtude dos seguintes fatores:

1. A função de energia E é simétrica, no sentido de que os estados correspondentes ao reverso das memórias fundamentais armazenadas também são estados estáveis;
2. Toda combinação linear de um número ímpar de estados estáveis também vai ser um estado estável (AMIT, 1989).
3. Para um grande número K de memórias fundamentais, a função de energia vai produzir pontos de equilíbrio que não estão correlacionados com nenhuma das memórias fundamentais armazenadas na rede (para se chegar às contorções requeridas que levam a vales, outras contorções indesejadas podem surgir).

8 Capacidade de memória da rede de Hopfield

- Infelizmente, as memórias fundamentais utilizadas para gerar os pesos da rede de Hopfield, de acordo com a seguinte equação:

$$w_{ji} = \frac{1}{N}\sum_{\mu=1}^{p}\xi_{\mu j}\xi_{\mu i}\,,$$

nem sempre conduzem a estados estáveis em todas as memórias fundamentais.

- Desse modo, a possível existência de estados espúrios, aliada à possibilidade de que memórias fundamentais não correspondam a vales da superfície de energia, tendem a reduzir a eficiência da rede de Hopfield como uma memória endereçável por conteúdo.
- Considere a ativação interna do neurônio *j*, dada na forma:

$$u_j = \sum_{i=1}^{N} w_{ji} y_i\,,$$

onde, para efeito de generalidade, estamos supondo agora que $w_{jj} \neq 0$.

- Denominando **x** um estado genérico da rede, temos que:

$$u_j = \frac{1}{N}\sum_{i=1}^{N}\sum_{\mu=1}^{p}\xi_{\mu j}\xi_{\mu i}x_i = \frac{1}{N}\sum_{\mu=1}^{p}\xi_{\mu j}\sum_{i=1}^{N}\xi_{\mu i}x_i$$

- Considere agora o caso especial em que o estado genérico **x** é tomado como uma das memórias fundamentais armazenadas na rede, por exemplo, ξ_ν:

$$u_j = \frac{1}{N}\sum_{\mu=1}^{p}\xi_{\mu j}\sum_{i=1}^{N}\xi_{\mu i}\xi_{\nu i} \Rightarrow u_j = \frac{1}{N}\xi_{\nu j}\sum_{i=1}^{N}\xi_{\nu i}\xi_{\nu i} + \frac{1}{N}\sum_{\substack{\mu=1\\ \mu\neq\nu}}^{p}\xi_{\mu j}\sum_{i=1}^{N}\xi_{\mu i}\xi_{\nu i}$$

$$u_j = \xi_{\nu j} + \frac{1}{N}\sum_{\substack{\mu=1\\ \mu\neq\nu}}^{p}\xi_{\mu j}\sum_{i=1}^{N}\xi_{\mu i}\xi_{\nu i}$$

- A parcela mais à esquerda, $\xi_{\nu j}$, é simplesmente o *j*-ésimo elemento da memória fundamental ξ_ν, constituindo o valor desejado (sinal) para u_j, já que a memória fundamental deve ser um estado estável. Este resultado justifica a necessidade da divisão por *N* na geração dos pesos.

- A parcela mais à direita, $\frac{1}{N}\sum_{\substack{\mu=1\\ \mu\neq\nu}}^{p}\xi_{\mu j}\sum_{i=1}^{N}\xi_{\mu i}\xi_{\nu i}$, é o <u>ruído</u> existente quando os padrões não são ortogonais, ou seja, quando $\sum_{i=1}^{N}\xi_{\mu i}\xi_{\nu i} \neq 0$.
- Através de um estudo estatístico, supondo, dentre outros aspectos, que as memórias fundamentais são formadas por padrões gerados aleatoriamente, é possível mostrar que a relação sinal-ruído é dada aproximadamente por

$$\rho \cong \frac{N}{K},\ \text{para valores elevados de } K \text{ (número de memórias).}$$

- Com isso, o componente de memória fundamental ξ_ν será estável (em sentido probabilístico) se, e somente se, a relação sinal-ruído for suficientemente alta.
- Valores sugeridos na literatura para ρ:

  1. $\rho = 7{,}25$, ou $\frac{1}{\rho} = 0{,}138$ ($K = 138$ quando $N = 1000$);

  2. $\rho \geq 2\ln N \Rightarrow K \leq \frac{N}{2\ln N}$ ($K \leq 72$ quando $N = 1000$).

9 Extensões (Parte I)

- A rede neural de Hopfield apresentada anteriormente usa dinâmica discreta e toma o neurônio de McCulloch-Pitts como unidade básica. No entanto, um desempenho muito superior, quanto à capacidade de memória, pode ser obtido empregando funções de ativação contínuas e não-monotônicas (MORITA, 1993). A rede passa a pertencer à segunda geração de redes neurais artificiais (Tópico 5 – Parte 1).
- Obviamente, como a função de ativação passa a assumir valores em um intervalo, é necessário aplicar a função sinal para recuperar a memória a partir da saída estabilizada da rede.
- O resultado de MORITA (1993) traz como consequência:

1. Aumento da capacidade de memória de $\dfrac{N}{2\ln N}$ para 0,4N, onde N é o número de neurônios;
2. Desaparecimento dos estados espúrios.

