Principal Components Analysis - PCA

RNAs para Decomposição de um Espaço Vetorial em Sub-Espaços

- Nos capítulos anteriores estudamos detalhadamente dois tipos de RNAs que apresentam a habilidade de aprender a partir de seu ambiente e, a partir do processo de **aprendizado supervisionado**, melhorar seu desempenho (as RNAs MLPs treinadas pelo algoritmo *backpropagation* e as RNAs RBFs).
- Neste capítulo estudaremos um tipo de RNA que é submetido a um processo de **aprendizado não-supervisionado** (ou auto-organizado).
- **O propósito de um algoritmo de aprendizado auto-organizado é descobrir padrões significativos ou características nos dados de entrada da RNA, e fazê-lo sem a presença de um tutor (ou seja, sem a presença de um conjunto de alvos de interesse, providos por um tutor externo).**
- Para atingir tal propósito, o algoritmo é provido de um conjunto de regras de natureza local, conjunto este que o habilita a aprender a computar um mapeamento entrada-saída, com específicas propriedades desejáveis.
- O termo “local” significa, neste contexto, que uma mudança aplicada ao peso sináptico de um neurônio é confinada à vizinhança imediata daquele neurônio.

- O modelamento de estruturas de RNAs usadas para **aprendizado auto-organizado tende muito mais a seguir as estruturas neuro-biológicas** do que as estruturas utilizadas para o aprendizado supervisionado.

- A heurística para aprendizado auto-organizado de RNAs que estudaremos neste capítulo é chamada "**Algoritmo Hebbiano Generalizado**" e é utilizada para proceder à Análise dos Componentes Principais (*Principal Components Analysis* - PCA) ou Decomposição em Sub-Espaços (DSE) de um conjunto da dados de interesse.
- O Algoritmo Hebbiano Generalizado foi proposto por Sanger em 1989 e combina a ortonormalização de Gram-Schmidt ao modelo de um único neurônio linear introduzido por Oja em 1982.

- A Análise dos Componentes Principais de um conjunto de dados é uma técnica padrão comumente utilizada para redução da dimensionalidade de dados, em processamento de sinais.

(1) Princípios que regem o aprendizado auto-organizado.

(2) Transformada Karhunen-Loève.

(3) RNAs utilizadas para PCA (= DSE de um espaço vetorial).

Princípios do Aprendizado Auto-Organizado

➔ O aprendizado não-supervisionado (ou auto-organizado) consiste do ajuste recorrente dos parâmetros livres de uma RNA em resposta a padrões de ativação, e de acordo com regras prescritas, até que seja desenvolvida uma desejada configuração final.

➔ Ordem global pode surgir a partir de interações locais.

- Esta afirmativa é tanto válida para o cérebro, quanto no contexto de redes neurais artificiais.

- Muitas interações locais entre neurônios vizinhos de uma rede podem "coalescer" (ou, em outra interpretação alegórica, "cristalizar") em estados de ordem global e conduzir a um comportamento coerente na forma de padrões espaciais ou temporais, o que é a essência da auto-organização.

A organização ocorre em dois diferentes níveis, que interagem entre si na forma de um elo de realimentação. Estes dois níveis são:

- Atividade ⇨ Certos padrões de atividade são produzidos por uma dada rede em resposta a sinais de entrada.
- Conectividade ⇨ Os pesos sinápticos da rede são modificados em resposta a sinais neurais nos padrões de atividade, devido à plasticidade sináptica.

Princípios da auto-organização:

1. Ajustes recorrentes em pesos sinápticos tendem a se auto-amplificar.

- Uma forte sinapse conduz à coincidência de sinais pré e pós-sinápticos.
- A sinapse é aumentada em força, por tal coincidência.

Este mecanismo nada mais é do que o postulado de aprendizado de Hebb, que diz: *"Se dois neurônios em cada lado de uma sinapse são ativados simultaneamente, então a força daquela sinapse é seletivamente aumentada."*

2. Limitação de recursos conduz à competição entre as sinapses e, consequentemente, à seleção das sinapses que crescem de forma mais vigorosa (portanto, as mais adequadas) às custas de outras.

- Para que o sistema possa ser estabilizado precisa haver alguma forma de competição por recursos limitados.
- Especificamente, um aumento na transmitância de alguma sinapse na rede deve ser compensado pelo decréscimo em outras.
- Assim, apenas as sinapses que obtêm sucesso podem aumentar sua transmitância, enquanto as que obtêm menos sucesso tendem a reduzir a sua transmitância e podem eventualmente desaparecer (transmitância zero) do contexto operacional da RNA .

3. Ajustes em pesos sinápticos tendem a cooperar.

- A presença de uma sinapse "vigorosa" pode aumentar a transmitância de outras sinapses, apesar da competição global na rede.

4. Ordem e estrutura nos padrões de ativação representam a informação redundante que é adquirida pela rede neural na forma de conhecimento, a qual é pré-requisito necessário ao aprendizado auto-organizado.

- Para que o aprendizado auto-organizado possa desempenhar uma função útil de processamento de informação, é necessário que haja redundância nos padrões de ativação supridos à rede pelo ambiente.

A Transformação Karhunen-Lòeve para PCA
ou decomposição em sub-espaços de um espaço vetorial

➔ A Transformação Karhunen-Loève (KLT) projeta um conjunto $\mathbf{X}$ de vetores de dados $\underline{x} \in \Re^M$ sobre uma base ortonormal em $\Re^M$ formada pelo conjunto dos M auto-vetores $\underline{e}_m \in \Re^M$, $m = 0, 1, \cdots, M-1$, da matriz de covariância de $\mathbf{X}$.

➔ A KLT é tal que a base será orientada de acordo com as direções de maior variância de $\mathbf{X}$ em $\Re^M$.

➔ A m-ésima projeção de $\mathbf{X}$ sobre a direção do auto-vetor $\underline{e}_m$ é chamada de ***m*-ésimo sub-espaço** (ou m-ésimo componente principal).

➔ O m-ésimo auto-valor λ_m associado ao auto-vetor $\underline{e}_m$ corresponde à **variância do *m*-ésimo sub-espaço** de $\mathbf{X}$.

➔ Ainda, a variância de cada sub-espaço é um máximo local, no universo de todas as variâncias resultantes da projeção de $\mathbf{X}$ sobre todas as possíveis direções em $\Re^M$.

➔ Embora qualquer espaço de dimensão menor do que $\Re^M$ seja um sub-espaço de $\Re^M$, este estudo refere-se muitas vezes a um sub-espaço de $\mathbf{X}$ como o **conjunto dos vetores de $\mathbf{X}$ que concentram-se de forma alinhada ao longo de uma particular direção em $\Re^M$**.

➔ Talvez a mais importante utilização da **Transformação Karhunen-Loève**, também conhecida por **Análise dos Componentes Principais** (PCA), seja a redução dimensional do conjunto $\mathbf{X}$.

➔ Esta característica torna a KLT bastante popular em processamento digital de sinais por permitir que as informações mais significativas contidas em um sinal possam ser representadas, mediante a introdução de algum erro considerado aceitável, em um espaço de dados de menores dimensões do que a dimensão original $\Re^M$.

➔ Como cada sub-espaço ou componente principal é orientado de acordo com as direções de maior variância de $\mathbf{X}$ em $\Re^M$, os sub-espaços de menor variância podem ser descartados – o que resultará em uma redução dimensional ótima no sentido do Erro Médio Quadrático (MSE - *Mean Square Error*).

➔ O desenvolvimento do método para aplicação da KLT à $\mathbf{X}$ apresentado neste estudo segue a proposta de Haykin em *Neural Networks*.

Interpretação Geométrica da KLT

- A decomposição em sub-espaços gerada pela KLT consiste em obter o conjunto de M auto-vetores e auto-valores da matriz de covariância $\mathbf{C}$ de um conjunto $\mathbf{U}$ de média zero, composto por L vetores de dados $\underline{u}_i \in \Re^M, i = 0,1,\cdots,L-1$. A matriz de covariância $\mathbf{C}$ também é referida como a matriz de correlação $\mathbf{C}$.

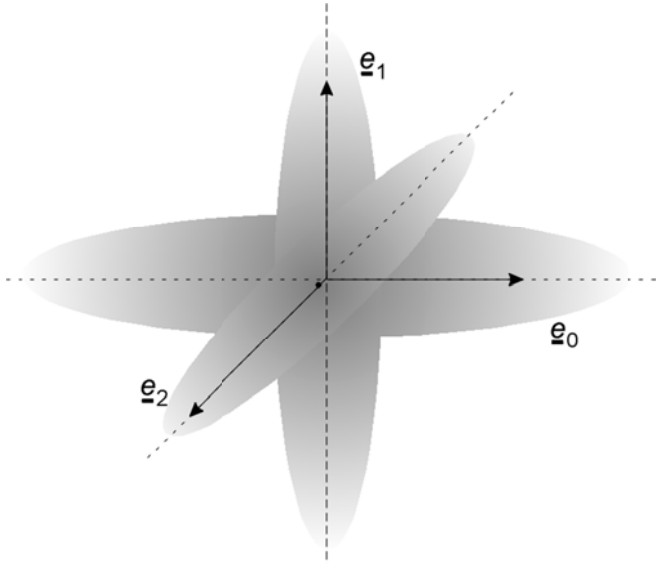
  - A restrição de que o conjunto $\mathbf{U}$ apresente média vetorial $\underline{0} \in \Re^M$ é transparente a nível de procedimento, já que o vetor média pode ser restaurado ao final.

- O conjunto de M auto-vetores $\underline{e}_m \in \Re^M$ obtido pela KLT, $m = 0,1,\cdots,M-1$, define uma base de vetores ortonormais em $\Re^M$ e, portanto, define um conjunto de M eixos ortogonais Cartesianos.

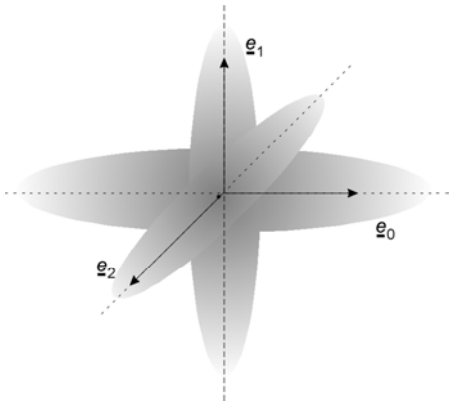
  - A coordenada de origem deste sistema Cartesiano é a coordenada de origem dos vetores da base ortonormal definida pelo conjunto de M auto-vetores $\underline{e}_m$.

  - Como a média do conjunto $\mathbf{U}$ é assumida zero, a coordenada de origem do sistema Cartesiano é $\underline{0} \in \Re^M$.

- A título de interpretação da KLT, para obter a KLT através de um processo manual e experimental, toma-se um vetor arbitrário de módulo unitário $\underline{e} \in \Re^M$ com origem em $\underline{0} \in \Re^M$, o qual define uma direção arbitrária sobre a qual o conjunto $\mathbf{U}$ de vetores $\underline{u}_i \in \Re^M$ será projetado. Note que obter a projeção ρ de um vetor $\underline{u} \in \Re^M$ do conjunto $\mathbf{U}$ sobre o vetor unitário $\underline{e} \in \Re^M$ significa efetuar o produto escalar entre eles, i.e., $\rho = \underline{u}^T \cdot \underline{e}$. A seguir procede-se da seguinte maneira:
  - Projetar a totalidade do conjunto $\mathbf{U}$ sobre a direção dada por $\underline{e}$ e medir a variância λ da projeção (variância = média da soma dos quadrados das projeções dos vetores $\underline{u}_i \in \Re^M$ do conjunto $\mathbf{U}$ sobre a direção dada por $\underline{e}$).
  - Após tentar todas as direções possíveis no espaço $\Re^M$, haverá uma direção dada por $\underline{e}$ na qual é obtida a maior variância λ_0.
  - O vetor $\underline{e}$ que define tal direção é igual ao auto-vetor $\underline{e}_0$ associado ao maior auto-valor, obtidos pela KLT, com valor do maior auto-valor dado por λ_0.



- O processo é repetido novamente para a obtenção do segundo maior auto-valor λ_1, com a restrição de que a busca da direção de maior variância em $\Re^M$ seja feita em direções ortogonais à do auto-vetor $\underline{e}_0$ associado ao maior auto-valor λ_0, recém determinados.
- A busca da direção de maior variância em $\Re^M$ para obtenção do terceiro maior auto-valor λ_2 é feita com a restrição de que as direções testadas sejam ortogonais às direções dos dois auto-vetores $\underline{e}_0$ e $\underline{e}_1$ associados aos maiores auto-valores λ_0 e λ_1 previamente encontrados.
- E assim prosseguiríamos neste processo recursivo até que os M auto-valores e auto-vetores fossem determinados.



Assim, como os sub-espaços obtidos pela KLT estão alinhados com as direções ortogonais de maior variância possível no espaço original $\Re^M$, a KLT é considerada uma transformação ótima no sentido do erro médio quadrático MSE para efeito de reconstrução do espaço original $\Re^M$ a partir de suas M projeções ou componentes principais.

Ou seja, o conjunto de M sub-espaços representa de maneira ótima, no sentido do MSE, o conjunto $\mathbf{U}$ de L vetores $\underline{u}_i \in \Re^M$, $i = 0,1,\cdots,L-1$.

