3
Concepção Técnica

3.1.
Análise: Descrição do Sistema

O sistema proposto baseia-se na cooperação entre técnicas já usadas na modelagem de sistemas não lineares. Sabe-se que não existe uma só técnica que possa modelar uma ampla variedade de sistemas não–lineares. Por esse motivo o sistema proposto tenta obter informação relevante de cada uma das técnicas utilizadas, de forma que cada uma destas possa ajudar as demais em um processo de cooperação.

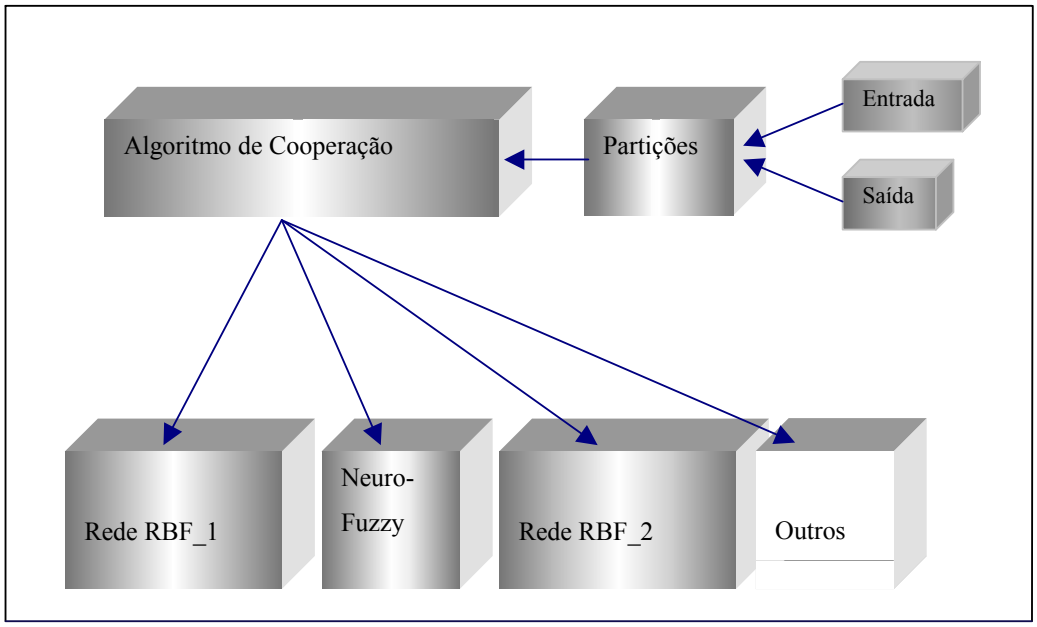

Figura 2 Diagrama em blocos do algoritmo

A Figura 2 ilustra os diversos componentes do sistema de identificação proposto. Como pode ser observado, nela temos indicados blocos de dados dos sinais de entrada e saída, à disposição do sistema, prontos para serem usados durante o treinamento.

O bloco seguinte, denominado "Partições", é encarregado de realizar uma análise do comportamento dos sinais de entrada e saída. Estes sinais serão representados pelos seus respectivos autovalores e autovetores através da técnica SVD, de uma forma particular a ser descrita na seqüência do trabalho. Ver-se-á que esta técnica permite distinguir melhor as diversas regiões dentro dos espaços dos sinais, que apresentem comportamentos diferenciados.

O bloco "Algoritmo de Cooperação" é encarregado de combinar as informações, resultantes da aplicação de cada uma das técnicas usadas, submetidas à modelagem das partições. No caso em estudo estas técnicas serão as redes RBF (onde as redes RBF1 e RBF2 na Figura 2, seriam redes com algoritmos distintos para ajustar os parâmetros) e o modelo Neuro-fuzzy.

Esta cooperação, gerenciada pelo Algoritmo de Cooperação, produz ao seu final uma aproximação aos sinais originais. O erro de cada aproximação, resíduo, pode ser recursivamente submetido ao mesmo processo, de tal forma que a energia da sucessão de resíduos se reduz até satisfazer o erro de modelagem esperado, ou alcançar um patamar inferior (quando o método não consegue mais extrair informação dos sinais).

Esta é uma descrição bastante geral do funcionamento dos blocos, que compõe o sistema identificador. Eles serão detalhados mais adiante.

3.1.1.
Técnica de Decomposição do Sinal: SVD

O primeiro passo no tratamento dos dados é realizar partições nos espaços dos sinais de entrada e de saída. Contrariamente às técnicas habitualmente empregadas (Redes neurais), as partições a serem efetuadas neste trabalho terão por base a técnica de decomposição SVD (Singular Value Decomposition) [3].

O comportamento de um sinal pode ser expresso em função de seus autovetores e autovalores, resultantes, por exemplo, do uso do SVD. A decomposição em valores singulares é um dos resultados mais importantes da Álgebra Linear, tanto computacional quanto teórica, tendo encontrado aplicações em praticamente todas as áreas da engenharia e da física. Uma breve descrição da técnica é feita a seguir [12].

SVD convencional:

$$[U,S,V] = SVD(X) \quad (1)$$

As matrizes U e V contém os autovetores à esquerda e direita, respectivamente, da matriz de dados X. A matriz S é uma matriz diagonal contendo os respectivos autovalores. O SVD nos permite expressar sinais representados em forma matricial por seus respectivos autovetores e autovalores associados Entretanto, na medida em que trabalhamos com maior número de amostras na janela de dados o procedimento de decomposição fica cada vez mais lento, podendo limitar a praticidade da técnica. Por isto, em determinadas etapas deste trabalho é usada uma nova versão de SVD [13] muito mais rápida. Ela é tratada em seguida.

3.1.1.1.
SVD Modificado

O custo computacional do SVD é geralmente elevado, levando à busca de técnicas aproximadas no afã de reduzi-lo. A maioria destas técnicas sacrifica exatidão por velocidade. Entretanto, esta nova versão do SVD é bastante menos onerosa e mais rápida que as tradicionais, sem por isso perder exatidão. Sua eficiência baseia-se na semelhança das matrizes, devido à continuidade dos sinais que elas descrevem, o que é particularmente atrativo para o caso em questão. Atrai no novo SVD sua característica de continuidade e razoável imunidade à dimensão da matriz de dados. Uma utilização especial ainda nos permite, se desejado, utilizar técnicas de esquecimento no caso de identificação de sistemas não estacionários.

O SVD rápido apresenta uma forma particular para a construção da matriz de dados. Em vez de, como é hábito, se trabalhar com a matriz de autocorrelação do sinal de entrada é proposto o uso da própria matriz de aquisição de dados, organizada sempre de forma simétrica. Isto implicaria na economia de alguns "flops" durante o processo de decomposição. A matriz dos dados é inicializada pelas primeiras cinco amostras dos dados, aumentando sua dimensão com amostras subseqüentes, até que atinja uma dimensão máxima. A cada par de amostras novas constrói-se a nova matriz de dados, cuja decomposição usa como estado inicial os autovetores e autovalores estimados no passo anterior.

A descrição detalhada do novo SVD [13] consiste num algoritmo rápido para a decomposição de uma matriz quadrada simétrica, baseado no método de Jacobi. Conseqüentemente, é necessário descrever as amostras dos dados no formato de uma matriz simétrica, mantendo a simetria a cada atualização. Como observado, devido à exigência de simetria, a matriz dos dados é inicializada pelas primeiras cinco amostras, aumentando o tamanho com amostras subseqüentes, até uma dimensão limite, pré-estabelecida, M. A partir deste ponto o algoritmo incorpora novas

amostras, descartando as mais antigas, sempre preservando a simetria da matriz e sua dimensão.

A matriz que atualiza o processo é executada para cada par de amostras novas. Conseqüentemente a matriz é atualizada para n = 5, 7, 9... A matriz A_k dos dados de entrada de dimensão kxk, têm a forma:

$$\mathbf{A}_k(n) = \begin{bmatrix} x(n-2k+2) & \cdots & x(n-k+1) \\ \vdots & \ddots & \vdots \\ x(n-k+1) & \cdots & x(n) \end{bmatrix} \tag{2}$$

De acordo com a Equação (2), a primeira matriz dos dados tem a dimensão k=3, contendo as primeiras cinco amostras iniciais. As novas amostras são incorporadas de acordo com a Equação (3), como:

$$\mathbf{A}_{k+1}(n) = \begin{bmatrix} \mathbf{A}_k(n-2) & & x(n-k) \\ & & \vdots \\ x(n-k) & \cdots & x(n) \end{bmatrix} \tag{3}$$

Quando o limite de dimensão (k = M) é alcançado, as matrizes subseqüentes têm que preservar a dimensão constante do limite. Para cada última inclusão de uma linha e coluna, a primeira linha e coluna são eliminadas. Isto é possível de ser executado para cada par de amostras novas. Conseqüentemente, aplicando a Equação (3) para k=M:

$$\mathbf{A}_M(n) = \begin{bmatrix} \mathbf{A}_{M-1}(n-2) & & x(n-M+1) \\ & & \vdots \\ x(n-M+1) & \cdots & x(n) \end{bmatrix} \tag{4}$$

De outra forma, a Equação (3) pode ser reescrita na forma:

$$\mathbf{A}_{k+1}(n) = \begin{bmatrix} \mathbf{A}_k(n-2) & \mathbf{x}_L \\ & \\ \mathbf{x}_L^T & x(n) \end{bmatrix} \tag{5}$$

onde:

$$\mathbf{x}_L = \begin{bmatrix} x(n-k) & \cdots & x(n-1) \end{bmatrix}^T \tag{6}$$

Sendo V_0 o autovetor para a decomposição de $A_k(n-2)$, têm-se:

$$\mathbf{D}_0 = \mathbf{V}_0 \mathbf{A}_k (n-2) \mathbf{V}_0^T \tag{7}$$

onde D_0 é uma matriz diagonal. Define-se V_1 como:

$$\mathbf{V}_1 = \begin{bmatrix} \mathbf{V}_0 & \vdots & \mathbf{0} \\ \cdots & \vdots & \cdots \\ \mathbf{0} & \vdots & 1 \end{bmatrix} \tag{8}$$

e aplicando a $A_{k+1}(n)$, temos:

$$\mathbf{D}_1^q = \begin{bmatrix} \mathbf{V}_0 & \vdots & \mathbf{0} \\ \cdots & \vdots & \cdots \\ \mathbf{0} & \vdots & 1 \end{bmatrix} \begin{bmatrix} \mathbf{A}_k (n-2) & \mathbf{x}_L \\ \mathbf{x}_L^T & x(n) \end{bmatrix} \begin{bmatrix} \mathbf{V}_0^T & \vdots & \mathbf{0} \\ \cdots & \vdots & \cdots \\ \mathbf{0} & \vdots & 1 \end{bmatrix} \tag{9}$$

$$\mathbf{D}_1^q = \begin{bmatrix} \mathbf{D}_0 & \vdots & \mathbf{V}_0 \mathbf{x}_L \\ \cdots & \vdots & \cdots \\ \mathbf{x}_L^T \mathbf{V}_0^T & \vdots & x(n) \end{bmatrix}$$

Em conseqüência, o procedimento rende uma matriz quase-diagonal $\mathbf{D_1}^q$, exceto pela adição da última linha e coluna. Entretanto, embora não diagonal, a matriz é ainda simétrica. O método de Jacobi é feito especialmente para a decomposição deste tipo de matriz simétrica quadrada, com um grande número de zeros fora da diagonal. A Equação (9) indica uma estimativa boa para os autovetores de $A_{k+1}(n)$.

Quando a matriz de aquisição de dados alcança a dimensão M, a estimativa considerada da matriz nova dos autovetores é simplesmente a matriz velha do autovetor. Este tipo de escolha mostrou os efeitos positivos atuais, melhorando a eficiência da avaliação. Observe-se que o método leva naturalmente à continuidade nos seus elementos básicos, induzindo à continuidade na seqüência de autovalores e autovetores.

3.1.2.
Partições

Ao contrário das técnicas usuais de partição, utilizadas em redes neurais, a empregada neste estudo terá por base a decomposição dos espaços de sinal pelo SVD, como dito anteriormente. Como o SVD gera os autovetores referentes ao espaço tratado, ele permite a análise do sinal por seus subespaços, de forma análoga à que já se conhece em tratamento de ruídos e interferências, com excelentes resultados [14].

Em determinadas partes deste trabalho será empregada a versão modificada do SVD, buscando maior simplicidade e velocidade na operação do método, em outras será usada a versão convencional, como será explicado mais adiante.

3.1.3.
Critério das Partições

3.1.3.1.
Obtenção dos autovetores e autovalores dos sinais de entrada e saída

Aplicando o SVD no nosso caso, para cada atualização da matriz de dados de entrada e saída, teremos um novo conjunto de autovetores e autovalores. Estes elementos descrevem um conjunto de trajetórias, mais ou menos bem comportadas, traduzindo de certa forma o comportamento do sinal. Na realidade, os autovetores e os correspondentes autovalores formam, cada par, uma espécie de vetor descritivo, que servirá à análise destas trajetórias. A formação destes vetores característicos é feita a seguir.