10 Extensões (Parte II)

- Uma regra mais eficiente para a definição dos pesos da rede de Hopfield convencional, em lugar da regra de Hebb generalizada, é a regra de projeção.
- Ela tem a desvantagem de não apresentar uma motivação biológica, mas a vantagem de explorar as propriedades algébricas dos pontos de equilíbrio.
- ξ_μ será um ponto de equilíbrio se $\mathbf{W}\xi_\mu = \xi_\mu$, $\mu=1,\ldots,K$.
- Seja $\mathbf{P} = [\xi_1 \quad \cdots \quad \xi_K]$, então temos que $\mathbf{WP} = \mathbf{P}$.
- Uma solução para **W** é dada na forma: $\mathbf{W} = \mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T$.
- Para que exista $(\mathbf{P}^T\mathbf{P})^{-1}$, basta que os vetores ξ_μ, $\mu=1,\ldots,K$, sejam linearmente independentes, pois esta condição garante que a matriz **P** tenha posto completo.
- A matriz $\mathbf{P}^+ = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T$ é denominada pseudo-inversa de **P** (Moore-Penrose).
- $\mathbf{W} = \mathbf{PP}^+$ é a matriz de projeção ortogonal de vetores do $\Re^N$ para o subespaço cuja base é o conjunto de memórias fundamentais ξ_μ, $\mu=1,\ldots,K$.

11 Problemas de natureza combinatória

- São problemas que se enquadram entre aqueles de mais difícil solução com base nas ferramentas matemáticas e computacionais hoje disponíveis;
- <u>Exemplo</u>: Problema do caixeiro viajante (TSP) $\rightarrow$ Dadas as localizações de um número específico N de cidades (distribuídas em um plano), o problema é encontrar o menor percurso que se inicia e termina numa mesma cidade, tendo passado uma única vez por todas as outras cidades. É um problema de fácil formulação, mas para o qual não se conhece nenhum método que garanta a obtenção da solução ótima, além do método exaustivo de testar todas as possibilidades e optar pela que produz o menor percurso. Em virtude da explosão (crescimento fatorial em N) de percursos possíveis com o aumento no número de cidades, o método exaustivo torna-se computacionalmente intratável, mesmo para problemas com um número reduzido de cidades (por exemplo, para 100 cidades, o número de percursos possíveis é da ordem de 10^{156}).

- Em termos de complexidade computacional, o problema do caixeiro viajante é *NP*-completo.
- A aplicação pioneira de redes de Hopfield no tratamento do problema do caixeiro viajante (uma abordagem extensível a outros problemas de natureza combinatória) se deu com o trabalho de HOPFIELD & TANK (1985). Basicamente, foi considerada uma rede neural com estados contínuos, com uma dinâmica representada na forma de um conjunto de equações diferenciais acopladas, na forma:

$$\frac{du_j}{dt} = \sum_{\substack{i=1 \\ i \neq j}}^{N} w_{ji}\varphi(u_i) - \frac{u_j}{\tau} - \theta_j,\ j = 1,\ldots,N.$$

$$y_j = \varphi(u_j)$$

- Os pesos sinápticos da rede são determinados com base nas distâncias entre as cidades e a solução corresponde a um ponto de equilíbrio (mínimo local da superfície de energia) no espaço de estados da rede neural.

- Ao mesmo tempo em que é necessário minimizar a função-objetivo, a qual avalia a distância total do percurso, também existem restrições a serem atendidas, como passar ao menos uma vez em cada cidade.
- Como a violação de uma única restrição torna a correspondente solução inválida, é necessário incorporar à função-objetivo termos que penalizam a violação de cada restrição. Além disso, esta função-objetivo estendida deve corresponder à superfície de energia da rede de Hopfield, de tal forma que a aplicação da dinâmica da rede conduza o estado sempre para pontos de menor energia. Com isso, uma possível representação da função de energia assume a forma:

$$E = E^{obj} + c_1 E_1^{restr} + \cdots + c_m E_m^{restr}$$

- A formulação original empregada por HOPFIELD & TANK (1985) é apresentada a seguir:

$$E = \frac{A}{2}\left(\sum_X \sum_i \sum_{j \neq i} y_{Xi} y_{Xj}\right) + \frac{B}{2}\left(\sum_i \sum_X \sum_{Y \neq X} y_{Xi} y_{Yi}\right) + \frac{C}{2}\left(\sum_X \sum_i y_{Xi} - N\right)^2$$
$$+ \frac{D}{2}\left(\sum_X \sum_{Y \neq X} \sum_i d_{XY} y_{Xi}\left(y_{Y(i+1)} + y_{Y(i-1)}\right)\right)$$

onde N é o número de cidades e A, B, C, e D são coeficientes de ponderação a serem devidamente definidos. Os neurônios devem ser organizados em uma grade $N \times N$, de tal modo que X e Y representam índices das linhas e i e j representam índices das colunas, todos assumindo valores no conjunto $\{1, 2, \cdots, N\}$.

- Primeiro e segundo termos penalizam mais de um neurônio ativo por linha e por coluna. Já o terceiro termo penaliza um número de neurônios ativos que difira de N. Por fim, o quarto termo mede o comprimento do percurso realizado pelo caixeiro viajante, sendo que o que se busca é minimizar este comprimento de percurso, mas sem violar o atendimento das restrições do problema.