Os sub-espaços obtidos pela KLT encontram-se alinhados com os eixos Cartesianos que definem as direções de maior variância possíveis no espaço original $\Re^M$.

Isto resulta em uma maior concentração de pontos definidos pelos vetores $\underline{u}_i \in \Re^M$ do espaço original nas vizinhanças dos eixos Cartesianos que definem cada sub-espaço.

Como o *m*-ésimo eixo Cartesiano define a *m*-ésima região em $\Re^M$ de maior variância λ_m, aqueles vetores cuja média do quadrado de suas normas Euclidianas é próxima ao valor λ_m (λ_m é a média do quadrado das normas Euclidianas das projeções destes vetores sobre o *m*-ésimo eixo) caracterizarão um sub-conjunto de vetores do espaço $\Re^M$ aproximadamente alinhados com o *m*-ésimo eixo.

- Parte destes vetores estará aproximadamente congruente (*i.e.*, alinhados no mesmo sentido) com o *m*-ésimo semi-eixo positivo, e parte dos vetores estará aproximadamente congruente com o *m*-ésimo semi-eixo negativo.
- Mas, independentemente do sentido positivo ou negativo, a média do quadrado da norma Euclidiana destes vetores será, em maior ou menor grau, próxima ao valor λ_m, grau que depende do quanto os vetores alinham-se com o *m*-ésimo eixo Cartesiano.
- Adicionalmente, a média dos vetores que são congruentes com o semi-eixo negativo tende a ser igual à média dos vetores que são congruentes com o semi-eixo positivo, já que $\mathbf{U}$ apresenta média vetorial $\underline{0} \in \Re^M$.
- Assim, deve-se esperar um certo equilíbrio entre os vetores alinhados com cada um dos dois semi-eixos.

Portanto, o *m*-ésimo sub-espaço identifica uma nuvem de pontos em $\Re^M$ nas vizinhanças do *m*-ésimo eixo Cartesiano, com coordenada de cada ponto definida pelo respectivo vetor $\underline{u}_i \in \Re^M$ do conjunto $\mathbf{U}$ nas vizinhanças do *m*-ésimo eixo.

Como a média do quadrado da norma Euclidiana dos vetores associados a estes pontos tende em maior ou menor grau para λ_m, a norma – ou distância – Euclidiana média destes pontos à origem aproxima-se do valor $\sqrt{\lambda_m}$.

Exemplo 1 – a KLT obtida através dos autovalores e autovalores da matriz de covariância do conjunto de vetores U:

Seja o conjunto $\mathbf{U}$ de $L = 6$ vetores $\underline{u}_i \in \Re^2$, $i = 0,1,\cdots,L-1$, de média zero definido por

$$\mathbf{U} = \left\{ \begin{bmatrix} 0.425 \\ -1.063 \end{bmatrix}, \begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}, \begin{bmatrix} 0.327 \\ -0.843 \end{bmatrix}, \begin{bmatrix} -0.497 \\ 0.820 \end{bmatrix}, \begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}, \begin{bmatrix} -0.359 \\ 0.897 \end{bmatrix} \right\}.$$

a) Plote os auto-vetores $\underline{e}_m$ da matriz $\mathbf{C}$ de covariância de $\mathbf{U}$ conjuntamente com os vetores de $\mathbf{U}$.

Para facilitar a visualização, escalone os auto-vetores $\underline{e}_m$ pela raiz dos respectivos auto-valores λ_m,

i.e., $\underline{\psi}_m = \underline{e}_m \cdot \sqrt{\lambda_m}$, $m = 0,1$.

b) Obtenha as projeções de $\mathbf{U}$ sobre os eixos Cartesianos definidos pelos auto-vetores de $\mathbf{C}$.

Solução:

$$\mathbf{C} = \frac{1}{L}\sum_{i=0}^{L-1} \underline{x}_i \cdot \underline{x}_i^{\,T} = \frac{1}{6}\sum_{i=0}^{5} \underline{x}_i \cdot \underline{x}_i^{\,T} =$$

$$= \frac{1}{6}\left\{ \begin{bmatrix} 0.425 \\ -1.063 \end{bmatrix}\begin{bmatrix} 0.425 \\ -1.063 \end{bmatrix}^T + \begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}\begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}^T + \begin{bmatrix} 0.327 \\ -0.843 \end{bmatrix}\begin{bmatrix} 0.327 \\ -0.843 \end{bmatrix}^T + \right.$$

$$\left. + \begin{bmatrix} -0.497 \\ 0.820 \end{bmatrix}\begin{bmatrix} -0.497 \\ 0.820 \end{bmatrix}^T + \begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}\begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}^T + \begin{bmatrix} -0.359 \\ 0.897 \end{bmatrix}\begin{bmatrix} -0.359 \\ 0.897 \end{bmatrix}^T \right\} = \begin{bmatrix} 0.21 & -0.429 \\ -0.429 & 0.915 \end{bmatrix}$$

De (6.32) em http://www.fccdecastro.com.br/pdf/RNA_C6.pdf, temos $\mathbf{C}\,\underline{e}_m = \lambda_m\,\underline{e}_m$ sendo $m = 0,1,\cdots,M-1$, com $M = 2$ ($M = 2$, porque $\underline{u}_i \in \Re^2$).

A solução de $\mathbf{C}\,\underline{e}_m = \lambda_m\,\underline{e}_m$ com $\mathbf{C} = \begin{bmatrix} 0.21 & -0.429 \\ -0.429 & 0.915 \end{bmatrix}$ consiste na determinação dos M auto-valores λ_m

e dos M auto-vetores $\underline{e}_m$ que satisfazem esta equação.

Utilizando as funções $\mathrm{eigenvals}(\cdot)$ e $\mathrm{eigenvecs}(\cdot)$ do aplicativo MathCad da MathSoft Inc., obtemos :

Auto-valores de **C**:

$$\lambda_0 = 1.118 \text{ e } \lambda_1 = 7.035 \times 10^{-3}$$

Auto-vetores de **C** associados:

$$\underline{e}_0 = \begin{bmatrix} -0.427 \\ 0.904 \end{bmatrix} \text{ e } \underline{e}_1 = \begin{bmatrix} 0.904 \\ 0.427 \end{bmatrix}.$$

Nota: Qualquer outro aplicativo da área de matemática pode ser utilizado para determinar os auto-valores e auto-vetores de **C**, como, por exemplo, o MatLab da MathWorks Inc.

A Figura 1 mostra o conjunto $\mathbf{U}$, os $M = 2$ auto-vetores escalonados $\underline{\psi}_m = \underline{e}_m \cdot \sqrt{\lambda_m}$, $m = 0, 1, \cdots, M-1$ e os eixos Cartesianos por eles definidos.

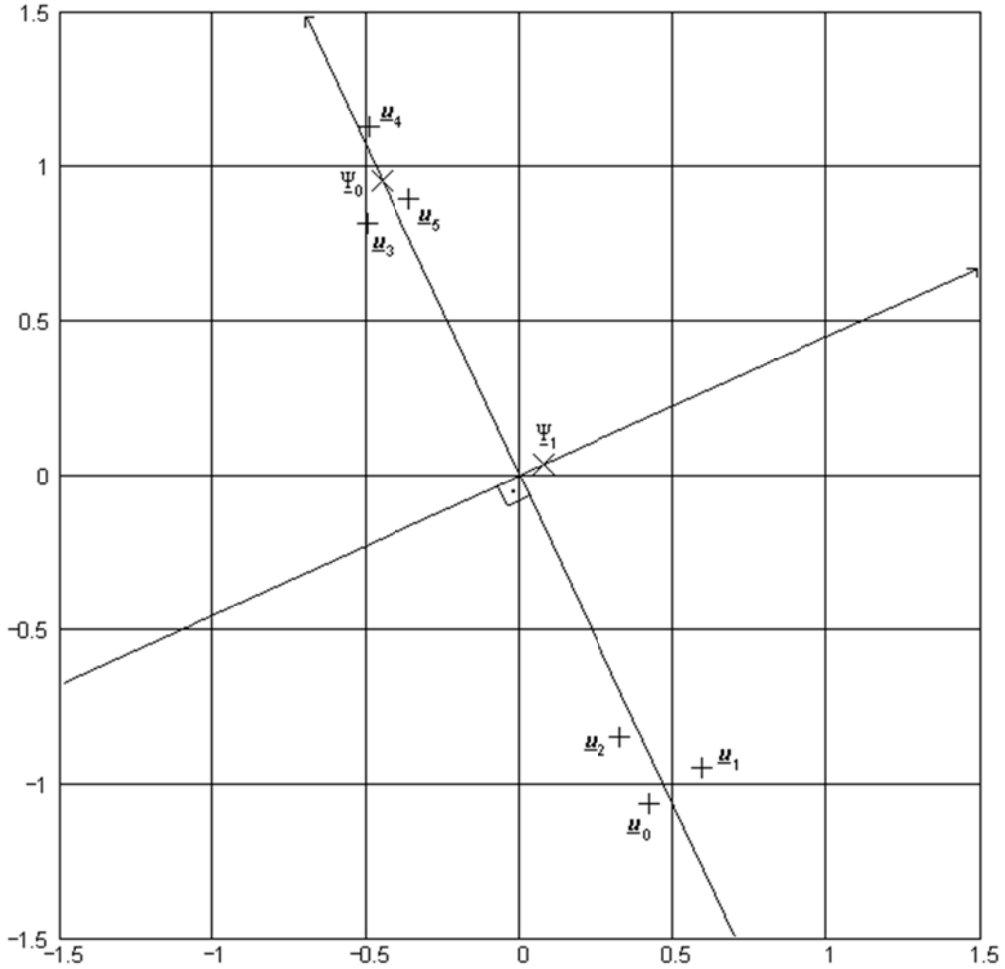


Figura 1: Conjunto $\mathbf{U}$, auto-vetores escalonados $\underline{\psi}_0$ e $\underline{\psi}_1$ e os eixos Cartesianos por eles definidos

A Tabela 1 mostra o valor da projeção de cada vetor $\underline{u}_i \in \Re^2$ do conjunto $\mathbf{U}$ sobre os eixos Cartesianos definidos por $\underline{e}_0$ e $\underline{e}_1$.

i	0	1	2	3	4	5
$a_{0i} = \underline{u}_i^T \cdot \underline{e}_0$	-1.142	-1.107	-0.902	0.953	1.234	0.964
$a_{1i} = \underline{u}_i^T \cdot \underline{e}_1$	-0.07	0.135	-0.065	-0.099	0.04	0.059

Tabela 1: Projeções $\mathbf{a}_0$ e $\mathbf{a}_1$ de $\mathbf{U}$ sobre os eixos Cartesianos definidos por $\underline{e}_0$ e $\underline{e}_1$.

Observe que, como os auto-vetores $\underline{e}_0$ e $\underline{e}_1$ têm norma unitária e são adimensionais, o valor absoluto da projeção de cada $\underline{u}_i \in \Re^2$ sobre cada eixo define a norma Euclidiana da *i*-ésima projeção.

Note que a variância do conjunto $\mathbf{U}$ é dada pela soma dos auto-valores, i.e.,

$$\frac{1}{L}\sum_{i=0}^{L-1}\left\|\underline{u}_i\right\|^2 = 1.125 = \lambda_0 + \lambda_1,$$

onde $\left\|\underline{u}\right\| = \sqrt{\sum_{m=0}^{M-1}\left(u_m\right)^2} = \sqrt{\underline{u}^T \cdot \underline{u}}$ é a norma Euclidiana do vetor $\underline{u} \in \Re^M$.

Note ainda que as variâncias das projeções $\mathbf{a}_0$ e $\mathbf{a}_1$ de $\mathbf{U}$ equivalem aos respectivos auto-valores, i.e.,

$$\frac{1}{L}\sum_{i=0}^{L-1}\left(\underline{u}_i{}^T \cdot \underline{e}_0\right)^2 = 1.118 = \lambda_0 \quad \text{e} \quad \frac{1}{L}\sum_{i=0}^{L-1}\left(\underline{u}_i{}^T \cdot \underline{e}_1\right)^2 = 7.035\times 10^{-3} = \lambda_1$$

Portanto, a variância do conjunto projetado, ou variância do sub-espaço, é dada pelo *m*-ésimo auto-valor λ_m.

Como observação adicional note que a distância Euclidiana média $\frac{1}{L}\sum_{i=0}^{L-1}\left|\underline{u}_i{}^T \cdot \underline{e}_m\right|$ dos vetores da *m*-ésima projeção à origem pode ser aproximada por $\sqrt{\frac{1}{L}\sum_{i=0}^{L-1}\left(\underline{u}_i{}^T \cdot \underline{e}_m\right)^2} = \sqrt{\lambda_m}$.

No caso,

$$\frac{1}{L}\sum_{i=0}^{L-1}\left|\underline{u}_i{}^T \cdot \underline{e}_0\right| = 1.051 \text{ para } \sqrt{\lambda_0} = 1.057$$

e

$$\frac{1}{L}\sum_{i=0}^{L-1}\left|\underline{u}_i{}^T \cdot \underline{e}_1\right| = 0.078 \text{ para } \sqrt{\lambda_1} = 0.083.$$

A compressão resultante da aplicação da KLT a um conjunto de vetores

Na seção anterior vimos que os M eixos Cartesianos resultantes da KLT alinham-se com as direções de maior variância (=energia) de um conjunto $\mathbf{U}$ de vetores $\Re^M$ dimensionais, sendo $\underline{0} \in \Re^M$ o vetor média de $\mathbf{U}$. (O Exemplo 1 ilustrou um caso para $M = 2$.)