Seja um sinal (entrada/saída) descrito pela curva indicada na Figura 3:

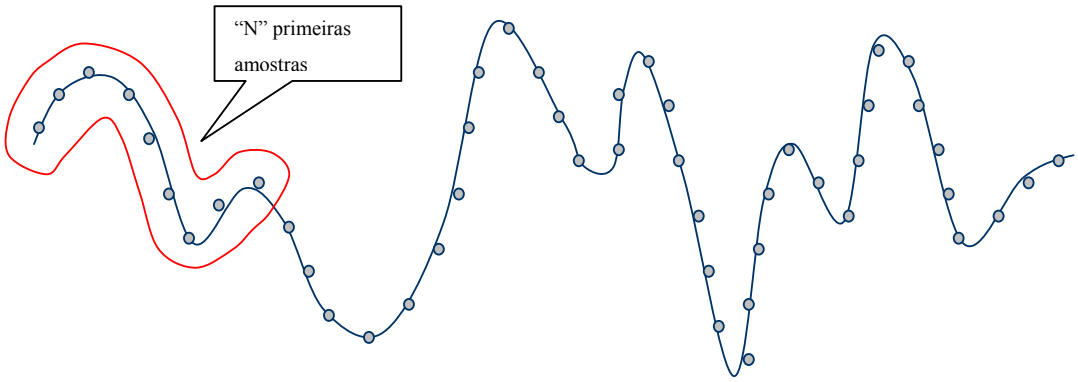

Figura 3 Sinal

As N primeiras amostras deste sinal formam a matriz de dados definida por

$$Matriz\_de\_dados = \begin{bmatrix} m_1 & m_2 & \cdots & m_M \\ m_2 & m_3 & \cdots & m_{M+1} \\ \vdots & \vdots & \cdots & \vdots \\ m_M & m_{M+1} & \cdots & m_{2M-1} \end{bmatrix}_{MxM} \tag{10}$$

onde N=2M-1.

Aplicando o SVD, a esta matriz de dados simétrica, vamos obter uma nova matriz composta pelos autovetores concatenados aos autovalores (vetores característicos) como segue:

$$SVD(Matriz\_de\_dados) \Rightarrow \left[\begin{array}{c|c} \text{autovetor 1} & \text{autovalor 1} \\ \text{autovetor 2} & \text{autovalor 2} \\ \text{autovetor 3} & \text{autovalor 3} \\ \vdots & \vdots \end{array}\right]_{\text{MxL}} \tag{11}$$

Após obter os autovetores e autovalores associados à matriz de dados, será preenchida uma nova matriz de dados, consistindo no deslocamento de uma amostra

dentro da janela de "N" amostras (para quando é aplicado o SVD convencional) ou duas amostras novas (para quando é aplicado a nova versão do SVD). Destaca-se aqui a nova versão do SVD, usada para realizar as partições nos espaços de sinais, enquanto que a versão convencional será usada na decomposição das saídas estimadas por cada uma das técnicas em cada uma das partições. Esta diferença é realçada ao longo da descrição do trabalho.

As equações (12) e (13) mostram as matrizes de dados preenchidas para aplicar o SVD convencional e a nova versão respectivamente.

São supostos conhecidos:

MxM: tamanho da matriz de dados
M: número de autovetores
M_d: matriz de dados
i : índice da amostra do sinal
m1,m2,..mi : amostra 1, amostra 2,.., última amostra

SVD convencional:

$$M\_d_1 = \begin{bmatrix} m1 & m2 & m3 & \cdots \\ m2 & m3 & m4 & \cdots \\ m3 & m4 & m5 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{MxM}$$ Primeira atualização da matriz de dados

$$M\_d_2 = \begin{bmatrix} m2 & m3 & m4 & \cdots \\ m3 & m4 & m5 & \cdots \\ m4 & m5 & m6 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{MxM}$$ Segunda atualização da matriz de dados

Última atualização da matriz de dados

$$M\_d_{i-(M+1)} = \begin{bmatrix} m_{i-(M+1)} & m_{i-(M+1)+1} & m_{i-(M+1)+3} & \cdots & \cdots \\ m_{i-(M+1)+1} & m_{i-(M+1)+2} & m_{i-(M+1)+4} & \cdots & \cdots \\ m_{i-(M+1)+2} & m_{i-(M+1)+3} & m_{i-(M+1)+5} & \cdots & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ \vdots & \vdots & \vdots & \cdots & m_i \end{bmatrix}_{MxM} \quad (12)$$

SVD nova versão:

$$M\_d_1 = \begin{bmatrix} m1 & m2 & m3 & \cdots \\ m2 & m3 & m4 & \cdots \\ m3 & m4 & m5 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{MxM}$$ Primeira atualização da matriz de dados

$$M\_d_2 = \begin{bmatrix} m3 & m4 & m5 & \cdots \\ m4 & m5 & m6 & \cdots \\ m5 & m6 & m7 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{MxM}$$ Segunda atualização da matriz de dados

Última atualização da matriz de dados

$$M\_d_{2(i-2)/M} = \begin{bmatrix} m_{i-(M+1)} & m_{i-(M+1)+1} & m_{i-(M+1)+3} & \cdots & \cdots \\ m_{i-(M+1)+1} & m_{i-(M+1)+2} & m_{i-(M+1)+4} & \cdots & \cdots \\ m_{i-(M+1)+2} & m_{i-(M+1)+3} & m_{i-(M+1)+5} & \cdots & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ \vdots & \vdots & \vdots & \cdots & m_i \end{bmatrix}_{MxM} \quad (13)$$

As equações (14) a (17) mostram a aplicação do SVD convencional às respectivas matrizes de dados, obtendo-se como resultado os autovetores e autovalores associados.

$$SVD(M\_d_1) = \underbrace{U_1}_{\begin{bmatrix} u_1 & u_2 & \cdots & u_M \\ u_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ u_{2M+1} & \cdots & \cdots & u_{MxM} \end{bmatrix}_{MxM}} * \underbrace{S_1}_{\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_M \end{bmatrix}_{MxM}} * \underbrace{V_1'}_{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}}$$

(14)

Sendo que a matriz de dados (M_d1) é simétrica, temos que U=V′, logo somente precisamos da matriz U e os autovalores correspondentes para expressar os vetores característicos como segue:

$$[\text{autovetores}_1][\text{autovalores}_1] = \overset{[U_1]}{\begin{bmatrix} u_1 & u_2 & \cdots & u_M \\ u_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ u_{2M+1} & \cdots & \cdots & u_{MxM} \end{bmatrix}_{MxM}} \overset{\text{autovalores}}{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix}_{Mx1}} \tag{15}$$

$$SVD(M\_d_{i-(M+1)}) = U_{i-(M+1)} * S_{i-(M+1)} * V'_{i-(M+1)}$$

$$SVD(M\_d_{i-(M+1)}) = \overset{U_{i-(M+1)}}{\begin{bmatrix} u_1 & u_2 & \cdots & u_M \\ u_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ u_{2M+1} & \cdots & \cdots & u_{MxM} \end{bmatrix}_{MxM}} \overset{S_{i-(M+1)}}{\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_M \end{bmatrix}_{MxM}} \overset{V'_{i-(M+1)}}{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}} \tag{16}$$

Arrumando as matrizes como segue, supondo que, como a matriz de dados é simétrica, V=U':

$$[\text{autovetores}_{i-(M+1)}][\text{autovalores}_{i-(M+1)}] =$$

$$\overset{[U_{i-(M+1)}]}{\begin{bmatrix} u_1 & u_2 & \cdots & u_M \\ u_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ u_{2M+1} & \cdots & \cdots & u_{MxM} \end{bmatrix}_{MxM}} \overset{\text{autovalores}_{i-(M+1)}}{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_M \end{bmatrix}_{Mx1}} \tag{17}$$

cada linha representa um vetor característico

As equações (18) a (21) mostram a aplicação da nova versão do SVD às matrizes de dados respectivas (seguindo a mesma nomenclatura usada no programa de implementação), obtendo-se como resultado os autovetores e autovalores associados (pressupondo a matriz de dados simétrica e conseqüentemente U=V′).

$$[B_1, V_1] = SVD_{novo}(M\_d_1) \quad \text{onde} \quad V_1 = \overset{\text{autovetores}}{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}}, \quad B_1 = \overset{\text{autovalores}}{\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_M \end{bmatrix}_{MxM}} \tag{18}$$

$$\text{autovetores}_1 \mid \text{autovalores}_1 = \overset{V_1}{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}} \overset{\text{autovalores}_1}{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_M \end{bmatrix}_{Mx1}} \tag{19}$$

$$[B_{2(i-2)/M}, V_{2(i-2)/M}] = SVD_{novo}(M\_d_{2(i-2)/M})$$

$$\text{onde} \quad V_{2(i-2)/M} = \overset{V_{2(i-2)/M}}{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}}, \quad B_{2(i-2)/M} = \overset{B_{2(i-2)/M}}{\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_M \end{bmatrix}_{MxM}} \tag{20}$$

Arrumando as matrizes como segue

$$\text{autovetores}_{2(i-2)/M} \mid \text{autovalores}_{2(i-2)/M} = \overset{V_{2(i-2)/M}}{\begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{M+1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ v_{2M+1} & \cdots & \cdots & v_{MxM} \end{bmatrix}_{MxM}} \overset{\text{autovalores}_{2(i-2)/M}}{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_M \end{bmatrix}_{MxM}} \tag{21}$$

cada linha representa um vetor característico

Logo, para cada atualização da matriz de dados vamos ter um conjunto de autovalores e autovetores (vetores característicos) associados tal como se mostra na Figura 4.

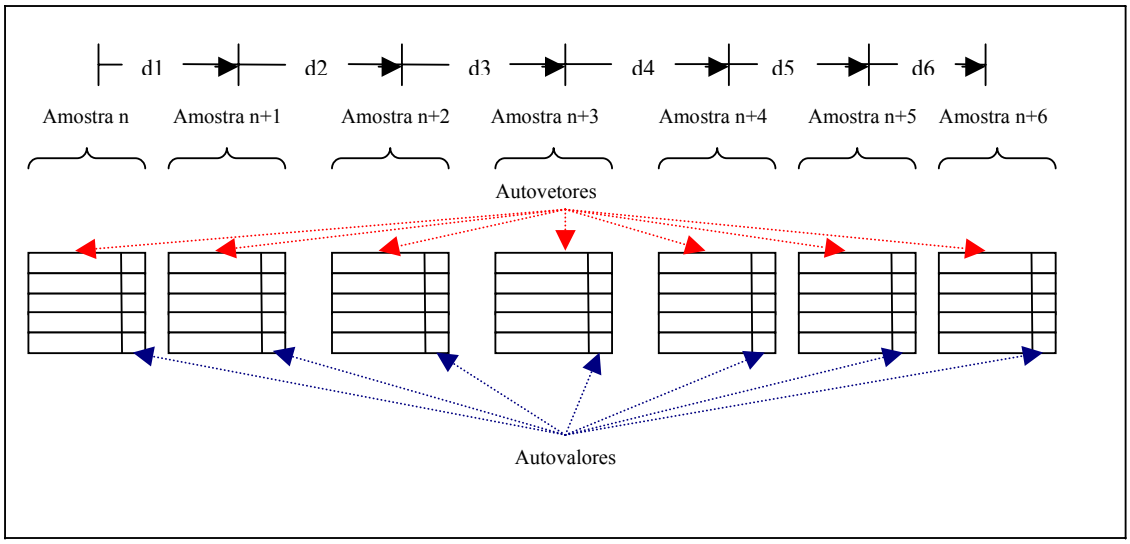

Figura 4 Arranjo de Vetores Característicos para cada nova amostra de dados

3.1.3.2.
Re-ordenamento dos vetores característicos de cada atualização da matriz de dados

Sabemos que cada autovetor representa uma componente do sinal, e é importante observar como estes autovetores vão mudando a cada nova atualização da matriz de dados. Portanto, ao aplicar o novo SVD temos, a cada duas amostras novas, novos autovetores e autovalores representativos da nova matriz, ou seja, temos uma nova atualização da matriz de dados. Cada um dos autovetores, obtidos na atualização "n" da matriz, tem seu correspondente autovetor semelhante (pela continuidade) na atualização "n+1" da matriz de dados, e assim por diante nas atualizações seguintes. É importante identificar estes autovetores semelhantes, pois a análise das respectivas trajetórias determinarão as partições, onde cada uma delas representa o comportamento de uma componente do sinal ao longo do tempo.

O procedimento a seguir consiste na procura dos autovetores mais "parecidos" em cada uma das atualizações da matriz de dados em relação aos obtidos na primeira atualização da matriz (descrito pela Equação 22).

Supondo:

i⇒ índice da amostra do sinal

M ⇒ número de vetores característicos

p,z ⇒ índices que fazem referência à atualização da matriz de dados.

q,g,k,j ⇒ índices que fazem referencia à posição do autovetor.

2(i-2)/M ⇒ número de atualizações da matriz de dados

V_{z_k} ⇒ Vetor característico "k"(autovetor "k" e autovalor "k") da matriz de atualização "z"

$d_{1,p_{q,g}}$ ⇒ Distância entre o autovetor de posição "q" da primeira atualização da matriz de dados e o autovetor de posição "g" da atualização "p" da matriz de dados

Calculam-se as distâncias $d_{1,p_{q,g}}$ (Equação 22):

$$d_{1,\,p\,q,g} = \left\| \text{autovetor}_{\;1\,q} - \text{autovetor}_{\;p\,g} \right\|_{\substack{p=2,..,2(i-2)/M \\ g=1,..,\;M \\ q=1,..,\;M}} \tag{22}$$

Para cada valor de $z = 2,..,2(i-2)/M$, temos que a nova ordem dos vetores característicos para cada atualização, z, da matriz de dados é dada por:

$$V_{z_k}\Big|_{k=1,..,M} = V_{z_j} = [\text{autovetor}_{z_j} \,|\, \text{autovalor}_{z_j}] \tag{23}$$

onde:

$$d_{1,z_{k,j}} = \min\left(d_{1,z_{k,g}}\right)_{g=1,..,M}$$

Na Figura 5 pode-se observar os autovetores mais "parecidos" em cada atualização da matriz de dados como os de uma mesma cor. De uma atualização da matriz de dados para a seguinte pode-se notar que estes autovetores parecidos não necessariamente vão conservar a mesma posição dentro da matriz.