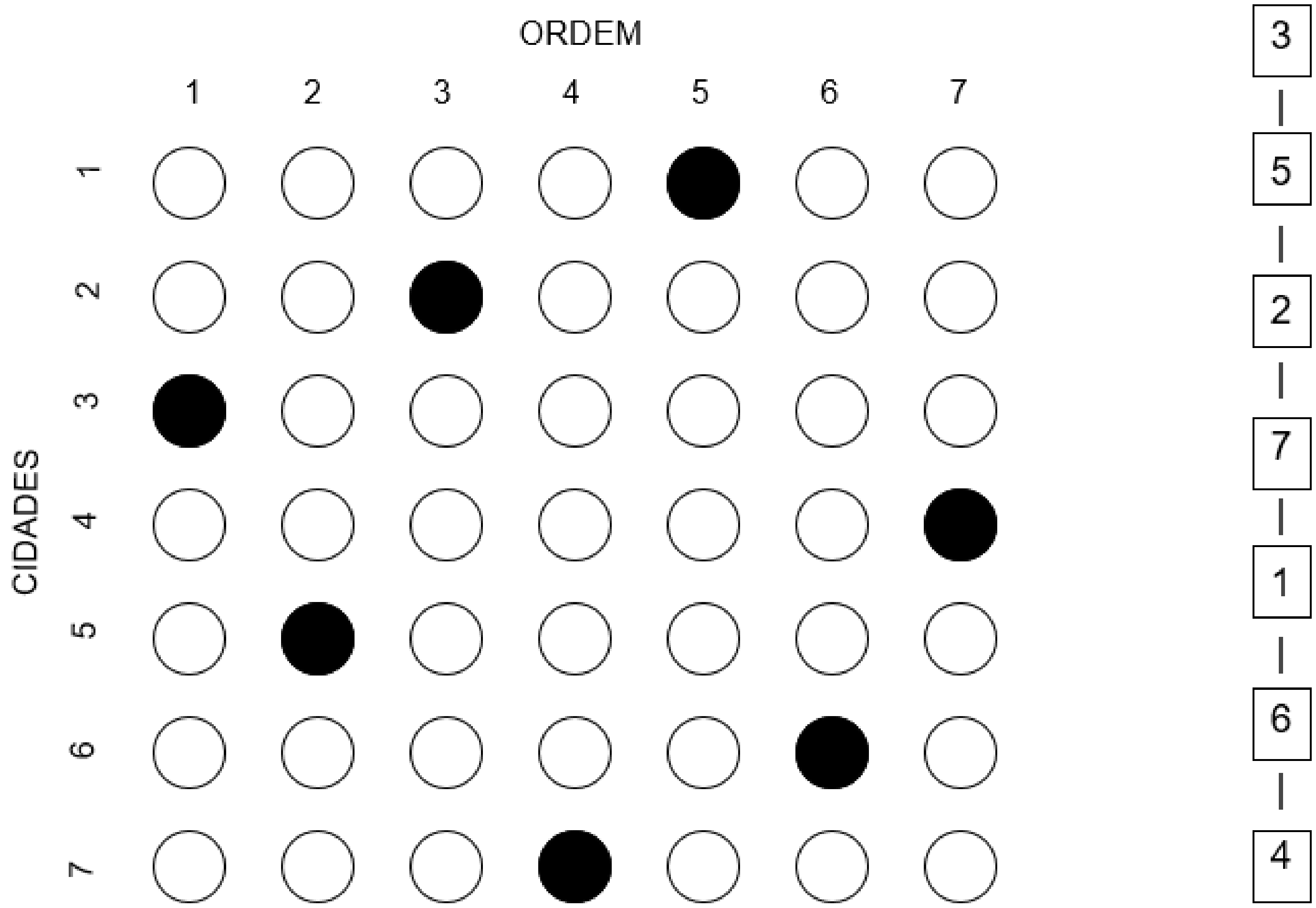


Figura 5 – Interpretação do ponto de equilíbrio como uma solução do problema do caixeiro viajante (há um e somente um neurônio ativo por linha e por coluna)

- A rede de Hopfield que leva a resultados como os da Figura 5 é derivada diretamente da superfície de energia, conduzindo ao sistema dinâmico dado na forma (HOPFIELD & TANK, 1985):

$$\frac{du_{Xi}}{dt} = -u_{Xi} - A\sum_{j \neq i} y_{Xj} - B\sum_{Y \neq X} y_{Yi} - C\left(\sum_{Y}\sum_{j} y_{Yj} - N\right) - D\sum_{Y} d_{XY}\left(y_{Y(i+1)} + y_{Y(i-1)}\right)$$

$$y_{Xi} = g(u_{Xi}) = \frac{1}{2}\left(1 + \tanh\left(\frac{u_{Xi}}{u_0}\right)\right)$$

para todo X e i e tomando $\tau_{Xi} = 1$.

- Para se chegar a este sistema dinâmico, os pesos da rede de Hopfield têm que ser tais que:

$$w_{Xi,Yj} = -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - C - Dd_{XY}\left(\delta_{j(i+1)} + \delta_{j(i-1)}\right)$$

onde δ_{ab} é igual a 1 quando $a = b$ e é igual a 0 quando $a \neq b$, e é preciso impor que $\theta_{Xi} = -CN$.

- Uma implementação computacional do sistema dinâmico de tempo contínuo apresentado na página anterior requer algum mecanismo de discretização no tempo, ilustrado aqui pelo método de Euler:

$$u_{Xi}(t+\Delta t) = u_{Xi}(t) + \Delta t \begin{bmatrix} -u_{Xi}(t) - A\sum_{j \neq i} y_{Xj}(t) - B\sum_{Y \neq X} y_{Yi}(t) - \dots \\ \dots - C\left(\sum_{Y}\sum_{j} y_{Yj}(t) - N\right) - D\sum_{Y} d_{XY}\left(y_{Y(i+1)}(t) + y_{Y(i-1)}(t)\right) \end{bmatrix}$$

- Infelizmente, não é uma tarefa elementar obter os coeficientes *A*, *B*, *C*, e *D*, sendo que uma escolha inadequada pode fazer com que soluções de boa qualidade, incluindo a solução ótima, deixem de ser mínimos locais da função de energia.
- Além do desempenho da rede de Hopfield na solução do problema do caixeiro viajante não ser correntemente tido como superior ao produzido por outras técnicas de solução já disponíveis, a extensão desta abordagem para outros problemas de natureza combinatória, embora possível, não é imediata.

- Na verdade, o potencial de aplicação de sistemas dinâmicos não-lineares em problemas de explosão combinatória é alto e continua a ser explorado na literatura, embora a complexidade envolvida no processo de mapeamento do problema em uma superfície de energia do sistema dinâmico associado crie desafios para um uso mais amplo desta ferramenta de solução.
- A implementação de um hardware dedicado pode ser considerada uma das mais promissoras frentes de aplicação, pois amplia a escala de problemas combinatórios que podem ser abordados, sem produzir incrementos significativos no tempo de processamento, visto que se emprega computação paralela.
- Para ilustrar o desempenho da rede de Hopfield, considere o código em Matlab apresentado a seguir. Ao ser executado para um caso de estudo que contempla um problema de caixeiro viajante contendo 10 cidades, obtém-se resultados promissores em poucos segundos, incluindo aqueles que estão ilustrados nas figuras apresentadas mais adiante.