A Figura 2 mostra uma representação pictórica hipotética de uma nuvem de dados em $\Re^3$. Cada ponto da nuvem define a ponta de um vetor deste conjunto $\mathbf{U}$ em $\Re^3$. A figura mostra a maneira como a base ortonormal formada pelos auto-vetores alinha-se com as direções de maior variância (energia) de $\mathbf{U}$.

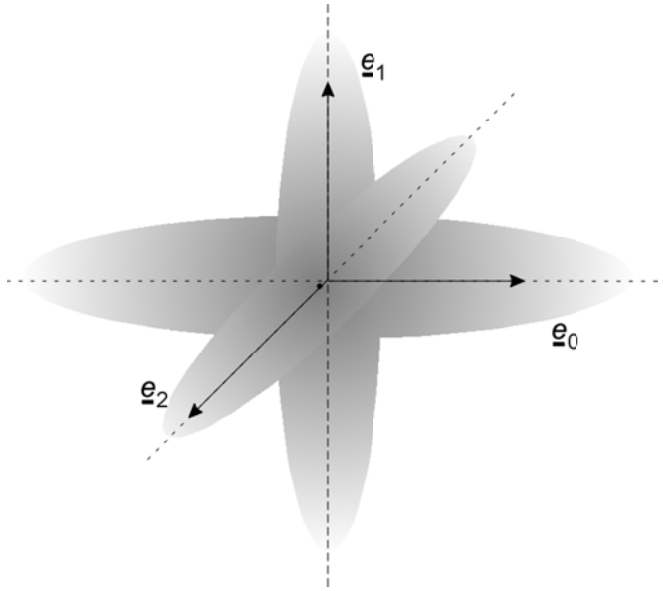
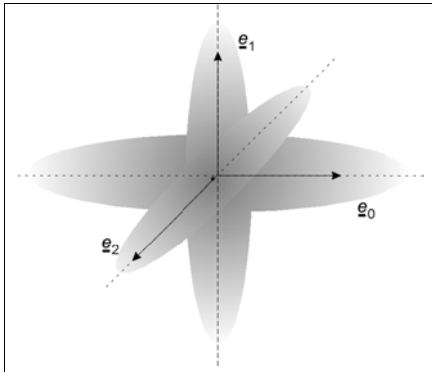


Figura 2: Representação pictórica hipotética de uma "nuvem" de dados em $\Re^3$ e a maneira como a base ortonormal formada pelos auto-vetores $\underline{e}_0$, $\underline{e}_1$ e $\underline{e}_2$ alinha-se com as direções de maior variância.



Assim, no caso genérico de um conjunto $\mathbf{U}$ em $\Re^M$, podemos imaginar a KLT "girando" uma base de M vetores ortonormais em $\Re^M$ em todas as possíveis direções até que os vetores alinhem-se com as direções de maior variância possível em $\mathbf{U}$.

Nesta situação os vetores da base ortonormal são os auto-vetores da matriz de covariância de $\mathbf{U}$.

Vimos também que a energia contida em uma determinada direção (= variância da projeção de $\mathbf{U}$ na direção) é dada pelo auto-valor associado ao auto-vetor que define a direção.

Portanto, podemos desprezar aquelas direções (= sub-espaços) definidas por auto-vetores cujo auto-valor associado é muito menor do que os demais auto-valores.

Isto é possível porque a projeção de $\mathbf{U}$ sobre tais direções é insignificante se comparada com as projeções cujo auto-valor associado não é comparativamente pequeno.

Ao desprezar sub-espaços de menor energia estamos realizando **uma compressão com perdas** do conjunto $\mathbf{U}$.

No entanto estas perdas são mínimas, pois as componentes (= sub-espaços) descartadas têm pouca influência na formação de $\mathbf{U}$ porque os auto-valores a elas associados supostamente têm valor muito menor do que os demais.

Neste sentido ocorre a **redução dimensional** de $\mathbf{U}$, porque o conjunto era originalmente representado em $\Re^M$ e, ao descartar sub-espaços não significativos, o conjunto passa a ser representado com boa aproximação em uma dimensão menor que M.

Este é o motivo de a KLT ser considerada uma transformação ótima sob o ponto de vista do MSE.

Exemplo 2 – a KLT como algoritmo para compressão

Seja o conjunto $\mathbf{U}$ do Exemplo 1 (slide 12):

a) Execute uma compressão com perdas de $\mathbf{U}$ utilizando a KLT.

b) Calcule a compressão obtida.

c) Calcule o MSE da compressão obtida.

d) Calcule a Relação Sinal – Ruído de Pico em dB, denotada PSNR (PSNR – *Peak Signal To Noise Ratio*), resultante do processo de compressão. Isto é, calcule o quadrado da componente de maior valor absoluto dentre todas as componentes dos vetores de $\mathbf{U}$ e normalize pelo MSE obtido em c).

Solução:

No Exemplo 1 foram determinadas as projeções de $\mathbf{U}$ sobre a base ortonormal formada pelos auto-vetores $\underline{e}_0$ e $\underline{e}_1$, conforme mostra a Tabela 2. Observe na Tabela 2 que as projeções sobre a direção $\underline{e}_1$ são muito menores do que as projeções sobre a direção $\underline{e}_0$.

i	0	1	2	3	4	5
$a_{0i} = \underline{u}_i^T \cdot \underline{e}_0$	-1.142	-1.107	-0.902	0.953	1.234	0.964
$a_{1i} = \underline{u}_i^T \cdot \underline{e}_1$	-0.07	0.135	-0.065	-0.099	0.04	0.059

Tabela 2: Projeções $\mathbf{a}_0$ e $\mathbf{a}_1$ de $\mathbf{U}$ sobre os eixos Cartesianos definidos por $\underline{e}_0$ e $\underline{e}_1$.

Esta característica das projeções está de acordo com o fato de que $\lambda_1 = 7.035 \times 10^{-3}$ é muito menor do que $\lambda_0 = 1.118$.

Sendo assim, podemos descartar as projeções ou componentes de $\mathbf{U}$ na direção $\underline{e}_1$ (a direção com menor auto-valor associado) e considerar as projeções de $\mathbf{U}$ na direção $\underline{e}_0$ (a direção com maior auto-valor associado) como uma boa aproximação de $\mathbf{U}$.

Portanto a compressão com perdas consiste em aproximar $\mathbf{U}$ através do auto-vetor $\underline{e}_0$ e do conjunto de projeções $a_{0i} = \underline{u}_i^{\,T} \cdot \underline{e}_0$ na direção por ele definida. A aproximação $\widetilde{\mathbf{U}}$ do conjunto de vetores originais $\underline{u}_i \in \mathbf{U}$, é obtida através de $\underline{\widetilde{u}}_i = a_{0i}\underline{e}_0$, $\underline{\widetilde{u}}_i \in \widetilde{\mathbf{U}}$, isto é,

I	0	1	2	3	4	5
$a_{0i} = \underline{u}_i^{\,T} \cdot \underline{e}_0$, $\underline{e}_0 = \begin{bmatrix} -0.427 \\ 0.904 \end{bmatrix}$	-1.142	-1.107	-0.902	0.953	1.234	0.964
$\underline{\widetilde{u}}_i = a_{0i}\underline{e}_0$	$\begin{bmatrix} 0.488 \\ -1.032 \end{bmatrix}$	$\begin{bmatrix} 0.473 \\ -1.001 \end{bmatrix}$	$\begin{bmatrix} 0.385 \\ -0.815 \end{bmatrix}$	$\begin{bmatrix} -0.407 \\ 0.862 \end{bmatrix}$	$\begin{bmatrix} -0.527 \\ 1.116 \end{bmatrix}$	$\begin{bmatrix} -0.412 \\ 0.871 \end{bmatrix}$

Tabela 3: Aproximação $\widetilde{\mathbf{U}}$ do conjunto original $\mathbf{U}$ obtida através de $\underline{\widetilde{u}}_i = a_{0i}\underline{e}_0$, onde $\underline{\widetilde{u}}_i \in \widetilde{\mathbf{U}}$.

Portanto, da Tabela 3, o conjunto $\widetilde{\mathbf{U}}$ resultante é

$$\widetilde{\mathbf{U}} = \left\{ \begin{bmatrix} 0.488 \\ -1.032 \end{bmatrix}, \begin{bmatrix} 0.473 \\ -1.001 \end{bmatrix}, \begin{bmatrix} 0.385 \\ -0.815 \end{bmatrix}, \begin{bmatrix} -0.407 \\ 0.862 \end{bmatrix}, \begin{bmatrix} -0.527 \\ 1.116 \end{bmatrix}, \begin{bmatrix} -0.412 \\ 0.871 \end{bmatrix} \right\}$$

Observe que o conjunto original

$$\mathbf{U} = \left\{ \begin{bmatrix} 0.425 \\ -1.063 \end{bmatrix}, \begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}, \begin{bmatrix} 0.327 \\ -0.843 \end{bmatrix}, \begin{bmatrix} -0.497 \\ 0.820 \end{bmatrix}, \begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}, \begin{bmatrix} -0.359 \\ 0.897 \end{bmatrix} \right\}$$

de $L = 6$ vetores $\underline{u}_i \in \Re^2$, $i = 0,1,\cdots, L-1$ necessita de 12 números em ponto-flutuante para ser representado.

Por outro lado, o conjunto aproximado $\widetilde{\mathbf{U}}$ foi gerado a partir do auto-vetor $\underline{e}_0 = \begin{bmatrix} -0.427 \\ 0.904 \end{bmatrix}$ e de um conjunto de 6 projeções

$\mathbf{a}_0 = \{-1.142, -1.107, -0.902, 0.953, 1.234, 0.964\}$.

Portanto, o conjunto $\widetilde{\mathbf{U}}$ necessita de apenas 8 números em ponto-flutuante para ser representado.

Define-se como fator de compressão ρ o quociente

$$\rho = \frac{\text{Total de unidades de armazenamento necessário para representar } \widetilde{\mathbf{U}}}{\text{Total de unidades de armazenamento necessário para representar } \mathbf{U}} \qquad (6.45)$$

Portanto o fator de compressão obtido é $\rho = 8/12 = 0.67$.

O MSE da aproximação $\widetilde{\mathbf{U}}$ pode ser obtido através de

$$\text{MSE} = \frac{1}{L}\sum_{i=0}^{L-1}\left(\underline{\widetilde{u}}_i - \underline{u}_i\right)^T\left(\underline{\widetilde{u}}_i - \underline{u}_i\right) \quad (6.46),$$

onde $\underline{u}_i \in \mathbf{U}$ e $\underline{\widetilde{u}}_i \in \widetilde{\mathbf{U}}$, $i = 0,1,\cdots,L-1$.

De (6.46) obtemos $\text{MSE} = 7.025\times10^{-3}$.

A PSNR é definida como $\text{PSNR} = 10\log\left(\dfrac{\left(\max comp\{\mathbf{U}\}\right)^2}{\text{MSE}}\right)$ (6.47)

sendo $\max comp\{\mathbf{U}\}$ a componente de maior valor absoluto dentre todas as componentes dos vetores de $\mathbf{U}$.

De (6.47) obtemos $\text{PSNR} = 10\log\left(\dfrac{(1.133)^2}{7.025\times10^{-3}}\right) = 22.6\,\text{dB}$.

Exemplo 3 – reconstrução do conjunto original usando todas as projeções da KLT

Obtenha o conjunto de vetores $\underline{\tilde{u}}_i \in \tilde{\mathbf{U}}$ a partir de **todos** os sub-espaços do conjunto de vetores $\underline{u}_i \in \mathbf{U}$ do Exemplo 6.2 (i.e., não execute nenhuma compressão, apenas reconstrua o conjunto original a partir de ambos sub-espaços $\mathbf{a}_0$ e $\mathbf{a}_1$). Determine a PSNR do conjunto reconstruído $\tilde{\mathbf{U}}$.

Solução:

A partir da Tabela 1 (slide 15) obtemos a Tabela 4 abaixo.

i	0	1	2	3	4	5
$a_{0i} = \underline{u}_i^T \cdot \underline{e}_0, \quad \underline{e}_0 = \begin{bmatrix} -0.427 \\ 0.904 \end{bmatrix}$	-1.142	-1.107	-0.902	0.953	1.234	0.964
$a_{1i} = \underline{u}_i^T \cdot \underline{e}_1, \quad \underline{e}_1 = \begin{bmatrix} 0.904 \\ 0.427 \end{bmatrix}$	-0.07	0.135	-0.065	-0.099	0.04	0.059
$\underline{\tilde{u}}_i = a_{0i}\underline{e}_0 + a_{1i}\underline{e}_1$	$\begin{bmatrix} 0.424 \\ -1.062 \end{bmatrix}$	$\begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}$	$\begin{bmatrix} 0.326 \\ -0.843 \end{bmatrix}$	$\begin{bmatrix} -0.496 \\ 0.819 \end{bmatrix}$	$\begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}$	$\begin{bmatrix} -0.358 \\ 0.897 \end{bmatrix}$

Tabela 4: Reconstrução $\tilde{\mathbf{U}}$ do conjunto original $\mathbf{U}$ obtida através de $\underline{\tilde{u}}_i = a_{0i}\underline{e}_0 + a_{1i}\underline{e}_1$, onde $\underline{\tilde{u}}_i \in \tilde{\mathbf{U}}$. Como nenhum sub-espaço é descartado, nenhuma compressão é obtida e o conjunto $\tilde{\mathbf{U}}$ é considerado uma reconstrução do conjunto original $\mathbf{U}$ a partir dos sub-espaços $\mathbf{a}_0$ e $\mathbf{a}_1$.