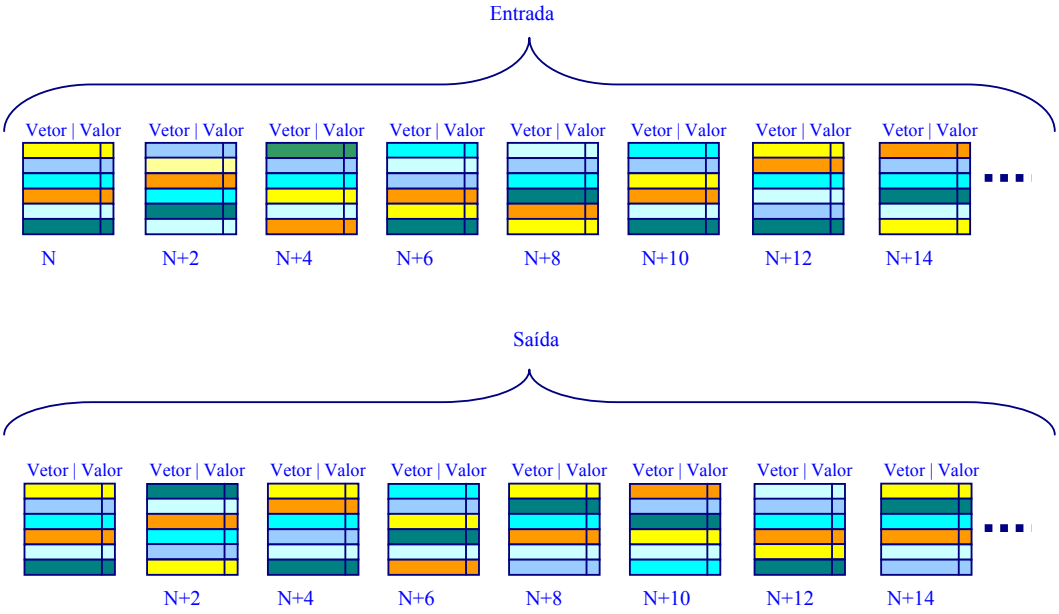

Figura 5 Autovetores e autovalores para cada atualização da matriz de dados em duas amostras novas

Como pode ser observado na Equação (23), a ordem dos vetores característicos da primeira atualização da matriz de dados ($V_{1_{z=1,..,M}}$) é tomada como referência para o re-ordenação dos vetores das atualizações seguintes ($V_{k_z}\Big|_{z=1,..,M}^{k=2,..,2(i-2)/M}$).

Após identificar os autovetores semelhantes em todas as atualizações da matriz de dados vamos arrumá-los da forma indicada na Figura 6. Observe-se que este procedimento é aplicado aos autovetores e autovalores resultantes da aplicação dos vários SVD, tanto do sinal de entrada quanto do sinal de saída. Para entender melhor esta idéia foram mostrados na Figura 5 os diferentes autovetores das

atualizações da matriz de dados através de diferentes cores, de tal forma que pode-se observar que de uma atualização para outra a ordem dos autovetores não se mantém.

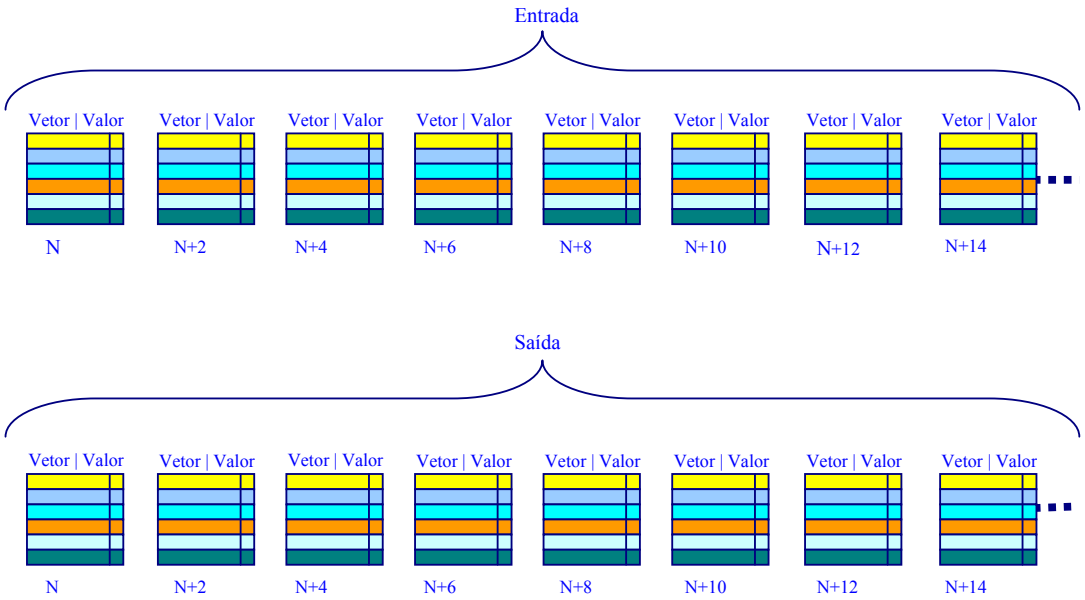

Figura 6 Ordenação dos autovetores e autovalores de cada atualização da matriz de dados segundo os autovetores mais parecidos.

No estudo, a ser detalhado mais à frente, é importante arrumar estes autovetores de acordo com sua semelhança, permitindo estudar o comportamento da trajetória que eles descrevem. Desta análise pode-se extrair a informação para realizar as "partições" nos espaços de sinais de entrada e saída.

A Figura 6 apresenta os autovetores, antes vistos desordenados na Figura 5, com autovetores semelhantes (de igual cor) já ordenados seguindo a ordem apresentada pelos autovetores resultantes da decomposição anterior da matriz de dados. O passo seguinte será realizar as partições dos sinais.

3.1.3.3.
Trajetória descrita pelos vetores característicos

A Figura 7 apresenta três curvas que representam 6 atualizações das trajetórias de três vetores característicos da matriz de dados.

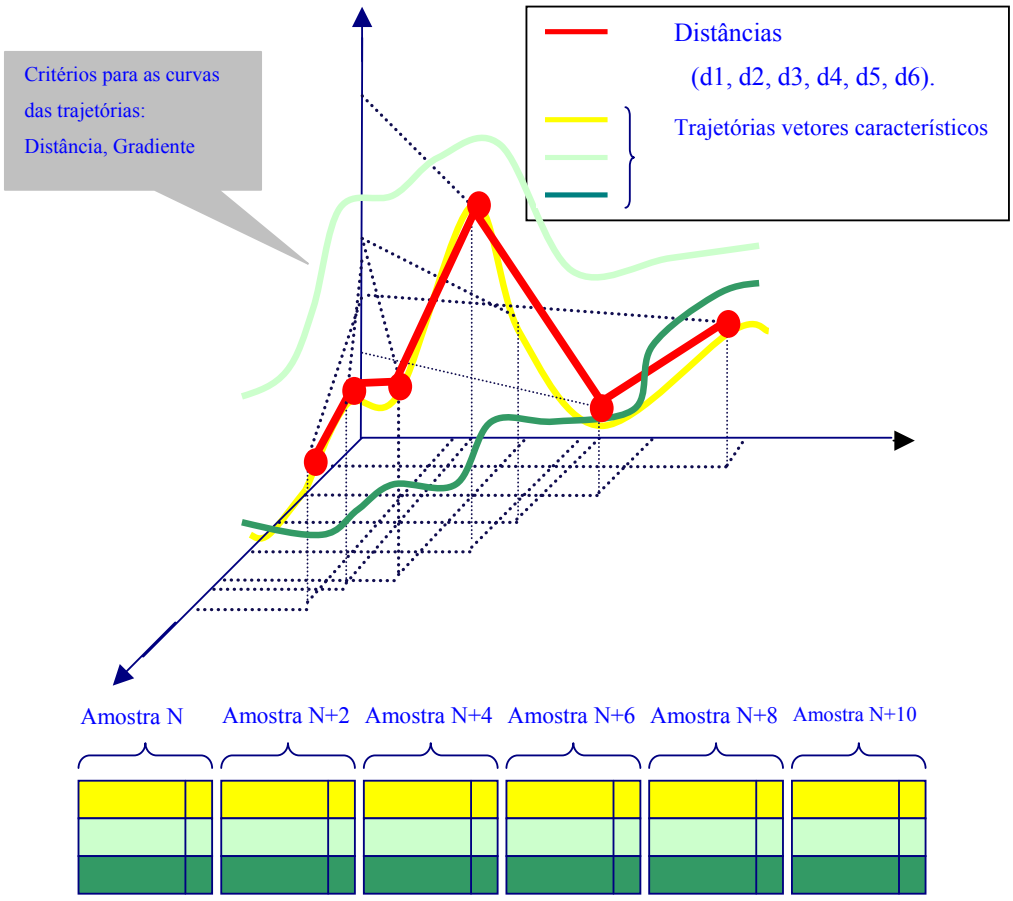

Figura 7 Trajetória das distâncias dos autovetores para 6 amostras novas considerando só três dimensões

Lembremo-nos que estes autovetores e seus correspondentes autovalores foram previamente ordenados segundo a semelhança na sucessão de atualizações; assim a curva de cor amarela representa a trajetória dos vetores característicos

(autovetores e autovalores) de cor amarela e assim por diante. Uma trajetória de vetores característicos (por exemplo, a de cor amarelo) tem entre um vetor característico e o consecutivo uma determinada distância (linhas vermelhas) e um vetor gradiente correspondente.

A distância utilizada busca a cada iteração, através da partição das trajetórias, qualificar a suavidade ou não de cada uma por meio de seu vetor gradiente. A distância entre o vetor característico e o seu sucessor na trajetória tem cada componente pesada pela correspondente magnitude da componente do gradiente definido pelos dois vetores. Assim, ao longo de uma trajetória as distâncias entre autovetores característicos sucessivos variam permitindo maior ou menor detalhamento na partição correspondente. Este procedimento é feito para cada uma das trajetórias dos vetores característicos. É importante lembrar que o software desenvolvido permite ao usuário estabelecer a distância "default" para cada uma das trajetórias. Em outras palavras, significa que as partições feitas na trajetória descrita por um vetor característico não vão ser as mesmas que as partições feitas nas outras trajetórias, permitindo ao usuário fazer uma análise particular em cada uma das trajetórias. Entretanto, mais adiante, através de uma alteração de procedimento, todas as partições feitas nas trajetórias dos vetores característicos dos espaços dos sinais de entrada e saída vão-se superpor em uma partição global, satisfazendo a cada uma delas individualmente.

3.1.3.4. Realização das partições

O critério para alcançar uma partição deve levar em consideração o tipo de superfície descrita pelos vetores característicos. Por exemplo, se o novo vetor característico está próximo ao anterior, caracterizando uma superfície suave, isto indica que muitos elementos ao redor podem ser representados por um único

elemento. Caso haja uma diferença acentuada entre vetores sucessivos, isto indica que apenas os elementos muito próximos de um ou do outro podem ser considerados equivalentes. Ou seja, uma possível escolha para distância deve levar em consideração uma estimativa do gradiente no ponto.

Dentro deste critério a partição apresentará regiões bastante irregulares, não uniformes, ao longo do espaço, sendo a região menor, ou seja, com "partição" mais detalhada, onde a superfície apresenta maior irregularidade ou saltos e menos detalhada onde ela é regular, ou com trajetória suave.

Temos então como ponto fundamental da partição a escolha da distância a ser adotada, estabelecendo-se o critério de partição. Para o caso aqui desenvolvido a distância entre vetores característicos sucessivos em uma trajetória é dada pelo produto interno entre o correspondente vetor gradiente (magnitudes) e o vetor das distâncias euclidianas (sem perda de generalidade). Note-se que este critério pode evoluir permitindo até mesmo privilegiar direções.

O programa, desenvolvido como suporte ao trabalho, assume como "default" um valor máximo de distância para cada uma das trajetórias correspondentes às componentes do sinal (mais adiante será explicado como se define este valor). Entretanto, ele permite ao usuário definir outros valores "default" segundo seus próprios critérios.

Como foi dito anteriormente, uma das vantagens do SVD é permitir a identificação das componentes de ruído dentro do sinal. Imagine-se ter a trajetória descrita pela componente de menor autovalor (o que poderia representar ruído, principalmente em situações de boa relação sinal-ruído). Devido à suposta forte variabilidade do ruído, ele deve oferecer uma componente de distância apreciável, influenciando fortemente o processo de partição. É, portanto preferível dar um valor "default" alto o suficiente para evitar partições devido a esta trajetória, pois qualquer

partição feita nesta componente vai-se superpor sobre as demais, já que as partições são feitas separadamente em cada vetor característico componente.

Na seqüência deste trabalho será apresentado um exemplo ilustrativo no procedimento de partições:

A Figura 8 mostra as coordenadas de um mesmo vetor característico ao longo de uma trajetória.

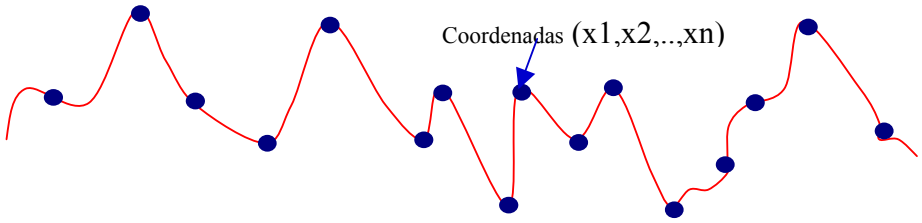

Figura 8 Trajetória descrita por um vetor característico no espaço de “n”dimensões

Entre uma coordenada e a seguinte existe uma distância euclidiana, e um vetor gradiente. Multiplicando estas distâncias euclidianas pelos vetores gradientes respectivos temos uma nova trajetória no espaço de “n”dimensões, para facilitar a compreensão do algoritmo de partições, vamos representar esta trajetória em uma dimensão, como na Figura 9, mas lembre-se, que nos casos reais, esta trajetória seria feita em espaços maiores a três dimensões.