- As coordenadas das 10 cidades podem ser encontradas no código e os parâmetros adotados foram os seguintes:
  - ✓ Δt = 0.0001
  - ✓ A = 500
  - ✓ B = 500
  - ✓ C = 1000
  - ✓ D = 500
  - ✓ u_0 = 0.02
  - ✓ Iterações até a convergência = 1000
- A função de ativação dos neurônios continua sendo $y_{Xi}(t) = g(u_{Xi}(t)) = \frac{1}{2}\left(1 + \tanh\left(\frac{u_{Xi}(t)}{u_0}\right)\right)$, mas resolveu-se adotar um passo adicional para acelerar a convergência da rede neural, produzindo:
  - ✓ Se $y_{Xi}(t) < 0.3$, então $y_{Xi}(t) = 0$.
  - ✓ Se $y_{Xi}(t) > 0.7$, então $y_{Xi}(t) = 1$.
- O programa pode fazer várias tentativas até obter uma solução válida.

```
% FEEC/Unicamp
% Hopfield neural network for TSP - Matlab code
%
clear all;
close all;
% Coordinates of the cities plus displacements of the labels
displace1 = 0.05;
displace2 = 0.03;
cities = [
    0.2439 0.1463 0 -displace1
    0.8488 0.3609 displace2 0
    0.6683 0.2536 displace2 -displace1
    0.6878 0.5219 displace2 0
    0.1707 0.2293 -displace1 0
    0.2293 0.7610 -displace1 0
    0.4000 0.4439 -displace1 0
    0.8732 0.6536 displace2 0
    0.5171 0.9414 displace2 0
    0.6195 0.2634 -displace1 -displace1];
n_cities = length(cities(:,1));
delta = 0.0001;
A = 500;
B = 500;
C = 1000;
D = 500;
u0 = 0.02;
for i=1:n_cities,
    for j=1:n_cities,
        d(i,j) = sqrt((cities(i,1)-cities(j,1))^2+(cities(i,2)-
cities(j,2))^2);
    end
end
fault = 1;
n_attempt = 0;
while fault == 1,
    n_attempt = n_attempt + 1;
    y = rand(n_cities,n_cities);
    u = atanh(2*y-1)*u0;
    for t=1:1000;
        for i=1:n_cities,
            for j=1:n_cities,
                sum1 = 0;
                for k=1:n_cities,
                    if k ~= j,
                        sum1 = sum1 + y(i,k);
                    end
                end
                sum2 = 0;
                for k=1:n_cities,
                    if k ~= i,
                        sum2 = sum2 + y(k,j);
                    end
                end
                sum3 = 0;
                for k=1:n_cities,
                    for p=1:n_cities,
                        sum3 = sum3 + y(k,p);
                    end
                end
                sum3 = sum3 - n_cities;
                sum4 = 0;
                for k=1:n_cities,
                    if j == 1,
                        sum4 = sum4 + d(i,k)*(y(k,j+1)+y(k,n_cities));
                    elseif j == n_cities,
                        sum4 = sum4 + d(i,k)*(y(k,1)+y(k,j-1));
                    else
                        sum4 = sum4 + d(i,k)*(y(k,j+1)+y(k,j-1));
                    end
                end
                du(i,j) = -u(i,j) - A*sum1 - B*sum2 - C*sum3 - D*sum4;
            end
        end
        u = u + delta*du;
        y = 0.5.*(1+tanh(u./u0));
        for i=1:n_cities,
            for j=1:n_cities,
                if y(i,j) < 0.3,
                    y(i,j) = 0;
                elseif y(i,j) > 0.7,
                    y(i,j) = 1;
                end
            end
        end
    end
    disp(sprintf('Attempt %d',n_attempt));
    fault1 = n_cities;
    for i=1:n_cities,
        for j=1:n_cities,
            fault1 = fault1 - y(i,j);
        end
    end
    fault2 = 0;
    for i=1:n_cities,
        for j=1:(n_cities-1),
            for k=(j+1):n_cities,
                fault2 = fault2 + y(i,j)*y(i,k);
            end
        end
    end
    fault3 = 0;
    for i=1:n_cities,
        for j=1:(n_cities-1),
            for k=(j+1):n_cities,
                fault3 = fault3 + y(j,i)*y(k,i);
            end
        end
    end
    if fault1 == 0 && fault2 == 0 && fault3 == 0,
        fault = 0;
    end
end
seq_cities = [];
for j=1:n_cities;
    [val,pos] = max(y(:,j));
    seq_cities = [seq_cities;pos];
end
seq_cities'
gen_traj(cities,seq_cities,1);
```

- O programa principal chama o procedimento a seguir, responsável por plotar a trajetória vinculada à solução encontrada.

```
% FEEC/Unicamp
% Generate the graphical solution for TSP
%
function [] = gen_traj(cities,seq_cities,nfig)
n_cities1 = length(cities(:,1));
n_cities2 = length(seq_cities);
L = sprintf('%c','A':'Z');
figure(nfig);
for i=1:n_cities1,
    plot(cities(i,1),cities(i,2),'ko');hold on;
    text(cities(i,1)+cities(i,3),cities(i,2)+cities(i,4),L(i));
    if i < n_cities2,

plot([cities(seq_cities(i),1);cities(seq_cities(i+1),1)],[cities(seq_cities(i),2);cities(seq_cities(i+1),2)]);
    elseif i == n_cities2,

plot([cities(seq_cities(i),1);cities(seq_cities(1),1)],[cities(seq_cities(i),2);cities(seq_cities(1),2)]);
    end
end
axis([0 1 0 1]);
hold off;
tot_dist = 0;
for i=1:n_cities2,
    if i < n_cities2,
        tot_dist = tot_dist + sqrt((cities(seq_cities(i),1)-
cities(seq_cities(i+1),1))^2+(cities(seq_cities(i),2)-cities(seq_cities(i+1),2))^2);
    else
        tot_dist = tot_dist + sqrt((cities(seq_cities(i),1)-cities(seq_cities(1),1))^2+(cities(seq_cities(i),2)-
cities(seq_cities(1),2))^2);
    end
end
title(sprintf('Total distance = %g',tot_dist));
```