Da Tabela 4 obtemos

$$\widetilde{\mathbf{U}} = \left\{ \begin{bmatrix} 0.424 \\ -1.062 \end{bmatrix}, \begin{bmatrix} 0.595 \\ -0.943 \end{bmatrix}, \begin{bmatrix} 0.326 \\ -0.843 \end{bmatrix}, \begin{bmatrix} -0.496 \\ 0.819 \end{bmatrix}, \begin{bmatrix} -0.491 \\ 1.133 \end{bmatrix}, \begin{bmatrix} -0.358 \\ 0.897 \end{bmatrix} \right\}.$$

De (6.46) obtemos $\mathrm{MSE} = 5.3 \times 10^{-7}$.

E, de (6.47) obtemos $$\mathrm{PSNR} = 10 \log \left(\frac{(1.133)^2}{5.3 \times 10^{-7}} \right) = 63.8 \, \mathrm{dB}.$$

Nota:

Observe que o MSE não é zero (e portanto a PSNR não é infinita) unicamente devido à precisão (3 casas após a vírgula) nas operações de ponto-flutuante efetuadas.

Se estivéssemos trabalhando com uma precisão suficiente o MSE seria zero porque neste exemplo, ao contrário do Exemplo 2, nenhuma informação é descartada.

Aqui o conjunto original $\mathbf{U}$ é apenas reconstruído a partir de todos os seus sub-espaços sem ocorrer qualquer compressão.

Exemplo 4 – Compressão de imagens através da KLT

Figura 3: Imagem *grayscale* “Lenna” de tamanho 128×128 pixels.

- Uma imagem em *grayscale* tem seus valores de pixel variando entre 0 e 255.
- O valor 0 representa preto e o valor 255 representa branco.
- Todos os demais valores intermediários representam tons de cinza (portanto, a tonalidade cinza de um pixel de uma imagem *grayscale* clareia à medida que o valor do pixel aumenta).
- A Figura 3 mostra a imagem *grayscale* “Lenna” de tamanho 128×128 pixels, a qual será utilizada em nosso estudo.

Para comprimir a imagem da Figura 3 através da KLT, adota-se o seguinte procedimento:

1. Subdivide-se a imagem em $L = 256$ blocos de 8×8 pixels e registra-se a posição de cada bloco na imagem (p/ certas imagens, melhor resultado é obtido c/ blocos 16×16) .
2. O *i*-ésimo bloco $\mathbf{B}_i$ de 8×8 pixels é convertido em um vetor $\Re^M$ dimensional $\underline{u}_i \in \mathbf{U}$, sendo $M = 64$ (devido ao bloco possuir 8×8 pixels), $i = 0,1,\cdots,L-1$. A conversão bloco-vetor $\mathbf{B}_i \rightarrow \underline{u}_i$ é efetuada lendo-se as 8 linhas de $\mathbf{B}_i$ da esquerda para a direita sendo a linha ao alto a primeira a ser lida e transferindo o valor de cada pixel lido nesta ordem para as respectivas posições em sequência no vetor $\underline{u}_i$.
3. Calcula-se o vetor média do conjunto de vetores $\mathbf{U}$ obtido em (2) e subtrai-se este vetor média de todos os $L = 256$ vetores $\underline{u}_i \in \mathbf{U}$.
4. Determina-se os sub-espaços de **U** conforme já discutido no Exemplo 2.
5. Descarta-se aqueles sub-espaços associados aos **menores** auto-valores .
6. Obtém-se o conjunto de vetores $\underline{\tilde{u}}_i \in \tilde{\mathbf{U}}$ sendo $\underline{\tilde{u}}_i = a_{0i}\underline{e}_0 + a_{1i}\underline{e}_1 + \cdots + a_{N-1i}\underline{e}_{N-1}$, onde $\{\mathbf{a}_0, \mathbf{a}_1, \cdots, \mathbf{a}_{N-1}\}$ são os sub-espaços definidos pelos auto-vetores $\{\underline{e}_0, \underline{e}_1, \cdots, \underline{e}_{N-1}\}$ cujos auto-valores associados são os *N* **maiores** auto-valores (os *N* sub-espaços de maior energia). *N* é obtido experimentalmente e define o compromisso entre o fator de compressão ρ e a relação sinal-ruído de compressão PSNR.
7. Soma-se ao conjunto de *L* vetores $\underline{\tilde{u}}_i \in \tilde{\mathbf{U}}$ o vetor média originalmente subtraído de **U**.
8. Obtém-se o conjunto de blocos $\tilde{\mathbf{B}}_i$ através da conversão bloco-vetor inversa $\underline{\tilde{u}}_i \rightarrow \tilde{\mathbf{B}}_i$, $i = 0,1,\cdots,L-1$, e reconstrói-se a imagem comprimida a partir do registro da posição de cada bloco.

A Figura 4 mostra a amplitude relativa em ordem decrescente dos 32 primeiros maiores auto-valores do conjunto $\mathbf{U}$ formado a partir da Figura 3 de acordo com o procedimento acima descrito.

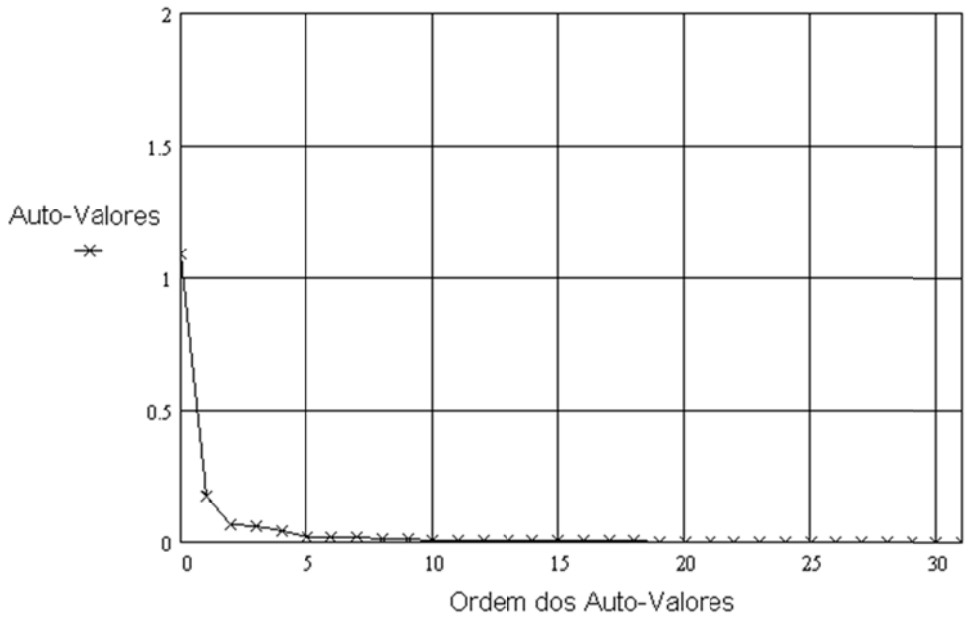


Figura 4: Amplitude relativa em ordem decrescente dos 32 primeiros maiores auto-valores do conjunto $\mathbf{U}$ formado a partir da Figura 3. Observe que o número total de auto-valores é 64, já que cada vetor de $\mathbf{U}$ é de dimensão $\Re^{64}$. No entanto, visto que aproximadamente a partir do vigésimo maior auto-valor a amplitude relativa torna-se pequena, escolheu-se representar apenas as 32 primeiras maiores amplitudes.

Figura 5:

Imagem da Figura 3 comprimida pela KLT utilizando os $N = 32$ primeiros sub-espaços de maior energia.

O coeficiente de compressão resultante é $\rho = 0.629$ (Equação (6.45)).

A fidelidade da imagem comprimida com relação à original é dada por $\mathrm{PSNR} = 37.3\,\mathrm{dB}$ (Eq. (6.47)).

Figura 6:

Imagem da Figura 3 comprimida pela KLT utilizando os $N = 16$ primeiros sub-espaços de maior energia.

O coeficiente de compressão resultante é $\rho = 0.316$ (Equação (6.45)).

A fidelidade da imagem comprimida com relação à original é dada por $\mathrm{PSNR} = 31.4\,\mathrm{dB}$ (Eq. (6.47)).

General Hebbian Learning para PCA (DSE de um espaço vetorial através de RNAs)

PCA via KLT de um Espaço Vetorial:

(1) Computar a matriz de covariância do conjunto de dados de entrada.

(2) Aplicar procedimento numérico que extraia os auto-valores e os correspondentes auto-vetores desta matriz.

(3) Os auto-vetores correspondentes aos auto-valores mais significativos são usados para extrair os componentes principais do espaço vetorial do conjunto de dados.

Desvantagens do método:

(1) Para grandes conjuntos de dados, as dimensões da matriz covariância crescem significativamente, tornando proibitiva a complexidade computacional para determinação dos auto-valores e auto-vetores.

(2) Em particular, a complexidade é alta porque todos os auto-valores e auto-vetores precisam ser computados, mesmo que somente poucos auto-vetores (aqueles correspondentes aos maiores auto-valores) venham a ser utilizados no processo.

Solução:

⇒ A solução eficiente para este problema deve encontrar os principais auto-vetores sem a necessidade de determinar a matriz covariância.

⇒ Tal solução é alcançada ao abordar o problema utilizando RNAs.

⇒ A técnica que utiliza RNAs para efetuar PCA de um espaço vetorial é denominada "Algoritmo Hebbiano Generalizado" (GHA – *General Hebbian Learning*).

⇒ O GHA foi proposto por Sanger em 1989. O algoritmo GHA combina a orto-normalização de Gram-Schmidt com o modelo de um único neurônio linear introduzido por Oja em 1982.

O GHA é um algoritmo computacionalmente eficiente para PCA porque:

- Extrai os componentes principais individualmente, um após o outro, em ordem decrescente de variância.
- Possibilita que, após treinada a RNA, os resultados obtidos possam ser aplicados a um outro conjunto de dados de estatística "semelhante" (ou seja, que resulte em uma matriz de covariância "semelhante").

O desenvolvimento aqui apresentado segue a exposição de Haykin em [7], Hassoun em [11] e Hertz, Krogh e Palmer em [15].

O Aprendizado Hebbiano

É uma classe de aprendizado não-supervisionado ou auto-organizado, em que os parâmetros livres da rede são adaptados pela exposição da RNA ao ambiente em que está contida, até que uma pré-determinada condição seja atingida, através:

– da repetição dos padrões de entrada por um número suficiente de vezes e

– da aplicação de um conjunto de regras de aprendizado específicas, adequadas à solução do problema.

➔ RNAs treinadas sob aprendizado não-supervisionado descobrem padrões significativos ou característicos dos dados sem o auxílio de um tutor externo.

➔ Não há realimentação do ambiente para auxiliar a rede a determinar se a saída está ou não correta.

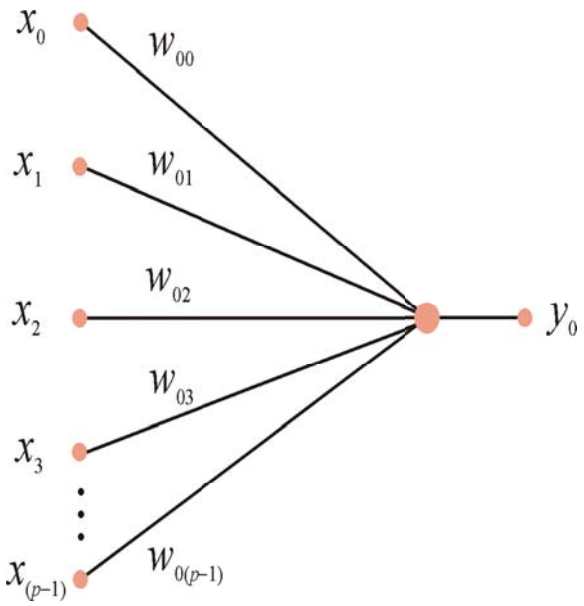
➔ A RNA deve descobrir por si padrões, características, regularidades, correlações ou categorias nos dados de entrada.

➔ As unidades e conexões que a compõem devem, portanto, apresentar alguma capacidade de auto-organização.

O conhecimento que "coalesce" nas sinapses da RNA é obtido da observação repetida de parâmetros estatísticos (média, variância e matriz correlação) dos dados de entrada

(redundância de padrões ⇒ conhecimento).

Figura 7: RNA de um único neurônio de índice “0” (zero), a ser treinada pelo Aprendizado Hebbiano.



➔ Os sinais pré e pós-sinápticos, respectivamente $\underline{x}$ e y são equacionados por

$$\underline{w}_0(n+1) = \underline{w}_0(n) + \eta\, y_0(n)\underline{x}(n) \qquad (6.48)$$

$$y_0(n) = \underline{x}(n)^T\, \underline{w}_0(n) = \underline{w}_0(n)^T\, \underline{x}(n) \qquad (6.49)$$

onde:

- η é uma constante positiva que determina a razão de aprendizado,
- $\underline{x}(n)$ é o vetor de entrada da rede no instante n e
- $y_0(n)$ é a saída da rede.