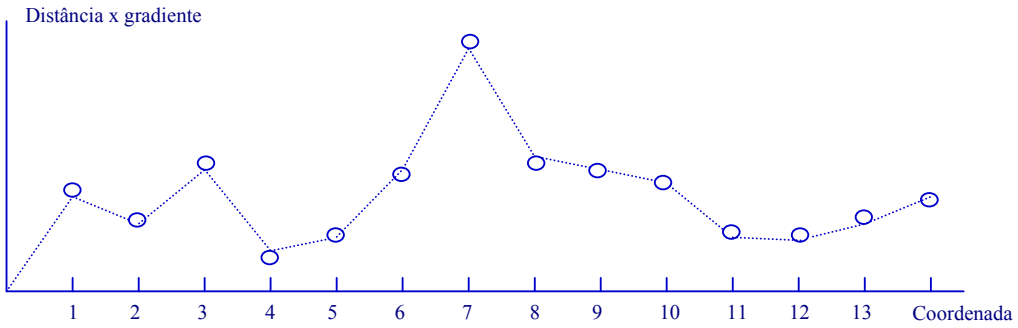

Figura 9 Gráfico “Gradientes por distâncias”

Do gráfico da Figura 10, podemos calcular as distâncias entre um ponto da trajetória e seu consecutivo, tal como mostra a Figura 10.

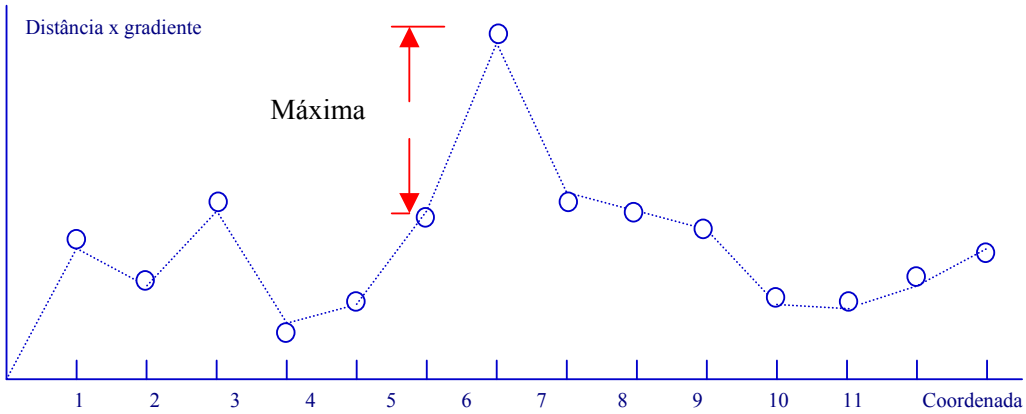

Figura 10 Curva "F" (Distâncias)

Da Figura 10 pode-se determinar qual seria a máxima distância, de forma tal que esta pode ser usada para determinar o valor "default" da distância (máxima distância/2). Isto é feito a seguir.

Sendo:

F $\Rightarrow$ curva representativa das distâncias descritas pelo produto interno da norma da diferença e pela magnitude do gradiente dos respectivos vetores característicos

d $\Rightarrow$ distância assumida para realizar as partições (valor por "default" igual à metade da distância máxima a qual será determinada mais adiante)

Logo temos:

$$\text{Dado}_{k\,j} = \text{norma}\left\| \vec{V}_{k\,j} - \vec{V}_{k+1\,j} \right\| * \text{gradiente}\left(\vec{V}_{k\,j} - \vec{V}_{k+1\,j} \right) \Big|_{\substack{k=1,..,2(i-2)/M \text{ on} \\ j=1,..,M}}$$

de sabe-se que:

M $\Rightarrow$ número de vetores característicos

k $\Rightarrow$ índice que indica a atualização da matriz de dados

$j \Rightarrow$ índice que indica a posição do vetor característico

$\vec{V}_{k\,j} \Rightarrow$ vetor característico da posição "j" da atualização "k" da matriz de dados

$\vec{V}_{k+1\,j} \Rightarrow$ vetor característico da posição "j" da atualização "k+1" da matriz de dados

$$F_{k\,j} = \left\| Dado_{k\,j} - Dado_{k+1\,j} \right\|_{k=1,..,2(i-2)/M-1} \Rightarrow$$ Trajetória do produto da distância pelo gradiente do vetor característico de posição "j" formado pelas amostras de posição "k".

Caso $\left\| F_{k\,j} - F_{k+1\,j} \right\| \geq d$ e $\left\| F_{z\,j} - F_{z+1\,j} \right\| \geq d$, onde $z>k$

então é feita uma partição constituída desde a amostra "k+1", até a amostra "z".

A Equação pode ser facilmente entendida pela representação na Figura 11.

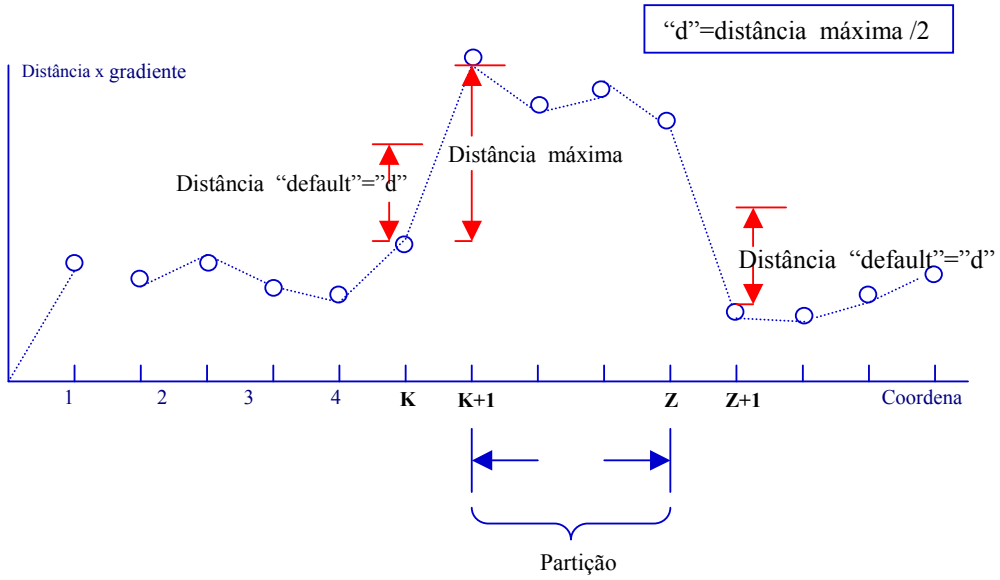

Figura 11 Curva "F" correspondente ao vetor característico da posição "j"

Por exemplo, para o caso de um sinal senoidal no qual se aplicou a decomposição a três autovetores e autovalores (matriz de dados 3x3), ve-se nas Figuras 13, 14 e 15, a cada atualização da matriz de dados, os gráficos das trajetórias dos vetores gradientes pelas distâncias entre os vetores característicos correspondentes.

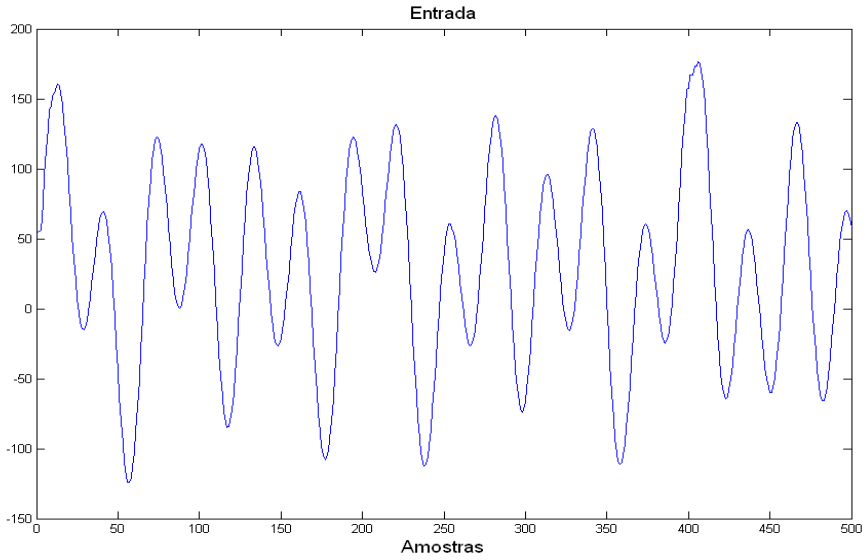

Figura 12 Sinal de Entrada

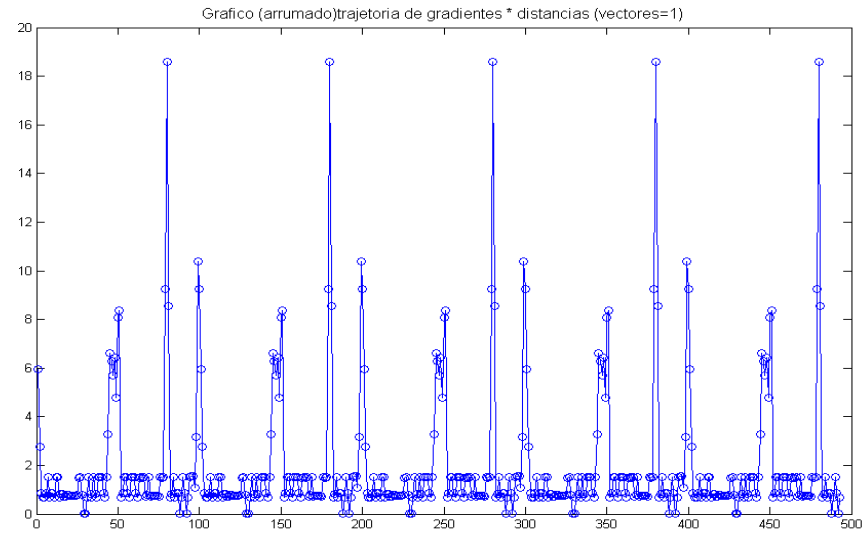

Figura 13 Gráfico “Gradientes x distâncias” do vetor característico 1 do sinal de entrada

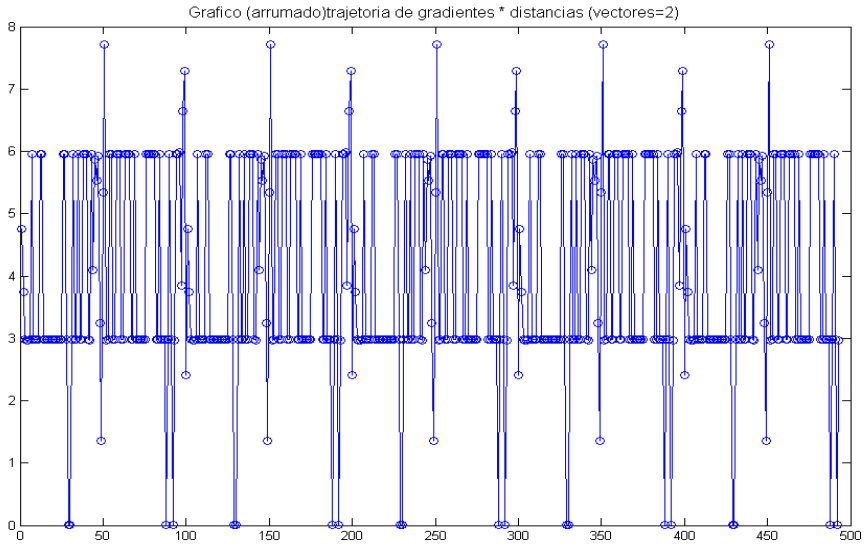

Figura 14 Gráfico “Gradientes x distâncias” do vetor característico 2 do sinal de entrada

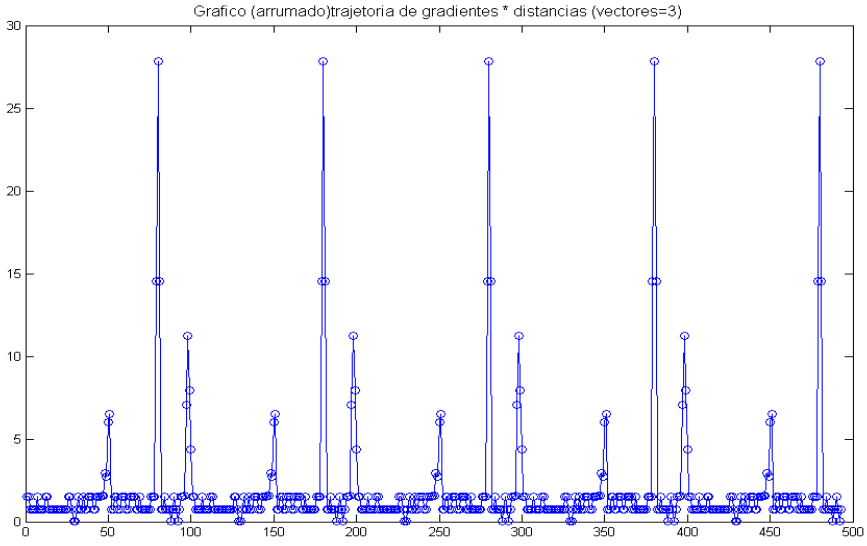

Figura 15 Gráfico “Gradientes x distâncias” do vetor característico 3 do sinal de entrada

Lembremos que as trajetórias são feitas no espaço de “n” dimensões, mas os gráficos mostrados nas Figuras 13, 14 2 15, representam apenas a distância entre cada coordenada das trajetórias.