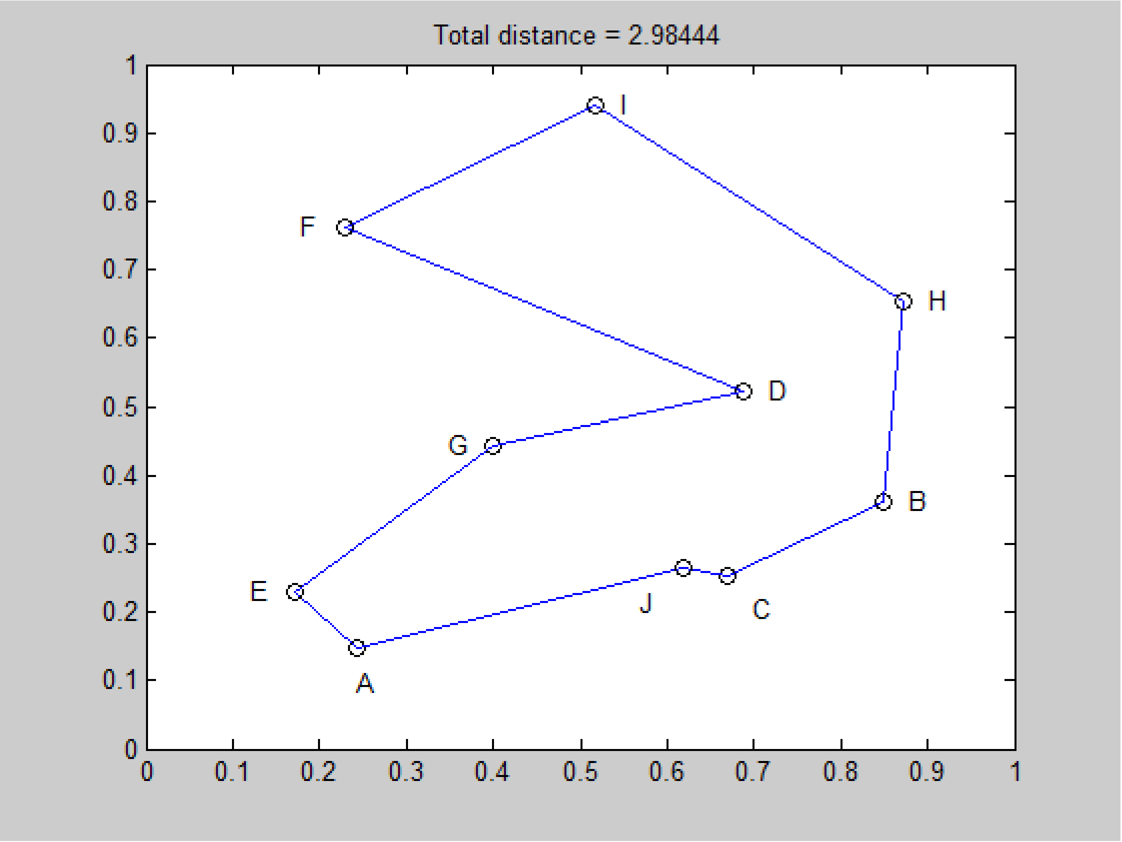


Figura 6 – Uma primeira proposta de solução obtida ao executar o programa em Matlab