Observe que a Equação (6.48) expressa o ajuste no vetor de pesos sinápticos $\underline{w}_0(n)$, que ocorre na proporção da **correlação entre os sinais pré e pós-sinápticos**, enquanto que a Equação (6.49) expressa a saída $y_0(n)$ da RNA no instante n.

O conjunto de vetores $\underline{x}$ de entrada possui distribuição de probabilidade arbitrária.

A cada instante n do tempo discreto um vetor $\underline{x}$ é apresentado à rede, escolhido aleatoriamente do conjunto de vetores do qual se deseja efetuar o processo PCA.

Segundo os princípios do Aprendizado Hebbiano:

(I) quanto mais provável for uma particular entrada $\underline{x}$, maior será a correlação da saída y com esta entrada. Assim, quanto mais provável for $\underline{x}$, maior será a saída y;

(II) quanto maior for a saída y, mais a variação do peso sináptico que a encorajou aumentará a sua transmitância.

As afirmações (I) e (II) nada mais são do que o 1º princípio da auto-organização de von der Malsburg:

"Modificações em pesos sinápticos tendem a se auto-amplificar".

Ou seja, a correlação entre os ajustes nos pesos sinápticos e as variações nos padrões de atividade deve ser positiva para que se obtenha auto-organização da rede.

O processo de auto-amplificação é restrito pelo requerimento de que modificações em pesos sinápticos devem ser baseadas em sinais pré-sinápticos e pós-sinápticos (disponíveis localmente).

Uma sinapse com alta transmitância conduz à coincidência de sinais pré e pós-sinápticos e, por outro lado, a transmitância da sinapse é aumentada por tal coincidência.

Ou, ainda, segundo o postulado de Hebb:

"Se dois neurônios biológicos em cada lado de uma sinapse são ativados simultaneamente, então a transmitância daquela sinapse é seletivamente aumentada."

- O "*loop*" recorrente expresso por (I) e (II), no entanto, faz com que a transmitância dos pesos sinápticos permaneça crescendo sem limite, impedindo a continuação do processo de aprendizado por *overflow* dos registradores numéricos do *hardware*.
- Pelo aumento demasiado da transmitância do peso sináptico, a rede é levada à saturação precoce e é incapaz de aprender os padrões apresentados.

Solução do problema de *overflow*: 2º princípio da auto-organização de von der Malsburg:

"Limitação de recursos conduz à competição entre as sinapses e, consequentemente, à seleção das sinapses que crescem de forma mais vigorosa (portanto, as mais adequadas) às custas de outras"

Neste contexto, é necessário ser estabelecido algum mecanismo de competição por "recursos limitados", para que o sistema possa ser numericamente estabilizado.

A solução, portanto, consiste em compensar o aumento na transmitância de uma particular sinapse do neurônio através do decréscimo da transmitância de outras sinapses.

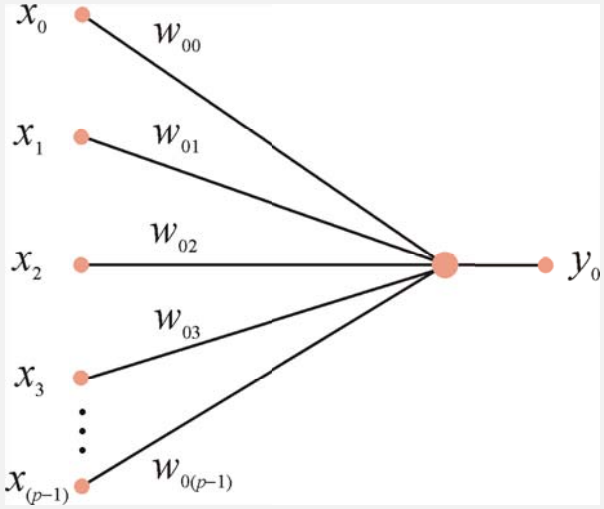
Assim, apenas as sinapses que obtêm sucesso no processo de correlação entre os sinais de entrada e saída tem a sua transmitância aumentada, enquanto as outras tem sua transmitância reduzida e, eventualmente, deixam de influenciar no processo de aprendizado como um todo.

Solução proposta por Oja para a instabilidade numérica (*overflow*):

Para evitar a divergência numérica do algoritmo de aprendizado Hebbiano é preciso restringir o aumento das transmitâncias das sinapses expressas nos componentes do vetor $\underline{w}$.

Oja propôs em 1982 como alternativa para a estabilização do Aprendizado Hebbiano, uma modificação no algoritmo que permite ao vetor peso sináptico $\underline{w}$ atingir norma unitária, além de corresponder ao auto-vetor associados ao maior auto-valor de C_x.

A regra de Oja corresponde a adicionar um decaimento ou deflação ao ajuste do vetor de pesos sinápticos $\underline{w}$, deflação que é proporcional à "energia" ou "potência" do sinal na saída do neurônio no instante discreto n, i.e., proporcional à $y_0^{\,2}(n)$.



A Equação (6.60) expressa a regra de Aprendizado Hebbiano, após deflacionada de um fator $\eta\, y_0^{\,2}(n)\underline{w}_0(n)$:

$$\underline{w}_0(n+1) = \underline{w}_0(n) + \eta\ \ y_0(n)\underline{x}(n) - \eta\, y_0^{\,2}(n)\underline{w}_0(n) \qquad (6.60)$$

A regra de ajuste deflacionado para o Aprendizado Hebbiano apresentada na Equação (6.60) pode ser reescrita como,

$$\Delta\underline{w}_0(n) = \eta\ \ y_0(n)\{\underline{x}(n) - y_0(n)\underline{w}_0(n)\} \qquad (6.61)$$

Esta regra de ajuste deflacionado conduz à convergência do vetor peso sináptico $\underline{w}_0$ para $\|\underline{w}_0\| = 1$, sendo, assim, uma solução numericamente estável.

Importante notar que, após a convergência para $\|\underline{w}_0\| = 1$, o vetor $\underline{w}_0$ resultante corresponde ao auto-vetor associado ao maior auto-valor de C_x (ver demonstração na seção 6.3.1 em http://www.fccdecastro.com.br/pdf/RNA_C6.pdf), i.e., corresponde ao auto-vetor $\underline{e}_0$, associado ao maior auto-valor $\lambda_0 = \lambda_{\max}$. Portanto um único neurônio linear submetido à regra Hebbiana de aprendizado auto-organizado extrai a primeira componente principal do espaço vetorial dos dados de entrada, componente esta que corresponde ao auto-vetor associado ao maior auto-valor da matriz de covariância dos dados de entrada.

O Algoritmo Hebbiano Generalizado

O processo de generalização consiste em aplicar as regras propostas por Sanger e Oja a uma RNA progressiva, composta não mais por um único neurônio linear (conforme Figura 7) mas sim, por uma camada de neurônios lineares, conforme mostra a Figura 8. A RNA treinada por esta regra executará a análise dos componentes principais sobre o espaço vetorial que representa os dados de entrada.

Os m neurônios da camada de neurônios lineares da RNA extraem ordenadamente e respectivamente os m componentes principais do espaço vetorial representado pelo conjunto de vetores $\underline{x}(n)$ de entrada. Após a convergência do algoritmo, o vetor de p elementos que armazena as transmitâncias sinápticas do j-ésimo neurônio, corresponderá ao j-ésimo auto-vetor da matriz de covariância (matriz de correlação) C_x do conjunto de vetores $\underline{x}(n)$. O ordenamento é tal que o neurônio $j = 0$ representará o auto-vetor associado ao maior auto-valor e o neurônio $j = m - 1$ representará o auto-vetor associado ao menor auto-valor.

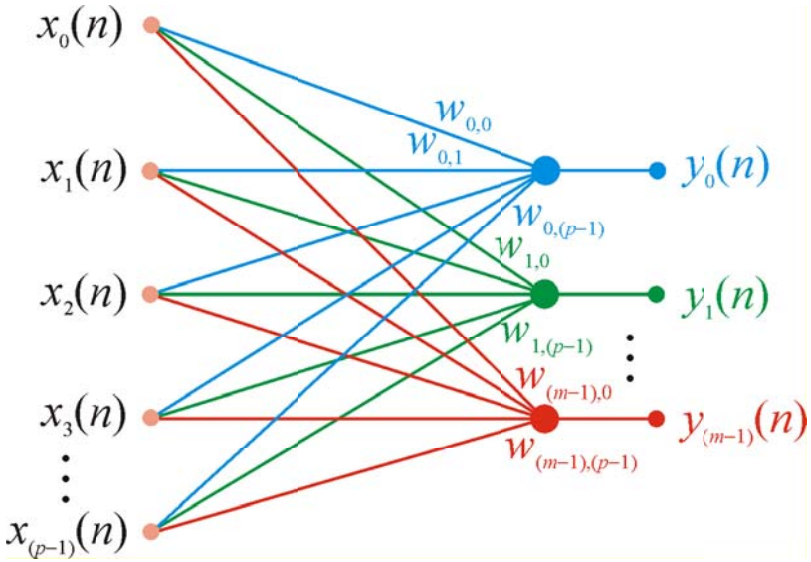


Figura 8: RNA progressiva com uma única camada de neurônios lineares, a ser treinada pelo Aprendizado Hebbiano Generalizado.

Note que:

- A RNA da Figura 8 tem p nós na camada de entrada e m neurônios na camada de saída, com $m \leq p$.
- Os pesos sinápticos w_{ji} conectam os nós de entrada i aos neurônios da camada de saída j, com $i = 0, 1, ..., p-1$ e $j = 0, 1, ..., m-1$ (considerando aqui, novamente, a notação expandida para o vetor de pesos sinápticos).
- A saída $y_j(n)$ produzida pelo neurônio j, no instante discreto n, em resposta ao conjunto de entradas $x_i(n)$ é dada por

$$y_j(n) = \sum_{i=0}^{p-1} w_{ji}(n) x_i(n), \qquad j = 0, 1, ..., m-1 \qquad (6.89)$$

- A adaptação do peso sináptico devida ao Algoritmo Hebbiano Generalizado é expressa pela Equação (6.90)

$$\Delta w_{ji}(n) = \eta \{ y_j(n) x_i(n) - y_j(n) \sum_{k=0}^{j} w_{ki}(n) y_k(n) \} \qquad (6.90)$$

onde $i = 0, 1, ..., p-1$, $j = 0, 1, ..., m-1$, $\Delta w_{ji}(n)$ é o ajuste aplicado ao peso sináptico $w_{ji}(n)$ do neurônio j no instante n, e η é a razão de aprendizado.

Para efeito de análise e interpretação da Equação (6.90), note que a equação 6.61do slide 34 é uma particularização de (6.90) para um único neurônio $j = 0$:

$$\Delta \underline{w}_0(n) = \eta\ y_0(n)\{\underline{x}(n) - y_0(n)\underline{w}_0(n)\} = \eta\{y_0(n)\underline{x}(n) - y_0(n)y_0(n)\underline{w}_0(n)\}$$

$$\Delta w_{0i}(n) = \eta \{y_0(n)x_i(n) - y_0(n)w_{0i}(n)y_0(n)\}$$

Neste contexto (6.90) pode ser interpretada como uma regra que adiciona deflação ao ajuste do vetor de pesos sinápticos $\underline{w}$ do j-ésimo neurônio, deflação que é proporcional à soma das "energias" ou "potências" cruzadas (correlação cruzada) entre o sinal na saída do j-ésimo neurônio e o sinal na saída de todos os demais neurônios associados a autovalores maiores que o autovalor do j-ésimo neurônio, sendo também proporcional à "energia" ou "potência" do sinal na saída do j-ésimo neurônio, i.e., proporcional à $y_j^2(n)$.

Note também que a Equação (6.90), que expressa o ajuste no peso sináptico através do Algoritmo Hebbiano Generalizado, pode ser alternativamente escrita como:

$$\Delta w_{ji}(n) = \eta \; y_j(n)\, x_i'(n) - \eta\, y_j{}^2(n)\, w_{ji}(n), \qquad (6.91)$$
$$i = 0, 1, ..., p\text{-}1$$
$$j = 0, 1, ..., m\text{-}1$$

onde o vetor $\underline{x}'(n)$ representa a forma deflacionada do vetor de entrada $\underline{x}(n)$, deflação que é proporcional à soma das "energias" ou "potências" cruzadas (correlação cruzada) entre o sinal na saída do *j*-ésimo neurônio e o sinal na saída de todos os demais neurônios associados a autovalores maiores que o autovalor do *j*-ésimo neurônio:

$$x_i'(n) = x_i(n) - \sum_{k=0}^{j-1} w_{ki}(n) y_k(n) \qquad (6.92)$$

Tabela 4: Operação do Algoritmo Hebbiano Generalizado.

(I) Para o primeiro neurônio $(j = 0)$:

- A Equação (6.91) reduz-se ao caso de um único neurônio (examinado anteriormente).
- O 1º neurônio a convergir é aquele associado ao maior auto-valor.
- A rede extrai o 1º componente principal dos vetores de dados de entrada $\underline{x}(n)$.