O mesmo é feito para o sinal de saída mostrada na Figura 16.

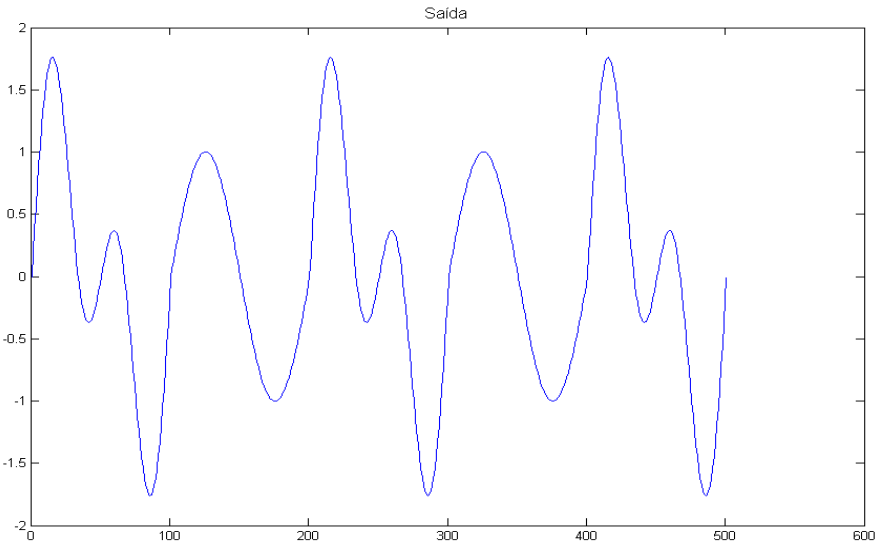

Figura 16 Sinal de Saída

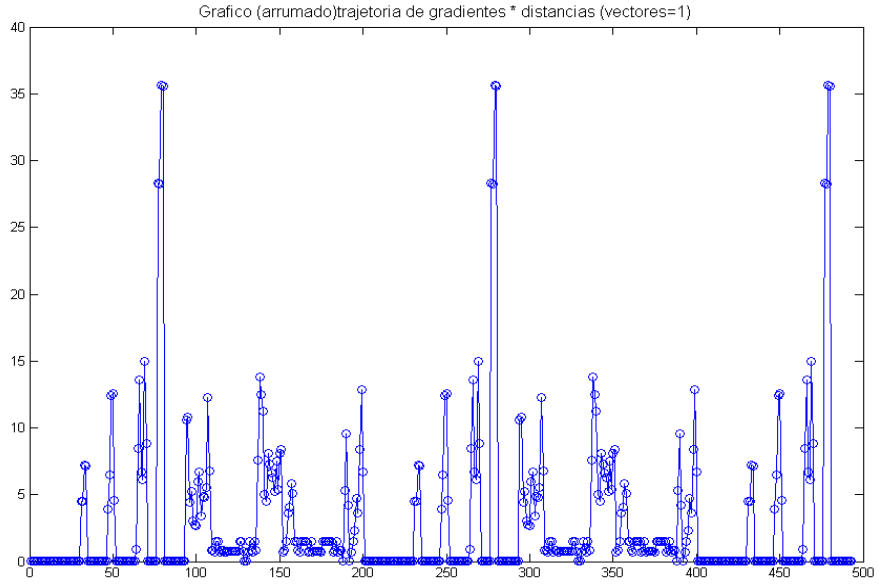

Figura 17 Gráfico “Gradientes x distâncias” do vetor característico 1 do sinal de saída

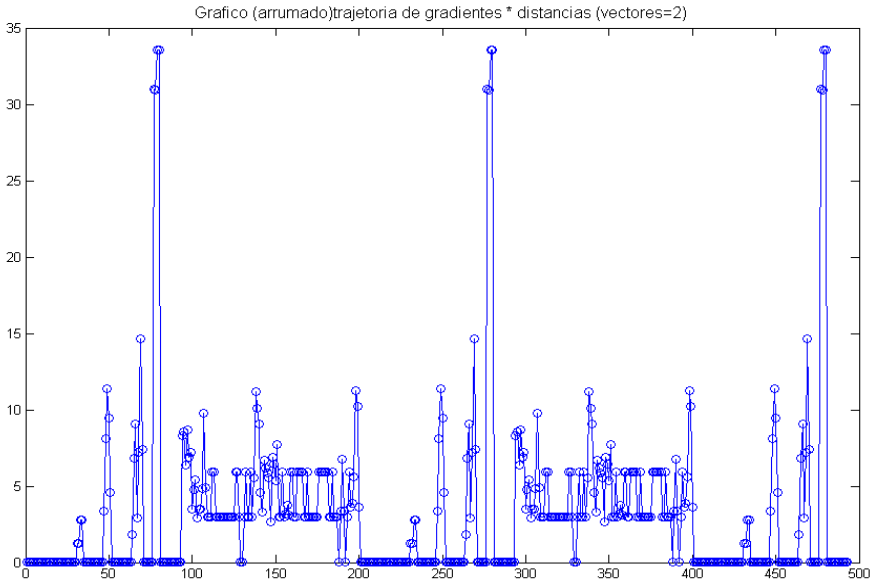

Figura 18 Gráfico “Gradientes x distâncias” do vetor característico 2 do sinal de saída

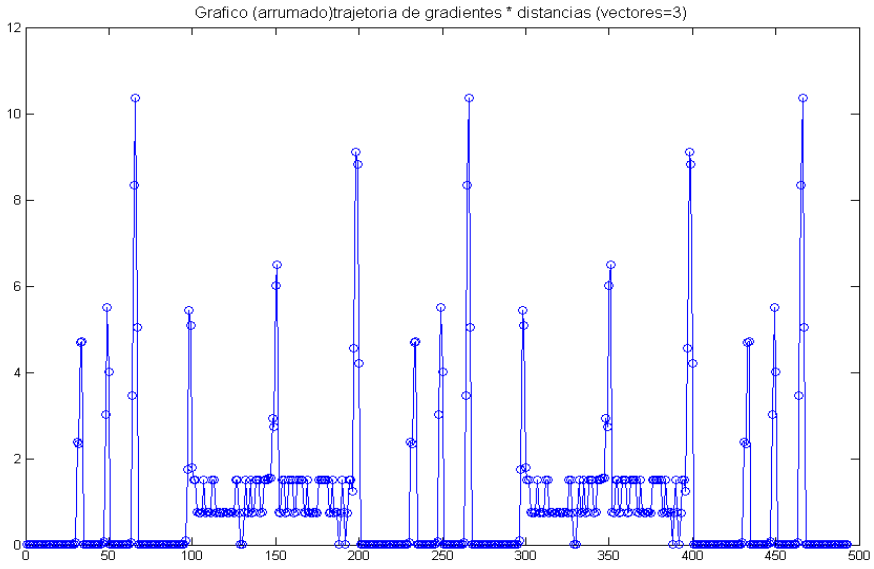

Figura 19 Gráfico “Gradientes x distâncias” do vetor característico 3 do sinal de saída

No software desenvolvido, se assumiu este valor como “default” para os limites de distância, servindo como referência para realizar as partições. Podem-se observar os pontos marcados na Figura 20, representando o passo de cada atualização

à seguinte. Entre os pontos percebe-se uma distância máxima, que servirá como parâmetro para realizar a partição. A metade deste valor é tomada como valor "default", significando que uma partição será executada sempre que a distância supere-o este valor.

O gráfico da Figura 20 mostra as distâncias, incluindo os gradientes.

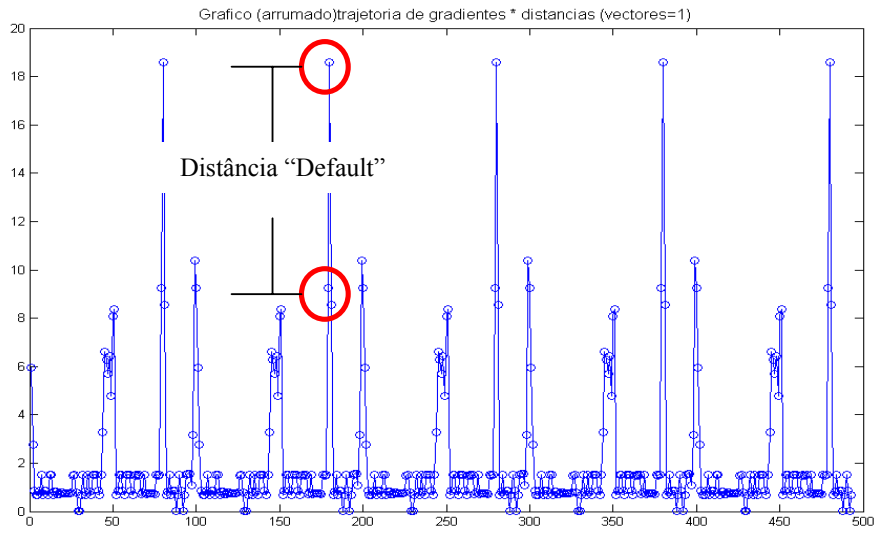

Figura 20 Distância "Default" assumida para realizar as partições.

Aplicando este conceito de distância, para realizar as partições nos sinais de entrada e saída representados pelos gráficos nas Figuras 12 e 16, temos as respectivas partições como se observa na Figura 21. Novamente, chamamos a atenção aqui para o fato que estas partições delimitam fronteiras de comportamento irregular dos sinais.

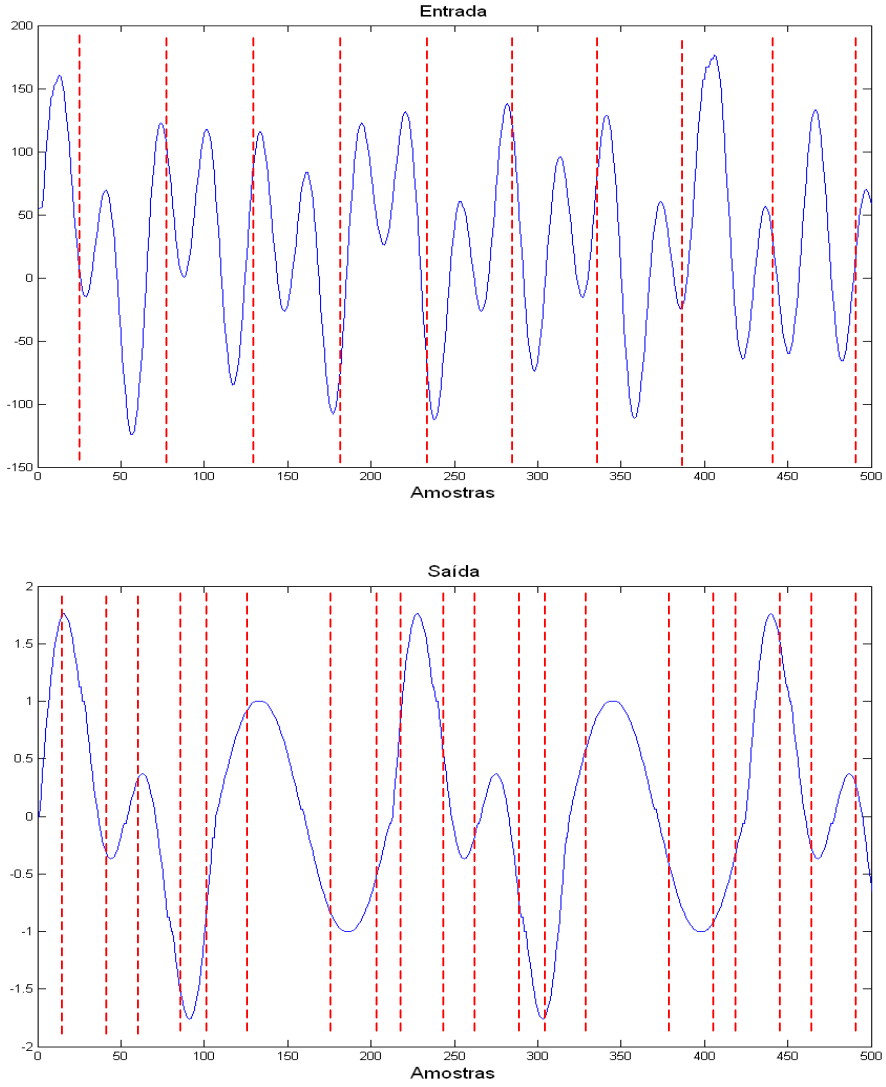

Figura 21 Partições do sinal de Entrada e Saída.

3.1.3.5. Mapeamento das partições dos sinais de entrada e saída

Uma vez realizada a partição no espaço do sinal de entrada, assim como no de saída, temos que relacionar estas partições, já que não necessariamente as partições do sinal de entrada vão coincidir com as partições do sinal de saída (O natural é que

não o façam). Cria-se então uma espécie de mapeamento, induzindo a partição desejada, como mostra simbolicamente a Figura 22.