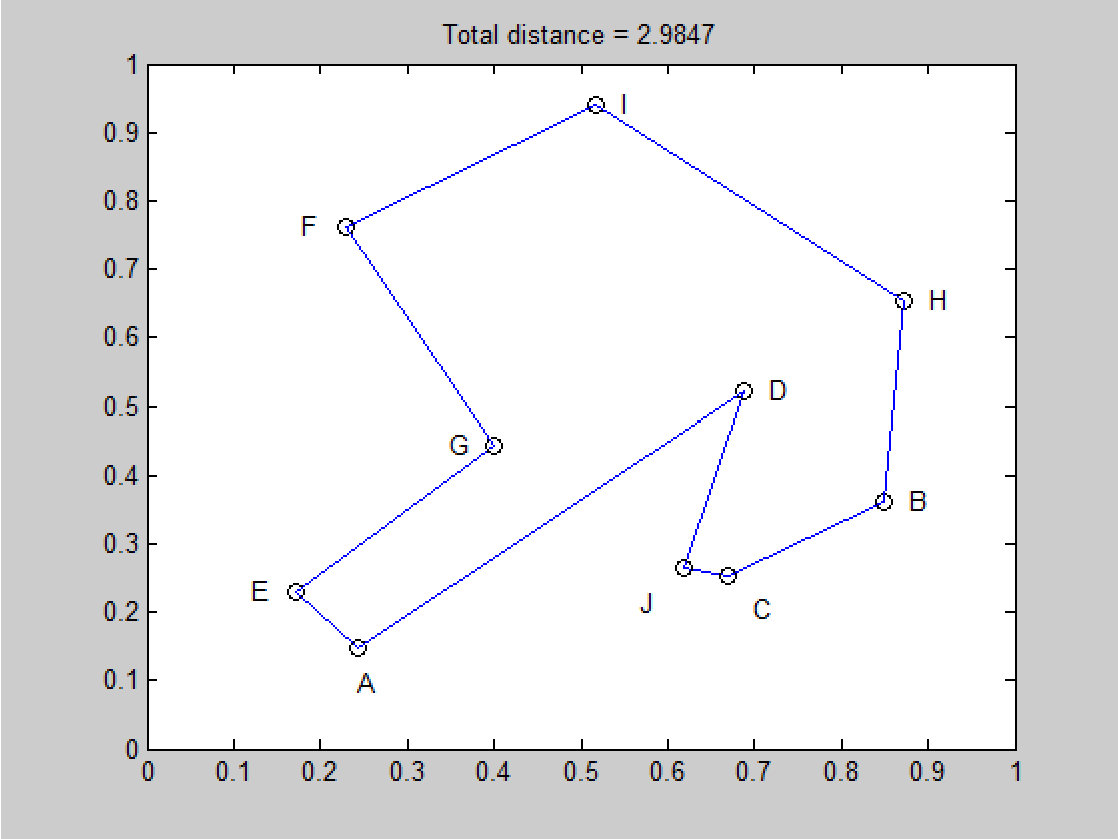


Figura 7 – Uma segunda proposta de solução obtida ao executar novamente o programa em Matlab

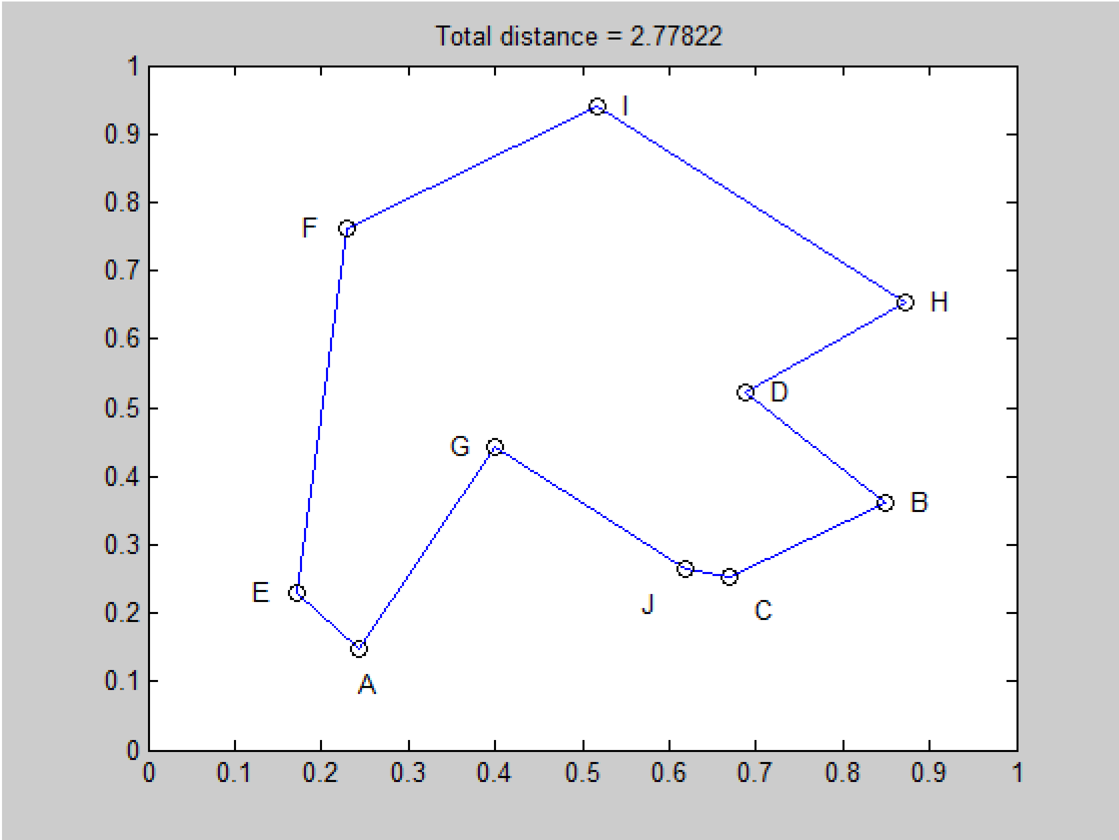


Figura 8 – Uma terceira proposta de solução obtida ao executar novamente o programa em Matlab