(II) Para o segundo neurônio $(j = 1)$:

- A Equação (6.92) torna-se

$$x_i'(n) = x_i(n) - w_{0i}(n)y_0(n). \qquad (6.93)$$

- Desde que o 1º neurônio já tenha convergido para o 1º componente principal, o 2º neurônio vê vetores de entrada $\underline{x}'(n)$ dos quais o 1º auto-vetor da matriz C_x já foi extraído (deflacionado), conforme Equação (6.90).
- O 2º neurônio extrairá, portanto, o 1º componente principal de $\underline{x}'(n)$, que é, na verdade, o 2º componente principal do espaço vetorial de entrada original dos vetores $\underline{x}(n)$ (2º auto-valor e correspondente auto-vetor da matriz C_x).

(III) Para os demais neurônios da rede:

- Cada conjunto de pesos sinápticos convergido representa um auto-vetor da matriz de correlação do conjunto de dados de entrada (espaço vetorial de entrada).
- Os auto-vetores obtidos desta forma encontram-se ordenados em ordem decrescente de valor dos auto-valores associados.

Exemplo 5 – Compressão de imagens através do GHA

➔ O **Algoritmo Hebbiano Generalizado** será utilizado para treinar uma RNA progressiva, composta por uma única camada de neurônios lineares, com o objetivo de proceder à **DSE (ou PCA) de um conjunto de vetores de dados** originado da vetorização dos pixels de uma imagem *grayscale*.

➔ o **Processo de compressão** resultante da DSE obtida através do Algoritmo Hebbiano Generalizado – GHAPCA *(Generalized Hebbian Algorithm to perform Principal Component Analysis*) é similar ao obtido pela **Karhunen-Loève Transform (KLT), mas sem a necessidade da determinação prévia da matriz de covariância e sem a necessidade de obter todos os auto-vetores.**

- ⇨ A matriz de pixels que representa a imagem é particionada em blocos (submatrizes).
- ⇨ As submatrizes são transformadas em vetores linha, que formarão o conjunto de treino da RNA.
- ⇨ A RNA é treinada sob a regra desenvolvida para o GHAPCA, tendo por objetivo extrair os componentes principais da imagem.

- ✧ A compressão obtida pelo método resulta do número de componentes principais da imagem que serão descartadas.
- ✧ Quanto mais componentes descartadas, maior será a compressão e menor a fidelidade da imagem reconstruída, com respeito à imagem original. Este compromisso é regido pelo tipo de aplicação a que se destina a compressão.

- ⇨ A RNA terá tantos neurônios quantas componentes principais se desejar preservar.
- ⇨ Após o treinamento, a rede terá armazenado em seus pesos sinápticos os auto-vetores correspondentes aos auto-valores da matriz covariância do conjunto de dados de entrada que se especificou preservar.

- ✧ A convergência do algoritmo baseia-se em um critério de parada baseado na estabilização da Norma euclidiana (módulo) dos vetores estimados obtidos após um número suficientemente grande de iterações.

- ⇨ Duas otimizações são propostas para o algoritmo, objetivando redução de tempo computacional:
  - ♦ o critério para atualização da razão de aprendizado;
  - ♦ a aceleração de convergência por janelamento de treino.

- ✧ A partir dos auto-vetores obtidos e das saídas dos neurônios após a convergência do algoritmo, para cada vetor pertencente ao conjunto de treino é obtida a reconstrução da imagem, estimada a partir das componentes principais identificadas.

- ➔ A fidelidade da imagem reconstruída pelo algoritmo é determinada pelo parâmetro Relação Sinal-Ruído de Pico.

A Representação dos Dados

Quando aplicado à compressão de imagens digitais, o conjunto de dados a ser apresentado à RNA é composto de um conjunto de vetores originado da vetorização dos elementos da imagem.

- ✧ Seja uma imagem digital, bidimensional, representada por uma matriz $f(x, y)$ de $N \times N$ posições, sendo N uma potência inteira de dois.
- ✧ Cada um dos elementos da imagem assume valores entre 0 e 255 (imagem em *grayscale*, quantizada em 256 níveis de cinza).
- ✧ Antes de qualquer processamento, os valores de pixel da imagem são normalizados para que o maior valor de pixel seja unitário.

- Os dados são apresentados à camada de entrada da RNA sob a forma de vetores unidimensionais de p elementos.
- Cada um destes vetores é obtido do particionamento da imagem em submatrizes não sobrepostas (ou blocos), de dimensões $l \times l$, onde $l = 2^n$ elementos (n é uma potência inteira de dois e é determinado de acordo com as características estatísticas da imagem).
- O número de elementos dos vetores unidimensionais é, portanto, $p = l^2$. (As imagens utilizadas neste trabalho possuem 128×128 elementos.)

- Em imagens, *pixels* vizinhos são, em geral, correlacionados.
- À medida que as distâncias entre pixels ↑, a correlação ↓ substancialmente.
- Portanto, o tamanho escolhido para as submatrizes deve ser tal que permita captar elementos suficientemente correlacionados.

Sendo originários de pixels correlacionados, os vetores gerados das submatrizes, quando apresentados à RNA, possibilitam uma boa estimativa dos componentes principais do conjunto de dados representativo da imagem.

- **"Ordem e estrutura nos padrões de ativação representam a informação redundante que é adquirida pela rede neural na forma de conhecimento, a qual é pré-requisito necessário ao aprendizado auto-organizado"** (4º princípio da auto-organização de Barlow).
- Para que o aprendizado auto-organizado possa desempenhar uma função útil de processamento de informação é necessário que haja redundância nos padrões de ativação supridos à rede pelo ambiente.

- Uma submatriz de 8×8 elementos foi determinada experimentalmente como a que provê melhores resultados para as imagens utilizadas de 128×128 pixels.
- Submatrizes maiores que 8×8 implicam em uma maior diversidade de variações por vetor de treino (menor correlação), tornando necessário aumentar o número de componentes principais utilizadas para que se obtenha a mesma *PSNR*.
- Submatrizes menores que 8×8 pixels têm seus elementos consideravelmente correlacionados – o que melhora a *PSNR* – no entanto, a taxa de compressão cai devido ao maior número de vetores utilizados para representar a imagem.

➔ As dimensões da matriz representativa da imagem são múltiplos inteiros das dimensões das submatrizes de dimensões $l \times l$ consideradas.

➔ Do particionamento resultam $n_b = (N/l)^2$ submatrizes.

➔ O nº n_b de submatrizes resultantes do particionamento da matriz define o nº de vetores que serão utilizados para treino da RNA.

➔ A apresentação de todo o conjunto de vetores à RNA, uma vez, corresponde a uma época.

No exemplo em questão, $N = 128$ e $l = 8$. Portanto, o conjunto de treino apresentado à RNA é composto de

$$n_b = \left(\frac{N}{l}\right)^2 = \left(\frac{128}{8}\right)^2 = 256 \quad \text{vetores } \underline{x}(n) \in \Re^p, \quad p = l^2 = 64 \tag{6.94}$$

As n_b submatrizes são transformadas em vetores linha $\underline{x}(n) \in \Re^p$. O n-ésimo vetor linha é obtido de acordo com a seguinte lei de formação:

(1) Os l primeiros elementos do vetor linha $\underline{x}(n) \in \Re^p$ são os pertencentes à primeira linha da submatriz de $l \times l$ elementos, lida da esquerda para a direita.

(2) Os próximos l elementos deste vetor linha são os pertencentes à segunda linha da submatriz, lida da mesma forma.

(3) Assim prossegue-se, até que sejam esgotados todos os l^2 elementos da submatriz $l \times l$.

- Para efetuar PCA sobre um espaço vetorial é necessário que o conjunto de vetores $\underline{x}(n) \in \Re^p$ possuam média zero. Portanto:

- Obtém-se o vetor média, resultante da média dos $n_b = (N/l)^2$ vetores e subtrai-se este vetor de cada vetor do conjunto.Os vetores centralizados assim obtidos constituirão o conjunto de treino da RNA.

- Cada vetor apresentado à RNA constitui uma amostra do conjunto de treino, em nosso caso, composto de 256 vetores de 64 elementos.

- A cada época, a ordem de apresentação dos vetores do conjunto de treino é mudada aleatoriamente, de forma semelhante ao ato de embaralhar um conjunto de cartas.
- São justificativas e características do ato de embaralhamento:
  - ✧ Os vetores de dados são apresentados à RNA em ordem aleatória, para impedir que a apresentação ordenada destes vetores gere um ciclo de aumento e decréscimo de valores dos pesos sinápticos.
  - ✧ Este ciclo poderá interferir no aprendizado da RNA, dificultando a convergência dos pesos sinápticos para os auto-vetores da matriz covariância dos dados.
  - ✧ O embaralhamento é implementado por um algoritmo que escolhe aleatoriamente dois vetores do conjunto, permutando-os entre si.
  - ✧ Esta operação é executada ao final de cada época, tantas vezes quantos forem o nº de vetores de dados.
  - ✧ A função densidade de probabilidade do gerador de números aleatórios utilizado é uniforme.

A Estrutura da RNA

★ RNA progressiva, com uma única camada de neurônios lineares, a ser treinada pelo GHAPCA.

★ Observe que a camada de entrada da RNA possui p nós.

★ Estes p nós recebem os elementos dos vetores de dados do conjunto de treino obtido do particionamento da matriz representativa da imagem.

★ Os vetores são compostos por $p = l^2$ elementos.

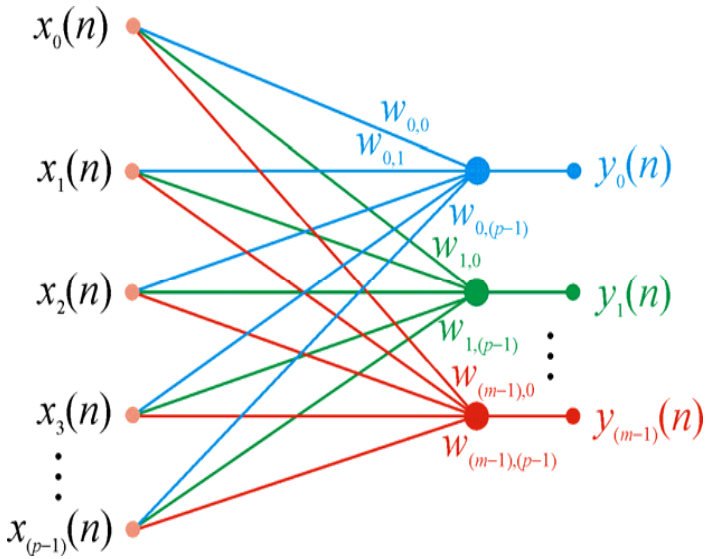


Figura 8

★ A camada de entrada da RNA possui, portanto, $p = l^2$ elementos.

★ Cada vetor representativo dos dados de entrada é da forma:

$$\underline{x} = [x_0 \; x_1 \; \ldots \; x_{p-1}]^T \qquad (6.95)$$

★ A camada de saída é formada por m neurônios, representados por:

$$\underline{y} = [y_0 \; y_1 \; \ldots \; y_{m-1}]^T \qquad (6.96)$$

★ Os pesos sinápticos que conectam os nós fonte i aos neurônios j da camada de saída são representados por

$$w_{ji}, \quad \begin{matrix} i = 0, 1, \ldots, p-1 \\ j = 0, 1, \ldots, m-1 \end{matrix} \qquad (6.97)$$

Após a recorrente apresentação do conjunto de vetores de treino $\underline{x}(n) \in \Re^p$ até a convergência do algoritmo GHA, o vetor $\underline{w}_j \in \Re^p$ (cujos elementos $w_{j\,i}$ representam as transmitâncias das sinapses do *j*-ésimo neurônio) "coalesce" para o auto-vetor $\underline{e}_j \in \Re^p$ (cujos elementos são os valores $e_{j\,i}$) associado ao auto-valor λ_j da matriz covariância dos dados de entrada.

λ_j é determinado através da média do quadrado das normas Euclidianas das projeções dos vetores $\underline{x}(n) \in \Re^p$ sobre o auto-vetor $\underline{e}_j \in \Re^p$.

Especificamente:

- O auto-vetor $\underline{e}_0$, associado ao maior auto-valor λ_0, terá seus elementos representados pelos elementos w_{0i} , $i = 0, 1, ..., p-1$, do vetor de pesos sinápticos $\underline{w}_0$, associado ao neurônio $j = 0$ da camada de saída (1° neurônio da camada de saída).
- O neurônio $j = 1$ (2° neurônio da camada de saída) conterá, nos elementos w_{1i} , $i = 0, 1, ..., p-1$, de seu vetor de pesos sinápticos associado $\underline{w}_1$, os elementos e_{1i} , $i = 0, 1, ..., p-1$, do segundo auto-vetor $\underline{e}_1$, associado ao segundo maior auto-valor e assim, sucessivamente até o último neurônio, $j = y_{m-1}$.

- Para o exemplo de compressão da imagem em questão, a imagem tem um tamanho $N \times N$ pixels e os blocos (submatrizes) são de tamanho $l \times l$, sendo $N = 128$ e $l = 8$. Neste contexto, a Figura 9 apresenta a representação gráfica dos pesos sinápticos (auto-vetores) "coalescidos" nos vetores das sinapses dos neurônios da camada de saída da RNA, após a convergência do algoritmo GHA.

- A RNA da figura possui $p = l^2 = 64$ nós computacionais na camada de entrada e são considerados *k* auto-vetores ($k \leq p$) para a reconstrução dos dados.