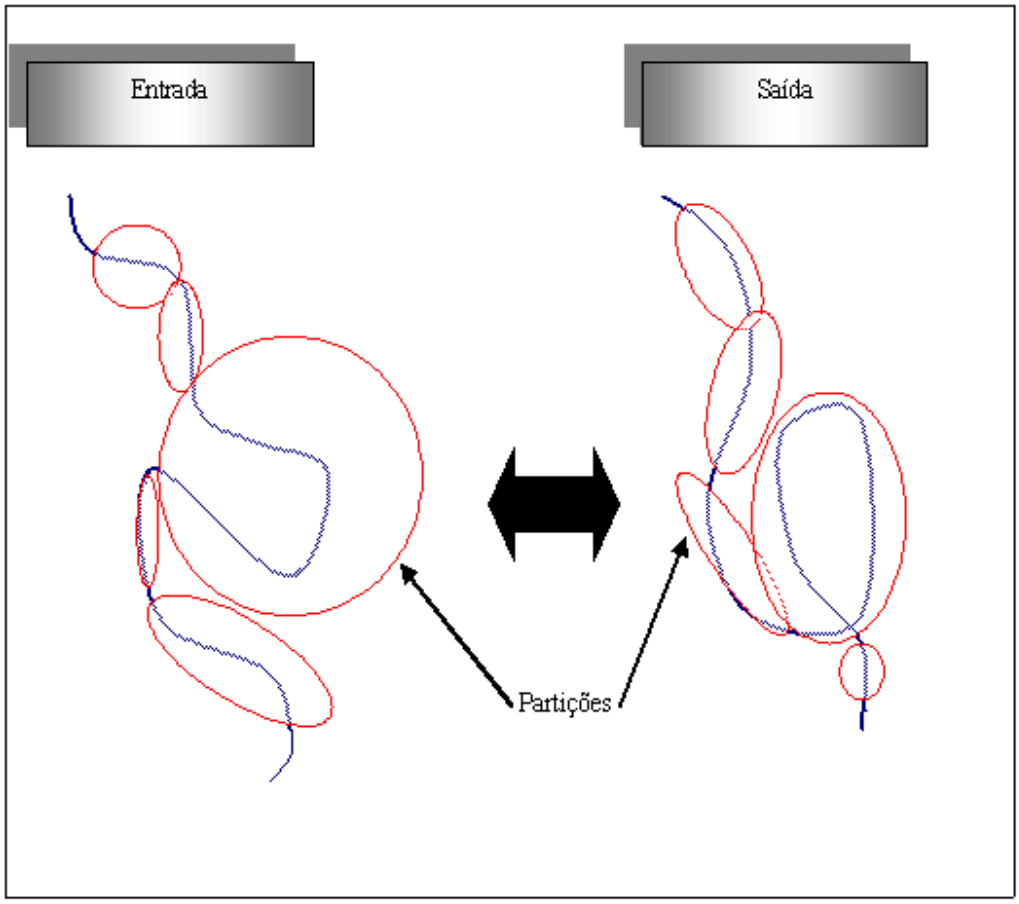

Figura 22 Mapeamento das partições de entrada e saída

O procedimento para estabelecer esta relação de partições consiste em "superpor" as partições do sinal de entrada sobre as partições do sinal de saída e vice-versa, de tal forma que ao final temos partições correspondentes no sinal de entrada e no sinal de saída. A Figura 23 mostra melhor este conceito.

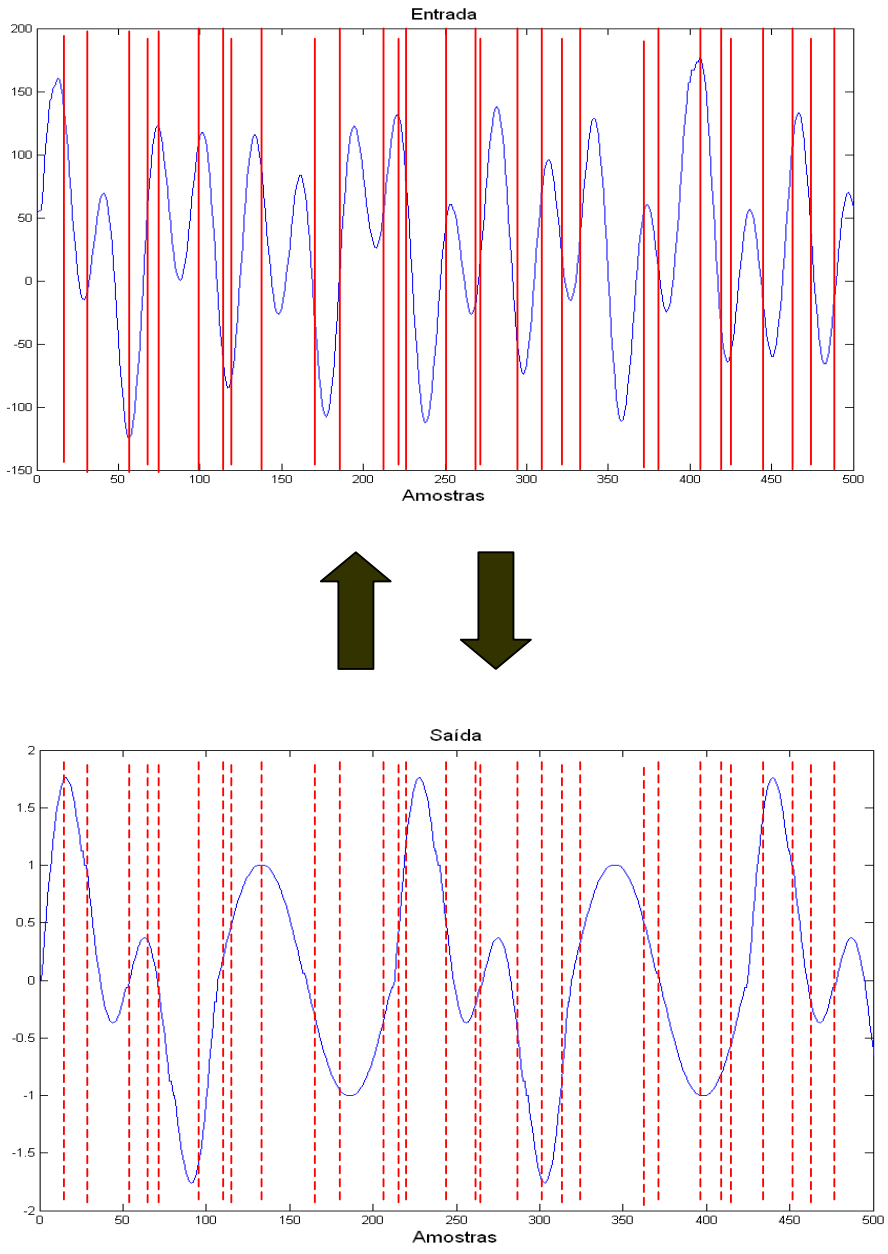

Figura 23 Superposição de partições do sinal de Entrada e Saída

Notemos que, segundo o comportamento dos sinais, pode acontecer que uma partição do sinal de entrada esteja relacionada a duas ou mais partições (comportamentos diferentes) no sinal de saída ou vice-versa. Comportamentos deste tipo são factíveis de acontecer em sistemas não lineares variantes no tempo. Por exemplo, suponha o caso em que temos as partições de entrada “1” e “2”

consecutivas, sendo que a partição "2" está mapeada em duas partições, "3" e "4", na saída. O algoritmo proposto permitirá identificar dentro destas partições da saída a correspondente a ser usada na síntese de uma nova amostra. Para isto poderá ser usado o comportamento da partição, que precede a entrada 2, entrada "1", para ajudar nesta identificação.

Uma vez feita a superposição das partições, cada partição resultante será submetida às técnicas selecionadas (Rede RBf, Neuro-fuzzy). Os parâmetros dos modelos resultantes serão armazenados em disco para seu uso na etapa da síntese.

3.1.4. Armazenamento e identificação dos vetores característicos representativos de cada partição

Realizada a superposição de partições dos sinais de entrada e de saída, temos que armazenar esta informação de forma que sirva como uma espécie de mapeamento para o processo de síntese. Cada amostra a ser estimada do sinal de saída tem uma respectiva amostra de entrada à qual corresponderá uma posição dentro de uma partição. Identificando tal partição (e, portanto a correspondente de saída) temos identificado os modelos associados que permitem estimar a respectiva amostra de saída.

Este mapeamento é feito da seguinte forma:

- Lembremos que tanto o sinal de entrada como o sinal de saída tem partições superpostas (Figura 24). Pode-se preencher a matriz de dados de ordem MxM com as 2xM-1 amostras da primeira partição do sinal. Aplica-se o SVD a ela e obtêm-se os correspondentes autovetores e autovalores associados. Arrumam-

se estes autovetores e autovalores em vetores característicos, como mostra a Figura 24. Observe-se que neste último vetor está toda a informação do comportamento das 2xM-1 primeiras amostras do sinal.

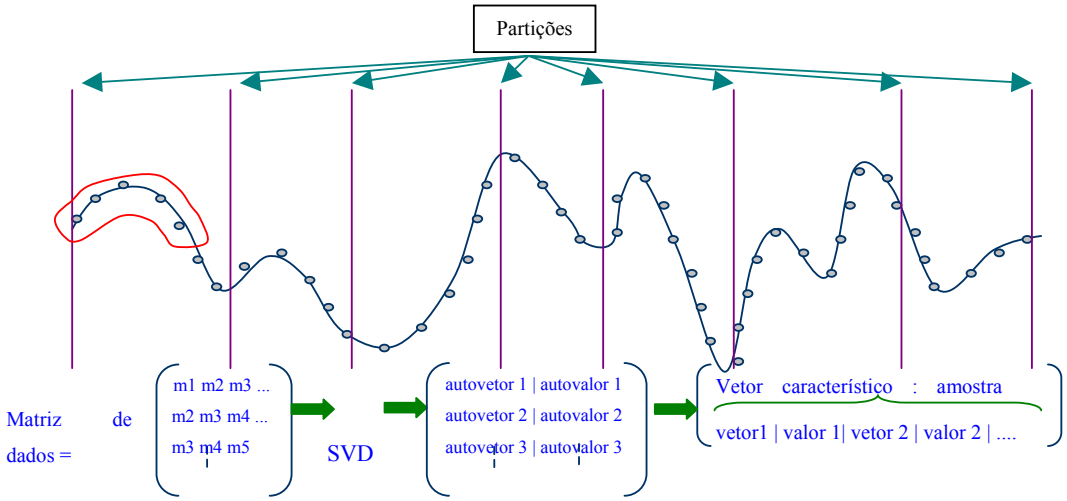

Figura 24 Vetor Característico amostra 2xM-1

- Preenchemos novamente a matriz de dados com as 2xM-1 amostras seguintes (deslocadas em uma amostra) e aplicamos o SVD. Arrumam-se os novos autovalores e autovetores, obtendo o novo vetor característico indicado na Figura 25.

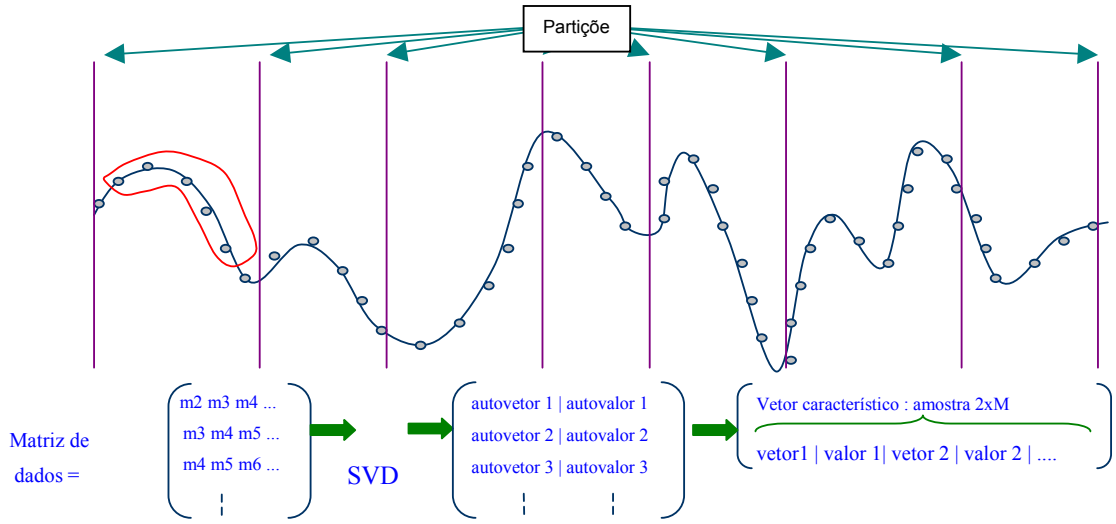

Figura 25 Vetor Característico, amostra 2xM

Este procedimento é realizado a cada novo deslocamento de uma amostra.

- O objetivo deste mapeamento é identificar a partição que melhor estimaria uma amostra na saída para sua respectiva amostra de entrada, mas devido ao número de amostras que possam ter nossos sinais de entrada e saída, selecioná-la pode ser demorado. Para reduzir o esforço computacional escolhe-se o vetor característico mais representativo da partição para representá-la. Basta comparar o vetor de entrada a este vetor para identificar a pertinência ou não do vetor de entrada com respeito à partição em questão. A Equação (24) apresenta a lógica para identificar este vetor representativo (Figura 26).