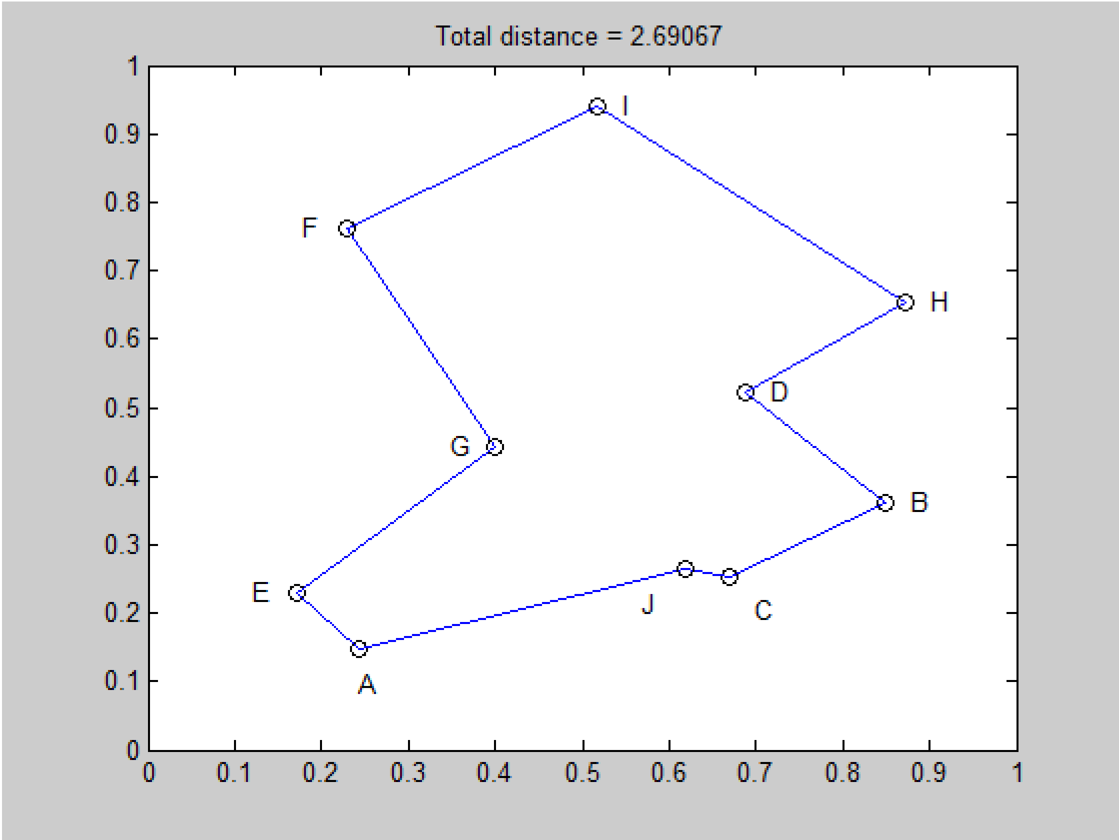


Figura 9 – Uma quarta proposta de solução obtida ao executar novamente o programa em Matlab (esta é a solução ótima do problema)

12 Solução de problemas de programação matemática

- Já foi dito que as redes neurais recorrentes podem ser implementadas em hardware específico, inerentemente paralelo e distribuído, o que torna atraente a sua aplicação a problemas de otimização de larga escala, possivelmente envolvendo restrições de tempo real.
- A primeira iniciativa de se empregar um computador analógico para se resolver um problema de programação linear pode ser atribuída a PYNE (1956). E após HOPFIELD & TANK (1985), houve muitas outras iniciativas voltadas para o tratamento de problemas de otimização a partir da especificação de funções de energia a serem minimizadas.
- KENNEDY & CHUA (1988) propuseram redes neurais recorrentes para a solução de problemas de programação não-linear, que incluem a formulação de HOPFIELD & TANK (1985) como um caso particular. No entanto, a presença de termos de penalidade permitia apenas a obtenção de soluções aproximadas.

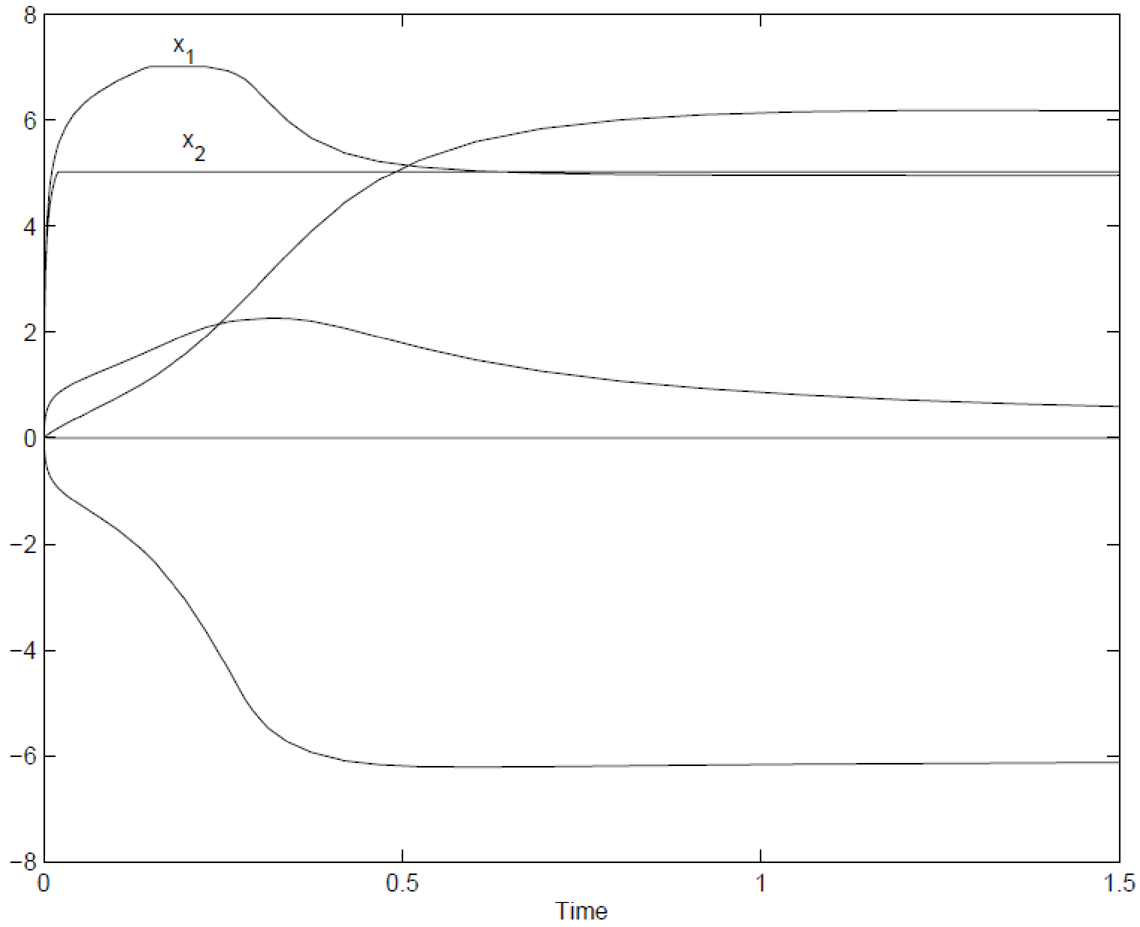
- Outras abordagens se sucederam, geralmente fundamentadas em termos de penalidade ou formulações lagrangeanas. Consulte CICHOCKI & UNBEHAUEN (1993) e LILLO *et al.* (1993) para uma revisão do estado-da-arte à época.
- Fazendo uso de formulações primais-duais e métodos de projeção, XIA & WANG (2001) apresentam uma revisão do uso bem-sucedido de redes neurais recorrentes na solução de problemas de programação quadrática e de programação linear.
- Como um exemplo (XIA & WANG, 2001), considere o problema quadrático a seguir:

$$
\begin{array}{ll}
\text{Minimize} & x_1^2 + x_2^2 + x_1 x_2 - 30x_1 - 30x_2 \\
\text{subject to} & \dfrac{5}{12}x_1 - x_2 \le \dfrac{35}{12}, \\
& \dfrac{5}{2}x_1 + x_2 \le \dfrac{35}{2}, \\
& -x_1 \le 5, \\
& x_2 \le 5.
\end{array}
$$

que pode ser reformulado como segue:

$$
\begin{array}{ll}
\text{Minimize} & x_1^2 + x_2^2 + x_1x_2 - 30x_1 - 30x_2 \\
\text{subject to} & \dfrac{5}{12}x_1 - x_2 + x_3 = \dfrac{35}{12}, \\
& \dfrac{5}{2}x_1 + x_2 + x_4 = \dfrac{35}{2}, \\
& -5 \le x_1 \le 7, -5 \le x_2 \le 5, 0 \le x_3 \le 10, 0 \le x_4 \le 35.
\end{array}
$$

- Este problema tem solução única $(x_1,x_2) = (5,5)$, e sua solução empregando uma rede neural recorrente produz como resultado o gráfico a seguir, na formulação primal-dual. Repare que, após um transitório, ocorre uma convergência para a solução desejada.



- Como trabalhos mais recentes na área, tem-se DA SILVA *et al.* (2006), LEUNG *et al.* (2003) e WEN *et al.* (2009).

13 Referências

AMIT, D.J. *Modeling Brain Function: The World of Attractor Neural Networks*. Cambridge University Press, 1989.

BHAYA, A., KASZKUREWICZ, E. & KOZYAKIN, V.S. Existence and Stability of a Unique Equilibrium in Continuous-Valued Discrete-Time Asynchronous Hopfield Neural Networks. *IEEE Transactions on Neural Networks*, vol. 7, no. 3, pp. 620-628, 1996.

BRUCK, J. On the convergence properties of the Hopfield model. *Proceedings of the IEEE*, vol. 78, pp. 1579-1585, 1990.

CARPENTER, G.A., COHEN, M.A. & GROSSBERG, A. Technical comments on "Computing with neural networks." *Science*, vol. 235, pp. 1226-1227, 1987.

CICHOCKI, A. & UNBEHAUEN, R. *Neural Networks for Optimization and Signal Processing*, John Wiley, 1993.

COHEN, M.A. & GROSSBERG, S. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13, pp. 815-826, 1983.

DA SILVA, I.N., AMARAL, W.C., ARRUDA, L.V.R. Neural approach for solving several types of optimization problems, *Journal of Optimization Theory and Applications*, vol. 128, pp. 563-580, 2006.

FUNAHASHI, K.-I., NAKAMURA, Y. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, vol. 6, no. 5, pp. 801-806, 1993.

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

HERKEN, R. (ed.) *The Universal Turing Machine: A Half-Century Survey*, Springer, 1995.

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.

HOPFIELD, J.J. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences of the U.S.A.*, vol. 81, pp. 3088-3092, 1984.

HOPFIELD, J.J. & TANK, D.W. 'Neural' computation of decisions in optimization problems. *Biological Cybernetics*, vol. 52, pp. 141-152, 1985.

HOPFIELD, J.J. & TANK, D.W. Computing with neural circuits: A model. *Science*, vol. 233, pp. 625-633, 1986.

KENNEDY, M.D. & CHUA, L.N. Neural networks for nonlinear programming, *IEEE Transactions on Circuits and Systems*, vol. 35, no. 5, pp. 554-562, 1988.

KHALIL, H.K. *Nonlinear Systems*. 2nd. edition, Prentice Hall, 1996.

LEUNG, Y., CHEN, K. & GAO, X. A high-performance feedback neural network for solving convex nonlinear programming problems, *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1469–1477, 2003.

LILLO, W.E., LOH, M.H., HUI, S. & ZAK, S.H. On solving constrained optimization problems with neural networks: A penalty method approach, *IEEE Transactions on Neural Networks*, vol. 4, no. 6, pp. 931-940, 1993.

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.

MEISS, J.D. Frequently Asked Questions about Nonlinear Science. Department of Applied Mathematics, University of Colorado at Boulder, http://amath.colorado.edu/faculty/jdm/faq-Contents.html.

MORITA, M. Associative memory with nonmonotonic dynamics. *Neural Networks*, vol. 6, pp. 115-126, 1993.

PYNE, I.B. Linear programming on an electronic analogue computer. *Transactions of the American Institute of Electrical Engineering*, vol. 75, pp. 139, 1956.

WEN, U.-P., LAN, K.-M. & SHIH, H.-S. A review of Hopfield neural networks for solving mathematical programming problems, *European Journal of Operational Research*, vol. 198, pp. 675-687, 2009.

XIA, Y. & WANG, J. Recurrent neural networks for optimization: The state of the art, in Medsker, L.R. & Jain, L.C. (eds.) Recurrent Neural Networks: Design and Applications, Chapter 2, pp. 23-55, CRC Press, 2001.