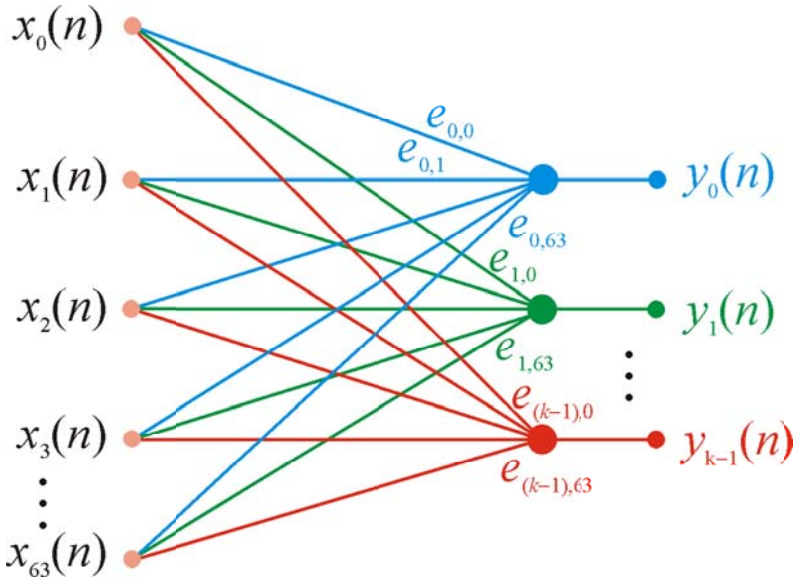


Figura 9:

Representação gráfica da RNA com:

- *p*=64 nós de entrada
- *k* neurônios na camada de saída,

após a convergência do GHA.

★ A saída $y_0(n)$ do neurônio da camada de saída da RNA para a iteração *n*, produzida em resposta às entradas $x_i(n)$, componentes do vetor $\underline{x}(n)$, é: $y_0(n) = \underline{w}_0(n)^T \underline{x}(n) = \underline{x}(n)^T \underline{w}_0(n) = \sum_{i=0}^{p-1} w_{0i}(n) x_i(n)$ (6.98)

★ A regra de Aprendizado Hebbiano, após deflacionada de um fator $\eta\, y_0^2(n) \underline{w}_0(n)$, pode ser expressa através da Equação (6.99):

$$\Delta w_0(n) = \eta\ y_0(n) \underline{x}(n) - \eta\, y_0^2(n) \underline{w}_0(n) \qquad (6.99)$$

★ Partindo das Equações (6.98) e (6.99), a regra generalizada fica definida pelas Equações (6.100) e (6.101), aqui apresentadas na forma expandida.

$$y_j(n) = \underline{w}_j(n)^T \underline{x}(n) = \sum_{i=0}^{p-1} w_{ji}(n) x_i(n), \qquad j = 0, 1, ..., m\text{-}1 \tag{6.100}$$

A atualização dos pesos sinápticos é dada pela Equação (6.101):

$$w_{ji}(n+1) = w_{ji}(n) + \eta \; y_j(n) \left\{ x_i(n) - \sum_{k=0}^{j} w_{ki}(n)\, y_k(n) \right\}, \quad \begin{matrix} i = 0, 1, \cdots, p\text{-}1 \\ j = 0, 1, \cdots, m\text{-}1 \end{matrix} \tag{6.101}$$

★ Sob esta regra de aprendizado, os *k* vetores peso sinápticos convergem para os *k* auto-vetores correspondentes aos *k* maiores auto-valores da matriz de covariância dos dados de entrada, após um número de iterações *n* suficientemente grande.

O Critério de Parada

- O critério de parada adotado para a convergência do algoritmo é derivado das características esperadas dos vetores estimados obtidos.
- A convergência do algoritmo ocorre quando os pesos sinápticos armazenados na RNA estabilizam em valores que correspondem aos auto-vetores associados aos auto-valores da matriz Covariância dos dados de entrada.
- Então, no equilíbrio , quando $n \to \infty$, é correto assumir que

$$E\{\Delta w_{ji}(n)\} = 0 \tag{6.102}$$

- Baseado nesta propriedade, pode-se considerar que os pesos sinápticos do neurônio *j* tenham convergido para o auto-vetor associado, quando as razões entre todos os seus *p* pesos sinápticos, de uma iteração para a próxima, sejam unitárias, isto é,

$$(w_{ji}(n+1)/w_{ji}(n)) \approx 1,\ i = 0, 1, \ldots, p-1.$$

- Para evitar efeitos de variação transiente em $w_{ji}(n)$ a cada iteração (devida ao uso eventual de uma alta razão de aprendizado), e no intuito de simplificar o critério de convergência, utilizou-se a média em três iterações da razão $\left\|\underline{w}_j(n+1)\right\| / \left\|\underline{w}_j(n)\right\|$, onde $\underline{w}_j$ representa o vetor de *p* pesos sinápticos do neurônio *j*. Especificamente, se

$$\frac{1}{3}\left\{\frac{\left\|\underline{w}_j(n+1)\right\|}{\left\|\underline{w}_j(n)\right\|} + \frac{\left\|\underline{w}_j(n+2)\right\|}{\left\|\underline{w}_j(n+1)\right\|} + \frac{\left\|\underline{w}_j(n+3)\right\|}{\left\|\underline{w}_j(n+2)\right\|}\right\} - 1 < 1 \times 10^{-4} \tag{6.103}$$

considera-se que o vetor peso sináptico do neurônio *j* tenha convergido para o auto-vetor associado.

- O limiar de 1×10^{-4} foi determinado experimentalmente.

O Critério para atualização da razão de aprendizado

★ É conveniente, para a convergência do algoritmo, que a razão de aprendizado seja inicializada com um valor baixo (da ordem de 1×10^{-9}, por exemplo).

★ Após a convergência do primeiro auto-vetor, no entanto, a razão de aprendizado deve ser aumentada, pois, como já ocorreu a deflação da primeira componente principal do vetor de treino, o processo de aprendizado pode ser acelerado sem detrimento da convergência.

★ O critério adotado baseia-se na proposta de Chen e Chang em [13]. A razão de aprendizado é feita, a cada época, igual ao inverso do auto-valor instantâneo dividido por uma constante arbitrária α que varia com o tipo de imagem.

$$\eta = \frac{1/\lambda}{\alpha} \qquad (6.104)$$

★ Experimentalmente, determinou-se que a constante α deve estar contida no intervalo [500, 2000].

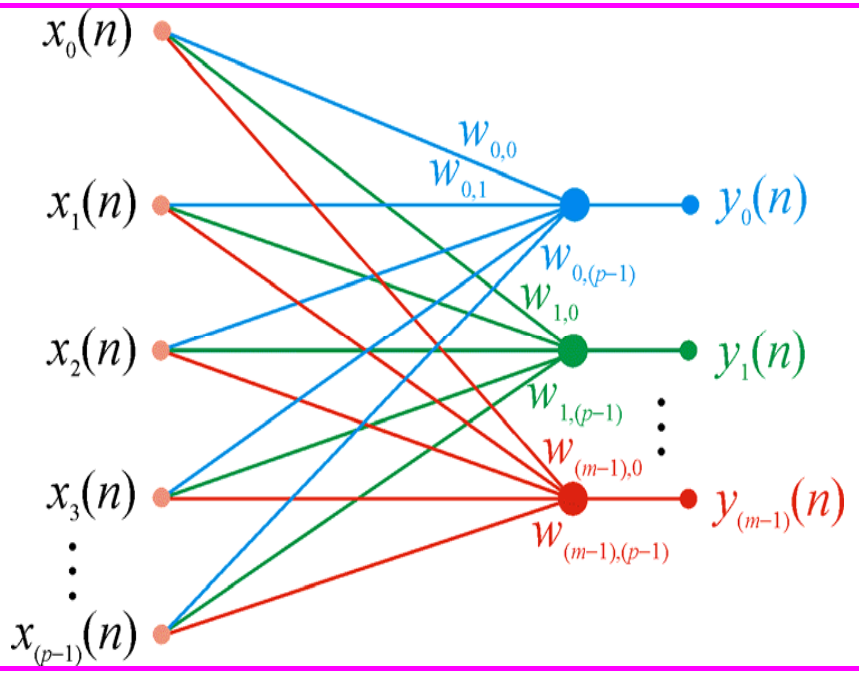
★ Um valor baixo para α acelera a convergência do algoritmo. No entanto, valores inferiores a 500 podem conduzir a *overflow* das variáveis de ponto flutuante de 8 bytes utilizadas.

★ Um valor muito alto para α torna a razão de aprendizado demasiadamente pequena, retardando a convergência.

A Aceleração de convergência do GHA por Janelamento de Treino

- À medida que um neurônio converge para um auto-vetor, todos os subseqüentes ajustam-se correspondentemente, obedecendo ao princípio da deflação, conforme Equação (6.101).
- No entanto, é característica do aprendizado Hebbiano que cada um dos *m* auto-vetores efetivamente inicie o próprio processo de convergência após a convergência do auto-vetor anterior, não sendo alterados os pesos sinápticos dos neurônios já convergidos.
- Observou-se experimentalmente que somente estão em um processo de ajuste coerente no sentido da convergência apenas alguns poucos neurônios imediatamente subseqüentes ao que já convergiu, permanecendo os demais neurônios em um processo incoerente de ajuste dos respectivos pesos sinápticos que não os leva a aproximar-se da convergência.
- Deixar habilitado o processo de ajuste dos pesos sinápticos para a totalidade de neurônios subseqüentes a um neurônio que já convergiu implica em um custo computacional desnecessário, já que somente estarão em um processo de ajuste coerente no sentido da convergência aqueles poucos neurônios imediatamente subseqüentes ao que já convergiu.

- Para evitar que todos os m vetores sinápticos (auto-vetores) sejam ajustados durante a deflação de um particular auto-vetor, é proposto para o *GHAPCA* o algoritmo denominado janelamento de convergência.
- O janelamento de convergência é um novo método que considera a convergência de k_{jt} $(k_{jt} < m)$ auto-vetores de cada vez.
- Havendo convergido o primeiro auto-vetor da janela de convergência, a janela se desloca uma posição para a frente.



* Este processo de janelamento é aplicado ao GHA sem detrimento algum para a precisão do processo de convergência.

* Determinou-se experimentalmente que uma janela de convergência que habilita a convergência de três auto-vetores simultaneamente é adequada a maioria dos casos.

* O ganho de tempo de processamento devido à aceleração de convergência do algoritmo por janelamento de treino, nos experimentos realizados neste exemplo (em que é considerada uma janela de convergência de três auto-vetores), é da ordem de

$$t_{cjt} \approx \frac{t_c}{10} \tag{6.105}$$

onde t_{cjt} denota o tempo de processamento com aceleração do algoritmo por janelamento de treino e t_c o tempo de processamento que considera o ajuste de todos os auto-vetores durante a convergência de um auto-vetor.

* Esta relação t_{cjt} é evidentemente dependente do número total de auto-vetores que estão sendo considerados (*m*), o que equivale a dizer que é dependente do número total de componentes principais que estão sendo consideradas.
* Quanto maior for *m*, maior será a redução de tempo computacional resultante do janelamento de treino.

O Treinamento da RNA

* A cada época de treino são apresentados à RNA as n_b submatrizes (ou blocos) da imagem, sob a forma de vetores de treino. Portanto, uma época é constituída de n_b iterações.

* A cada iteração *n*, $n = 0, 1, ..., n_b - 1$, é apresentado o *n-ésimo* vetor do conjunto de treino à RNA , produzindo a saída *y(n)*.

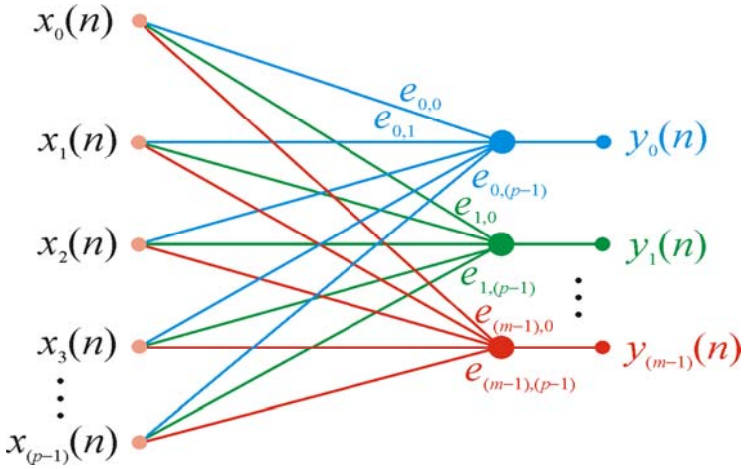
* Após a convergência, a RNA apresenta um conjunto de pesos sinápticos constituído por *m* vetores de *p* elementos, equivalente aos auto-vetores associados aos auto-valores da matriz covariância dos dados de entrada, denotados por $\underline{e}_j$ ($\underline{e}_j$ (equilíbrio)$= \underline{e}_j$).

* A ordem de convergência dos pesos sinápticos é tal que, nos pesos sinápticos associados ao primeiro neurônio da camada de saída da RNA $(\underline{w}_0(n))$ estão as componentes do auto-vetor $(\underline{e}_0)$ associado ao maior auto-valor (λ_0) da matriz de covariância dos dados de entrada.