$d_{vetor\_característico\_i,\, vetor\_característico\_j}$: Distância entre os vetores característicos "i" e "j".

k: número de vetores característicos

$$\text{vetor representativo} = \min\left(\text{Soma}_{i=1,..,k}\right) \tag{24}$$

onde:

$$\text{Soma}_i = \sum_{\substack{j=1,..,k \\ j \neq i}} \text{norma}\left[d_{vetor\_característico\_i,\, vetor\_característico\_j}\right]$$

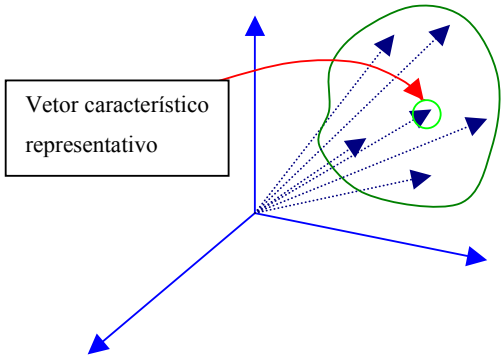

Figura 26 Vetor representativo da partição

- Identificando cada vetor representativo de cada partição temos a seguintes matrizes (Figura 27).

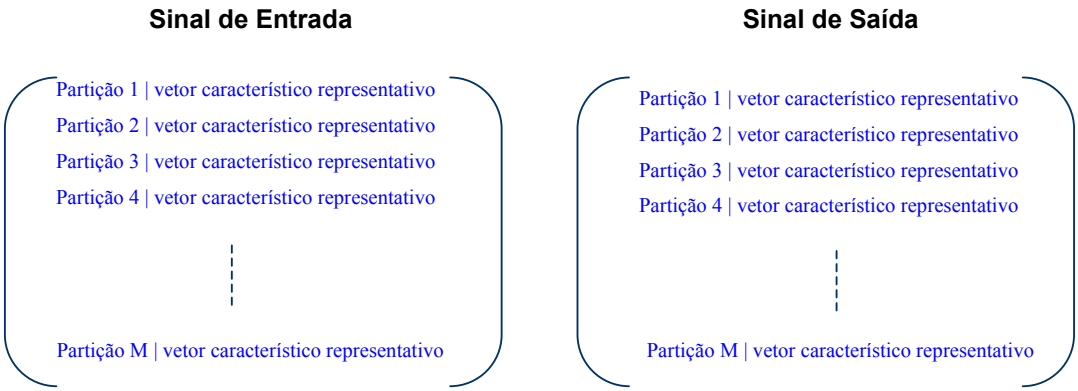

Figura 27 Matrizes Vetores representativos

- Uma vez feita a modelagem de cada partição verifica-se caso o erro de modelagem esperado tenha sido satisfeito, em caso contrário são feitas sub-partições até satisfazê-lo (função do Algoritmo de Cooperação que será explicado mais adiante). Pode-se, portanto chegar ao caso em que cada partição seja constituída por uma só amostra. Neste caso o vetor representativo da partição será o próprio vetor característico respectivo a essa amostra. Assim, temos as seguintes matrizes (Figura 28).

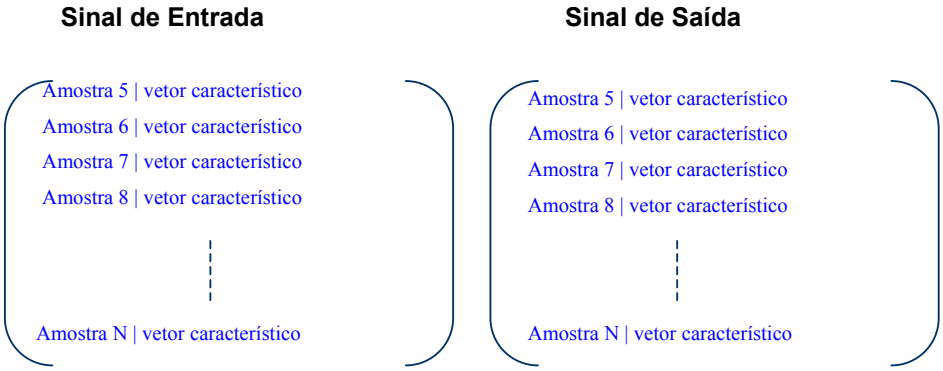

Figura 28 Matrizes de vetores representativos (Partições de um só elemento)

- Finalmente as matrizes (Figura 27, 28) podem estar misturadas, ou seja: pode acontecer que existam partições de um só elemento enquanto outras partições apresentam maior número de elementos.

A técnica de partição dos espaços de entrada e saída é uma das propostas apresentadas neste trabalho. Uma segunda contribuição é descrita na próxima seção.

3.1.5.
Algoritmo de Cooperação entre técnicas de redes

O Algoritmo de Cooperação tem como função combinar a informação provemiente as técnicas utilizadas de forma a minimizar o erro da saída. A seguir explicaremos passo a passo a sua lógica de funcionamento.

3.1.5.1.
Modelagem das partições

Até o momento efetuamos as partições nos espaços dos sinais, sendo o passo seguinte treiná-las com as diferentes técnicas selecionadas (redes RBF, modelos Neuro-fuzzy, etc.). Cada partição dos espaços de sinal é submetida a todas as diferentes técnicas de modelagem. Cada técnica apresenta sua estimação da saída adequada. Esta saída apresenta um erro traduzindo a eficiência da técnica correspondente.

É claro que cada uma das técnicas empregadas possui particularidades, que dificultam a sua operação em conjunto com as demais, ou seja, a troca de informação entre elas. Sabemos [2] também que cada técnica apresenta resultados de modelagem dependentes das características do sinal em questão.

Vejamos como é feita a modelagem de uma partição. Os dados do sinal de entrada e seus respectivos dados de saída serão treinados por modelos RBF e Neuro-fuzzy. Cada uma das técnicas devolverá uma saída estimada para as entradas em questão, estas saídas estimadas e a saída real serão expressas em função de seus autovetores e autovalores respectivos. Estes autovetores e autovalores podem ser obtidos usando os comandos SVD e EIG do Matlab, mas devido ao fato de trabalhar aqui com matrizes de dados quadradas preferiu-se utilizar o comando EIG do Matlab,

poupando assim tempo computacional implícito ao uso do comando SVD. Este é guardado para casos gerais, onde a matriz de dados não necessariamente é quadrada. O EIG é definido como:

$$[U, S] = EIG(X) \tag{25}$$

onde

U: autovetores

S: autovalores

X: matriz quadrada

A Figura 29 mostra o processo de obtenção dos autovetores e autovalores das saídas estimadas por cada uma das técnicas.

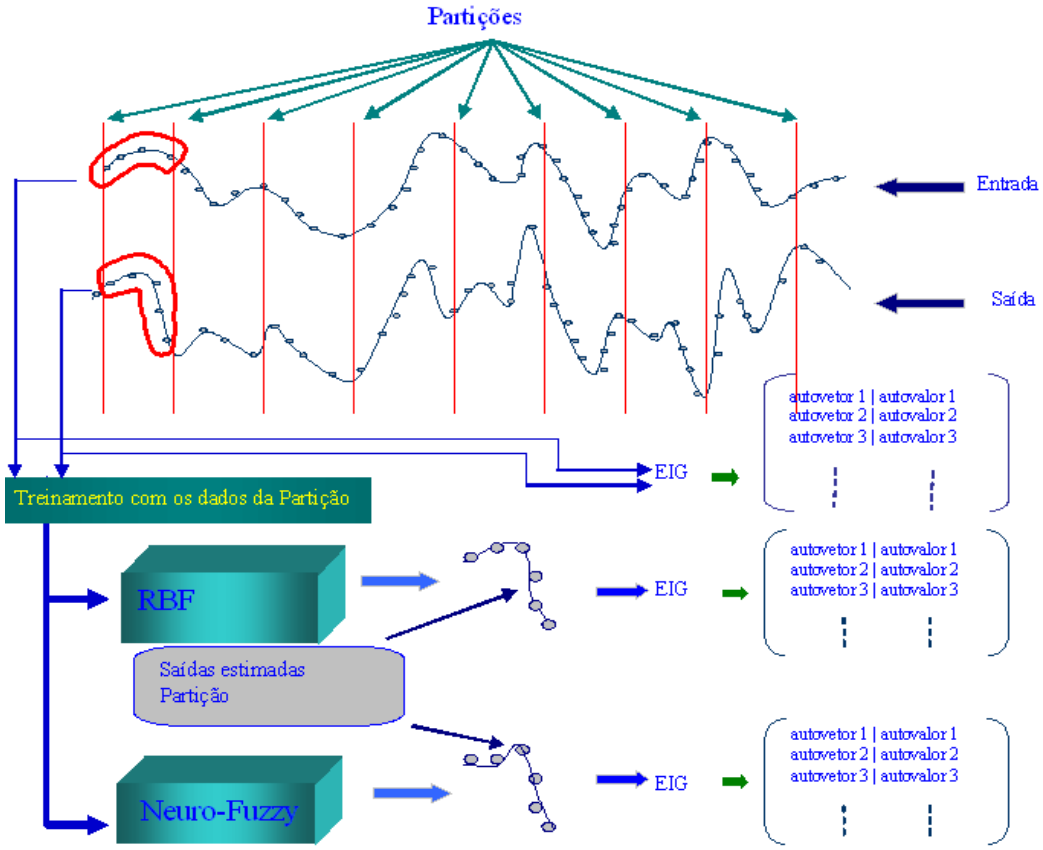

Figura 29 Comparação de vetores característicos Reais e Estimados.

3.1.5.2.
Atribuição de pesos

Sabendo que as técnicas possuem diferentes estruturas e assim fornecem estimativas diversas para as correspondentes saídas, estendemos a análise SVD a estas saídas, fazendo a decomposição das saídas estimadas em autovetores e autovalores.

Será efetuado um procedimento de cálculo para os pesos, tanto para os autovetores das saídas estimadas como para os autovalores, de forma que a combinação linear dos autovetores das saídas estimadas esteja o mais próximo dos correspondentes autovetores da saída real. Do mesmo modo, procurar-se-á que a combinação linear dos autovalores respectivos das saídas estimadas estejam o mais próximos dos autovalores correspondentes da saída real. A seguir será explicado o procedimento de cálculo dos pesos dos autovetores. Mas adiante será explicado o procedimento de cálculo dos pesos dos autovalores respectivos.
Para cada um dos autovetores das saídas estimadas, são estimados "pesos" de forma a satisfazer a equação (26), aqui restrita a duas entradas, ou seja, duas redes. O método é facilmente extensível a um número maior de redes.

pesos		**autovetores RBF**		**pesos**		**autovetores Neuro - fuzzy**		**autovetores Saída real**
p1	*	**autovetor1**	+	p2	*	**autovetor1**	=	**autovetor1**
p3	*	**autovetor2**	+	p4	*	**autovetor2**	=	**autovetor2**
⋮		⋮	⋮	⋮		⋮	=	⋮
p_n	*	**autovetor_n**	+	p_n + 1	*	**autovetor_n**	=	**autovetor_n**

Generalizando:

$$\begin{bmatrix} p1_j & p2_j & \cdots & pk_j \end{bmatrix} \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix} = \begin{bmatrix} \mathbf{u}^r_j \end{bmatrix} \tag{26}$$

onde:

j: índice do autovetor

k: número de técnicas

p1,p2,..,pk: pesos

u: autovetores das técnicas (técnicas 1,2,..,k)

$\mathbf{u}^r_j$: autovetor verdadeiro

Como o número de técnicas utilizadas e a dimensão da matriz de dados dificilmente coincidem, os pesos (p1j, p2j,.., pkj, etc.) podem ser calculados através da inversa generalizada, sendo então aceito que as igualdades em questão serão interpretadas como aproximações

$$\begin{bmatrix} p1_j & p2_j & \cdots & pk_j \end{bmatrix} [\mathbf{I}] = \begin{bmatrix} \mathbf{u}^r_j \end{bmatrix} * \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix}^t * \left[\begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix} * \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix}^t \right]^{-1} \tag{27}$$

Se dois ou mais autovetores das técnicas(u1j,u2j,..,ukj) estiverem muito próximos, temos que a matriz:

$$\begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix} * \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix}^t$$

é mal condicionada e eventualmente singular. Portanto, não existe neste caso a matriz inversa. Isto pode ser resolvido considerando somente um autovetor representativo dentre todos aqueles que estiverem muito próximos, evitando assim a formação de uma matriz singular e sem prejudicar a qualidade da aproximação.

Da equação (27) pode-se observar também o uso da inversa generalizada, a qual por trabalhar com a solução de norma mínima, influenciará mais adiante no erro de modelagem.

Multiplicando os pesos achados na equação (27) pelos autovetores das técnicas correspondentes, temos novos autovetores, resultados das combinações, como a seguir:

$$\begin{bmatrix} p1_j & p2_j & \cdots & pk_j \end{bmatrix} \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix} = \begin{bmatrix} \mathbf{u}^{e}_{j\ \text{combinação}} \end{bmatrix} \tag{28}$$

$$\begin{bmatrix} \mathbf{u}^{e}_{j\ \text{combinação}} \end{bmatrix} - \begin{bmatrix} \mathbf{u}^{r}_{j} \end{bmatrix} = \begin{bmatrix} \text{erro}_j \end{bmatrix}$$

Mas os autovetores obtidos ($\mathbf{u}^{e}_{j\ \text{combinação}}$) não são necessariamente ortonormais.

Devido a isto é realizado o processo de ortonormalização dos autovetores de Gram Schimdt [59], descrito como segue.

É escolhido para servir como vetor de referência aquele autovetor ($\mathbf{u}^{e}_{j\ combinação}$), resultado da combinação, que estiver mais próximo do seu correspondente autovetor real ($\mathbf{u}^{r}_{j}$).