* Nos pesos sinápticos associados ao segundo neurônio da camada de saída da RNA $(\underline{w}_1(n))$ estão as componentes do auto-vetor $(\underline{e}_1)$ associado ao segundo maior auto-valor (λ_1) da matriz covariância dos dados de entrada e, assim, em ordem decrescente até o último peso sináptico $(\lambda_0 \Leftrightarrow \underline{e}_0,\ \lambda_1 \Leftrightarrow \underline{e}_1,\ \lambda_2 \Leftrightarrow \underline{e}_2,\ \cdots,\ \lambda_j \Leftrightarrow \underline{e}_j)$.

A Reconstrução dos dados

A Figura 10 abaixo apresenta uma RNA já convergida para os auto-vetores $\underline{e}$, com *p* nós na camada de entrada e *m* neurônios na camada de saída. Está sendo apresentada à rede a *n-ésima* submatriz que compõe a imagem, representada pelo vetor $\underline{x}(n)$.



A reconstrução dos dados é obtida por $\hat{x}_i(n) = \sum_{j=0}^{m-1} y_j(n)\, e_{ji}$ (6.106)

Isto é, efetua-se a combinação linear das m projeções $y_j(n)$ do vetor $\underline{x}(n)$ em questão sobre cada auto-vetor $\underline{e}_j$.

Note que $\hat{x}_i(n)$ representa a reconstrução do *i-ésimo* componente do vetor $\underline{x}(n)$ associado ao bloco ou submatriz de ordem *n* da imagem estimada.

Por exemplo, na reconstrução da submatriz $l \times l$ para $n = 0$ representada pelo vetor $\underline{x}(0)$, as componentes $\hat{x}_i(0)$ do vetor $\underline{\hat{x}}(0) = [\hat{x}_0(0) \quad \cdots \quad \hat{x}_{p-1}(0)]$ com $i = 0, 1, \ldots, p-1$ são reconstruídas a partir das projeções $y_j(0)$ pela combinação linear

$$\underline{\hat{x}}(0) = \sum_{j=0}^{m-1} y_j(0)\underline{e}_j \,.$$

Ou, especificamente em termos das componentes $\hat{x}_i(0)$:

$$\begin{aligned}
\hat{x}_0(0) &= y_0(0)\, e_{00} + y_1(0)\, e_{10} + \cdots + y_{m-1}(0)\, e_{(m-1)0} \\
\hat{x}_1(0) &= y_0(0)\, e_{01} + y_1(0)\, e_{11} + \cdots + y_{m-1}(0)\, e_{(m-1)1} \\
&\cdots \\
\hat{x}_{p-1}(0) &= y_0(0)\, e_{0(p-1)} + y_1(0)\, e_{1(p-1)} + \cdots + y_{m-1}(0)\, e_{(m-1)(p-1)}
\end{aligned} \qquad (6.107)$$

Ao *n-ésimo* vetor reconstruído, $\hat{\underline{x}}(n)$, com $n = 0, 1, ..., n_b - 1$, é acrescentado o vetor média previamente extraído.

O vetor resultante $\hat{\underline{x}}(n)$ é transformado na *n-ésima* submatriz (ou bloco) componente da imagem estimada, de acordo com o seguinte algoritmo:

- A primeira linha da *n-ésima* submatriz (vista aqui como uma matriz de $l \times l$ elementos) é formada pelos *l* primeiros elementos do *n-ésimo* vetor.
- A segunda linha da submatriz é formada pelos segundos *l* elementos do *n-ésimo* vetor.
- Procede-se desta forma até que todos os $p = l^2$ elementos do *n-ésimo* vetor sejam transferidos para a *n-ésima* submatriz.

O conjunto de n_b submatrizes irá compor a imagem, sendo $N \times N$ o tamanho da imagem original.

A Compressão dos Dados

- A compressão decorrente do uso do método é função do número de componentes principais da imagem que vierem a ser descartadas.
- Quanto maior o número de componentes principais descartadas, maior a compressão e menor a fidelidade da imagem reconstruída com relação à imagem original.

- A compressão é definida a partir da equação de reconstrução dos dados (Equação 6.106).
- Para reconstrução completa dos dados, todos os *m* auto-vetores (associados aos *m* componentes principais) são considerados.
- A compressão será obtida se, ao invés de *m* auto-vetores, forem considerados *k* auto-vetores, com $k < m$.
- Para exemplificar, considere-se a Figura 11.

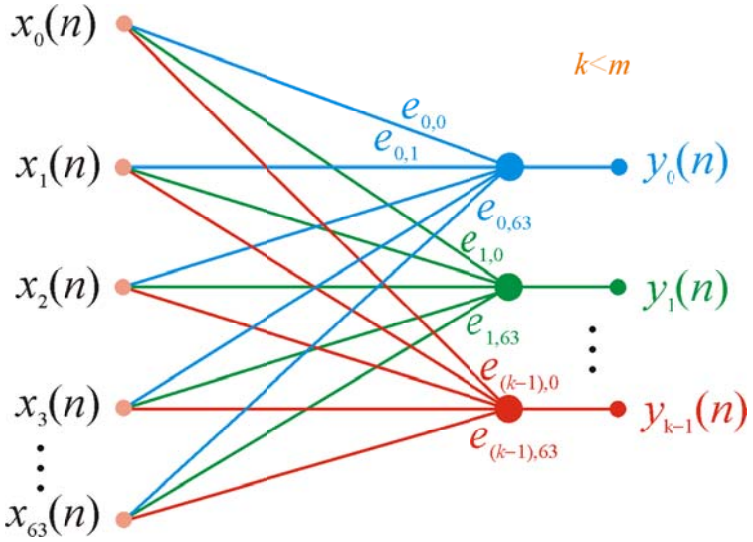


Figura 11: Modelo de RNA utilizada para compressão. A rede apresenta *p*=64 nós de entrada (para submatrizes com $p = 64$ elementos) e *k* neurônios na camada de saída (para extrair *k* componentes principais).

- Na rede mostrada na Figura 11, as componentes do vetor $\underline{\hat{x}}(0)$, $\hat{x}_i(0)$, com $i = 0, 1, ..., p-1$ são reconstruídas por:

$$\hat{x}_0(0) = y_0(0)\, e_{00} + y_1(0)\, e_{10} + \cdots + y_{k-1}(0)\, e_{(k-1)0}$$
$$\hat{x}_1(0) = y_0(0)\, e_{01} + y_1(0)\, e_{11} + \ldots + u_{k-1}(0)\, e_{(k-1)1}$$
$$\ldots \qquad (6.108)$$
$$\hat{x}_{p-1}(0) = y_0(0)\, e_{0(p-1)} + y_1(0)\, e_{1(p-1)} + \cdots + y_{k-1}(0)\, e_{(k-1)(p-1)}$$

- Como $k < m$, as componentes do vetor $\underline{\hat{x}}(0)$, $\hat{x}_i(0)$, com $i = 0, 1, ..., p-1$, sofrem efetiva redução dimensional.

- Os auto-vetores $\underline{e}_{ji}$, com $j = k, k+1, \cdots, m$ - 1 não considerados na reconstrução de $\underline{\hat{x}}(0)$ e responsáveis pela redução dimensional, não representam perda considerável de informação por estarem associados aos auto-valores de menor energia da matriz covariância do conjunto de dados.

- Se a arquitetura da rede for tal que possua *k* neurônios na camada de saída, cada um com $p = l^2$ sinapses, após a convergência do algoritmo a rede apresentará *k* vetores de *p* elementos (referentes aos auto-vetores convergidos) e n_b vetores de *k* elementos (referentes aos $n_b = (N/\, l)^2$ conjuntos de valores de saída dos *k* neurônios da camada de saída da rede).

- A partir destes elementos e do vetor média previamente extraído (composto de *p* elementos) será reconstruída a imagem comprimida.

➔ A razão entre o número de unidades de armazenamento necessárias para representar a imagem original e o número de unidades de armazenamento necessárias para representar a imagem comprimida é chamada de razão de compressão e é denotada por ρ. A razão de compressão pode, então, ser determinada por:

$$\rho = \frac{k\,(\,p + n_b\,) + p}{N \times N}, \text{ com } n_b = (N/\,l\,)^2 \qquad (6.109\ )$$

➔ O método permite atingir uma compressão efetiva maior do que a compressão expressa pela razão de compressão ρ, através da adoção de um critério para alocação dos bits necessários para o armazenamento da imagem comprimida.

Observação:

Neste estudo a compressão é avaliada através da razão de compressão ρ. No entanto, são sugeridos dois critérios para otimizar a compressão, baseados na alocação de diferentes números de bits por unidade de armazenamento da imagem comprimida:

(I) No primeiro critério a otimização é alcançada através da atribuição de um maior número de bits para o armazenamento dos auto-vetores mais significativos da imagem (associados aos maiores auto-valores) e de um menor número de bits para os auto-vetores menos significativos (associados aos menores auto-valores).

(II) A segunda sugestão é a utilização do Código Huffman que é baseado na entropia da imagem comprimida.

Avalia-se a fidelidade dos resultados obtidos pelo *GHAPCA*, através da determinação da Relação Sinal-Ruído de Pico (*PSNR*), definida pela Equação (6.47).

A Figura 12 mostra novamente a imagem em *grayscale* "Lenna" de tamanho 128×128 pixels, utilizada no Exemplo 4, em que a compressão foi resultante da Transformada Karhunen-Loève.

A Figura 13 mostra a imagem comprimida resultante, formada a partir dos 32 primeiros auto-vetores, associados aos 32 auto-valores mais significativos; isto é, a RNA utilizada para treinamento pelo GHAPCA para extração dos componentes principais da imagem é composta de 32 neurônios na camada de saída.

A Figura 14 mostra a imagem comprimida resultante, formada a partir dos 16 primeiros auto-vetores, associados aos 16 auto-valores mais significativos; caso em que a RNA apresenta 16 neurônios na camada de saída.

Figura 12: Imagem *grayscale* "Lenna" de tamanho 128×128 pixels.

Figura 13:

Imagem da Figura 12 comprimida através do GHAPCA, considerando $k = 32\,(k < m)$ neurônios na camada de saída da RNA.

O coeficiente de compressão resultante é $\rho = 0.632812$.

A fidelidade da imagem comprimida com relação à original é dada por $\mathrm{PSNR} = 37.3\,\mathrm{dB}$.

Figura 14:

Imagem da Figura 12 comprimida através do GHAPCA, considerando $k = 16\,(k < m)$ neurônios na camada de saída da RNA.

O coeficiente de compressão resultante é $\rho = 0.316406$.

A fidelidade da imagem comprimida com relação à original é dada por $\mathrm{PSNR} = 31.4\,\mathrm{dB}$.

Sumário do GHAPCA aplicado à compressão de imagens digitais

I – Inicialização:
1. A imagem é normalizada para que o maior valor de pixel seja 1 e o menor valor de pixel seja zero.
2. A imagem é particionada em submatrizes não sobrepostas, de dimensões $l \times l$, onde $l = 2^n$ elementos (*n* é uma potência inteira de dois e é determinado de acordo com as características estatísticas da imagem). Os dados são, portanto, apresentados à RNA sob a forma de vetores unidimensionais de *p* elementos, $p = l^2$. As imagens utilizadas neste trabalho possuem 128×128 elementos.
3. Determina-se o vetor média do conjunto de vetores unidimensionais.
4. Subtrai-se o vetor média do conjunto de vetores, formando o conjunto de treino a ser apresentado à RNA.
5. Inicializa-se os pesos sinápticos de maneira aleatória (distribuição de probabilidade uniforme) com valores pertencentes ao intervalo $\left[-\frac{r}{p}, \frac{r}{p} \right]$, sendo *p* o número de pesos sinápticos por neurônio e *r* um parâmetro a ser experimentalmente estipulado, de acordo com o conjunto de treino da RNA.
6. Arbitra-se uma razão de aprendizado inicial (usualmente no intervalo $[1 \times 10^{-9}, 5 \times 10^{-9}]$).

	II - Treinamento:
1.	Apresenta-se o conjunto de treino à RNA.
2.	A cada vetor apresentado, atualiza-se os pesos sinápticos de acordo com a regra de aprendizado do *GHAPCA*.
3.	Procede-se desta maneira até que todos os vetores de treino sejam apresentados à RNA, o que constitui uma época de treino.
4.	Ao final de cada época embaralha-se o conjunto de vetores de treino de maneira aleatória (distribuição de probabilidade uniforme).
5.	Determina-se o auto-valor λ_j do neurônio que está convergindo, neurônio que é associado ao maior auto-valor na janela de treino. λ_j é determinado através da média do quadrado das normas Euclidianas das projeções dos vetores $\underline{x}(n) \in \Re^p$ sobre o auto-vetor $\underline{e}_j \in \Re^p$.
6.	Determina-se a nova razão de aprendizado $\eta = \frac{1/\lambda}{\alpha}$, proporcionalmente inversa do auto-valor (Equação 6.104).
7	Atualiza-se a razão de aprendizado deste neurônio e dos próximos incluídos na janela de treino com a nova razão de aprendizado.
8.	Os passos (5) a (7) são repetidos até a convergência do neurônio associado ao maior auto-valor na janela de treino, após o que, a janela de treino é deslocada um neurônio para a frente.
9.	Repete-se os passos (5) a (8) até que todos os neurônios convirjam.
10.	Reconstrói-se a imagem a partir dos componentes estimados pela RNA (auto-vetores e saídas dos neurônios da RNA).
11.	Soma-se a todos os vetores reconstruídos o vetor média previamente subtraído.
12.	A imagem é desnormalizada para que o maior valor de pixel seja 255 e o menor valor de pixel seja zero.