Logo:

$$erro_j = norma\_euclidiana\left\| \mathbf{u}^{e}_{j\ combinação} - \mathbf{u}^{r}_{j} \right\| \quad (29)$$

$j = 1,..., n$

n : número de autovetores

$$\mathbf{autovetor}_{escolhido\,1} = \frac{\mathbf{u}^{e}_{i\ combinação}}{norma\_euclidiana(\mathbf{u}^{e}_{i\ combinação})}, \quad (30)$$

$erro_i$ = erro mínimo

Temos identificado então o primeiro autovetor (de posição "i") e os pesos respectivos de forma tal que:

$$\mathbf{autovetor}_{escolhido\,1} = \frac{\mathbf{u}^{e}_{i\ combinação}}{norma\_euclidiana(\mathbf{u}^{e}_{i\ combinação})}$$

caso de duas técnicas

$$\mathbf{autovetor}_{escolhido\,1} = p1_i * \mathbf{u}_{i\,RBF} + p2_i * \mathbf{u}_{i\,Neuro-fuzzy},$$

A seguir são calculados novamente os pesos (p1, p2) dos autovetores restantes satisfazendo sempre a ortonormalidade:

$j = 2,..,n$(número de autovetores)
k = número de técnicas

$$\mathbf{autovetor}_{\text{escolhido}\,j} = p1_j * \mathbf{u}1_j + p2_j * \mathbf{u}2_j + ... + pk_j * \mathbf{u}k_j$$

$$\text{ortogonalidade} \begin{cases} \mathbf{autovetor}_{\text{escolhido}\,j} * (\mathbf{autovetor}_{\text{escolhido}\,1})^t = 0 \\ \mathbf{autovetor}_{\text{escolhido}\,j} * (\mathbf{autovetor}_{\text{escolhido}\,2})^t = 0 \\ \vdots \\ \mathbf{autovetor}_{\text{escolhido}\,j} * (\mathbf{autovetor}_{\text{escolhido}\,(j-1)})^t = 0 \end{cases} \quad (31)$$

Observe-se que a normalidade ($\text{norma\_euclidiana} \left\| \text{autovetor}_{\text{escolhido}\,j} \right\| = 1$, j=1,..,n) é imediata e não uma condição.

Logo:

$$\begin{cases} j = 1,..,n \\ k = \text{número de técnicas} \end{cases} \begin{cases} \mathbf{autovetor}_{\text{escolhido}\,j} = p1_j * \mathbf{u}1_j + p2_j * \mathbf{u}2_j + ... + pk_j * \mathbf{u}k_j \\ \mathbf{autovetor}_{\text{escolhido}\,1} \perp \mathbf{autovetor}_{\text{escolhido}\,2} \perp ..... \perp \mathbf{autovetor}_{\text{escolhido}\,n} \\ \text{norma\_euclidiana} \left\| \mathbf{autovetor}_{\text{escolhido}\,j} \right\| = 1 \end{cases}$$

Desta forma tem-se estimados os pesos $(p1_j, p2_j)$ da combina (32) os autovetores, cumprindo as características de ortonormalização.

3.1.5.3.
Aplicação do processo inverso do EIG

Lembremos que os autovetores e autovalores das saídas estimadas pelas técnicas foram obtidos pelo comando EIG do Matlab descrito a seguir.

[autovetores,autovalores] = EIG(Matriz de dados) (33)

onde:
Matriz de dados * autovetores = autovetores * autovalores

Tendo os novos autovetores originados das combinações ($\text{autovetores}_{\text{escolhidos}}$) calculam-se os novos autovalores como a seguir (34):

$$\text{autovalores}_{\text{combinação}} = (\text{autovetores}_{\text{escolhidos}})^{-1} * \text{Matriz\_de\_dados} * \text{autovetores}_{\text{escolhidos}}$$

$$\text{autovalores}_{\text{combinação}} = \text{autovalores}_{\text{combinação}} \quad (34)$$

De (34) temos os novos autovalores que minimizam o erro, considerando os novos autovetores obtidos da combinação lineal dos autovetores das saídas estimadas por cada uma das técnicas. Estes novos autovalores serão expressos como uma combinação lineal dos correspondentes autovalores das saídas estimadas pelas técnicas.

Considerando duas técnicas temos:

pesos		**autovalores RBF**		**autovalores Neuro - fuzzy**		**autovalores Combinação**
q_1	*	$autovalor_{1_{RBF}}$		$autovalor_{1_{Neurofuzzy}}$		$autovalor_{1_{combinação}}$
q_2	*	$autovalor_{2_{RBF}}$	+	$autovalor_{2_{Neurofuzzy}}$	=	$autovalor_{2_{combinação}}$
⋮		⋮		⋮		⋮
q_n	*	$autovalor_{n_{RBF}}$		$autovalor_{n_{Neurofuzzy}}$		$autovalor_{n_{combinação}}$

(35)

De tal forma temos que:

$$Matriz de dados * autovetores_{combinação} = autovetores_{combinação} * autovalores_{combinação}$$

onde:

pesos		**autovetores RBF**		**pesos**		**autovetores Neuro - fuzzy**		**autovetores Combinação**
p1	*	**autovetor1**	+	p2	*	**autovetor1**	=	**autovetor1**
p3	*	**autovetor2**	+	p4	*	**autovetor2**	=	**autovetor2**
⋮		⋮		⋮		⋮	=	⋮
p_n	*	**autovetor_n**	+	p_n + 1	*	**autovetor_n**	=	**autovetor_n**

e

pesos		**autovalores RBF**		**autovalores Neuro - fuzzy**		**autovalores Combinação**
q_1	*	$autovalor_{1_{RBF}}$		$autovalor_{1_{Neurofuzzy}}$		$autovalor_{1_{combinação}}$
q_2	*	$autovalor_{2_{RBF}}$	+	$autovalor_{2_{Neurofuzzy}}$	=	$autovalor_{2_{combinação}}$
⋮		⋮		⋮		⋮
q_n	*	$autovalor_{n_{RBF}}$		$autovalor_{n_{Neurofuzzy}}$		$autovalor_{n_{combinação}}$

Os pesos das combinações dos autovetores e os autovalores são armazenados para em seguida serem usados na etapa da síntese.

3.1.5.3.1. Caso ilustrativo

Vejamos o procedimento anterior com um exemplo:

Seja o sistema $y = x^3 + 6000 * \sin(x)$, o sinal de entrada e saída são mostrados nas Figuras 30 e 31 respectivamente.

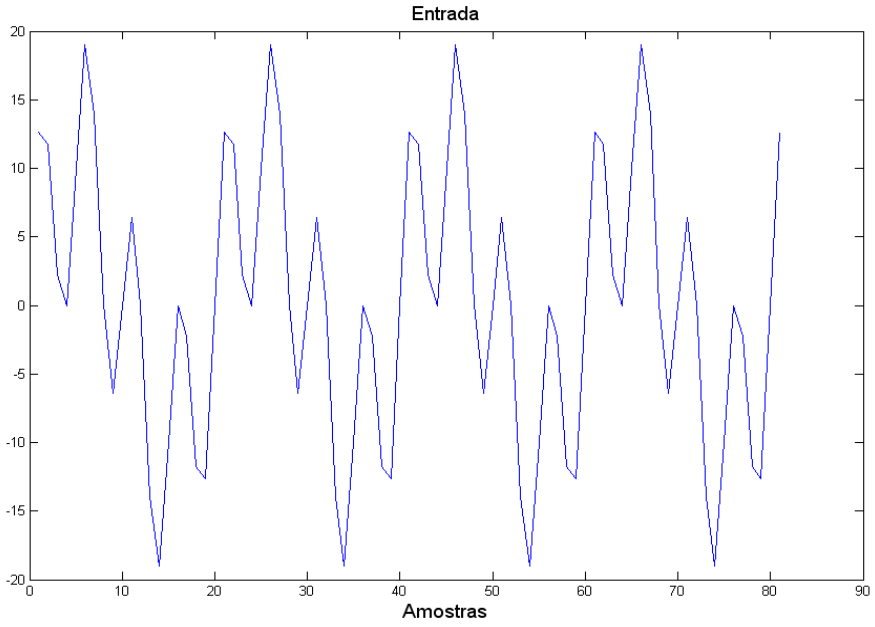

Figura 30 Sinal de Entrada

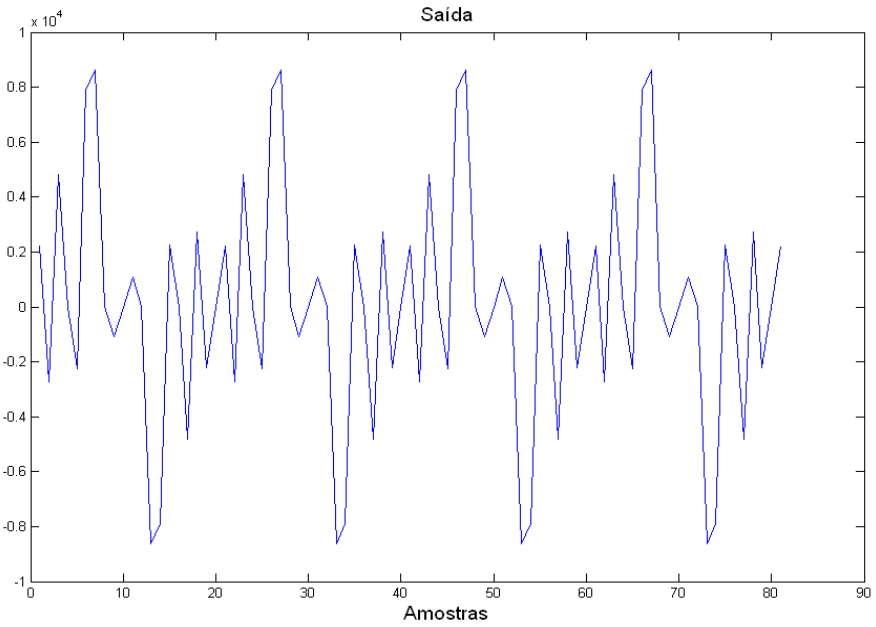

Figura 31 Sinal de Saída

Vamos usar as 40 primeiras amostras para o treinamento das técnicas e as 40 próximas amostras para testar os modelos obtidos.

Sendo:

$x = [12.6007 \quad 11.7557 \quad 2.2123 \quad -0.0000 \quad 10.0000 \quad 19.0211 \quad 13.9680 \quad 0.0000 \quad -6.4204 \quad -0.0000$
$6.4204 \quad 0.0000 \quad -13.9680 \quad -19.0211 \quad -10.0000 \quad -0.0000 \quad -2.2123 \quad -11.7557 \quad -12.6007 \quad -0.0000$
$12.6007 \quad 11.7557 \quad 2.2123 \quad -0.0000 \quad 10.0000 \quad 19.0211 \quad 13.9680 \quad 0.0000 \quad -6.4204 \quad -0.0000$
$6.4204 \quad 0.0000 \quad -13.9680 \quad -19.0211 \quad -10.0000 \quad -0.0000 \quad -2.2123 \quad -11.7557 \quad -12.6007 \quad -0.0000$
$12.6007];$

$y = [2206.9 \quad -2723.9 \quad 4817.9 \quad -0.0000 \quad -2264.1 \quad 7906.3 \quad 8639.6 \quad 0.0000 \quad -1085.3 \quad -0.0000 \quad 1085.3$
$0.0000 \quad -8639.6 \quad -7906.3 \quad 2264.1 \quad -0.0000 \quad -4817.9 \quad 2723.9 \quad -2206.9 \quad -0.0000 \quad 2206.9 \quad -2723.9$
$4817.9 \quad -0.0000 \quad -2264.1 \quad 7906.3 \quad 8639.6 \quad 0.0000 \quad -1085.3 \quad -0.0000 \quad 1085.3 \quad 0.0000 \quad -8639.6$
$-7906.3 \quad 2264.1 \quad -0.0000 \quad -4817.9 \quad 2723.9 \quad -2206.9 \quad -0.0000 \quad 2206.9];$

No processo de treinamento foram usadas as técnicas RBF e Neuro-fuzzy, obtendo-se as respectivas saídas estimadas (Figuras 32 e 33 respectivamente).

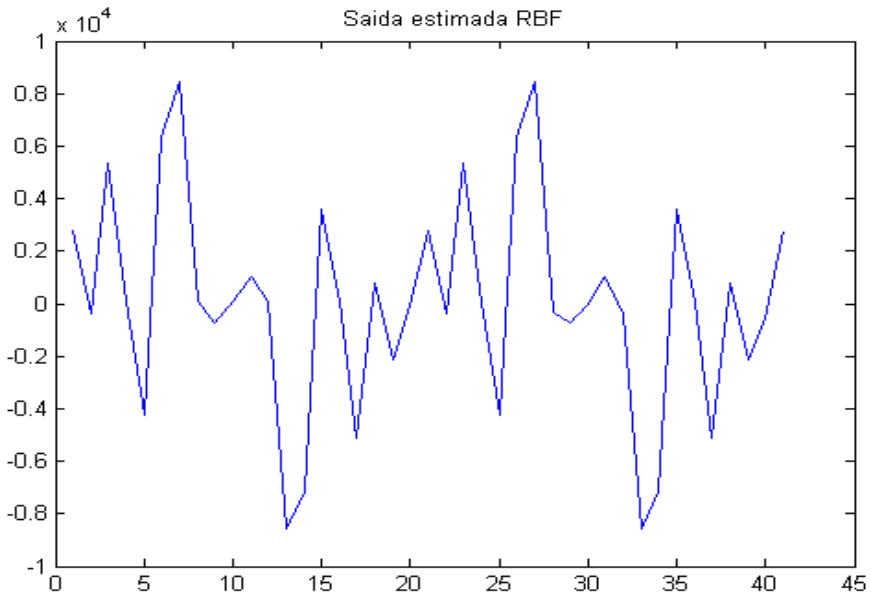

Figura 32 Saída estimada pela rede RBF

Figura 33 Saída estimada pelo Neuro-fuzzy

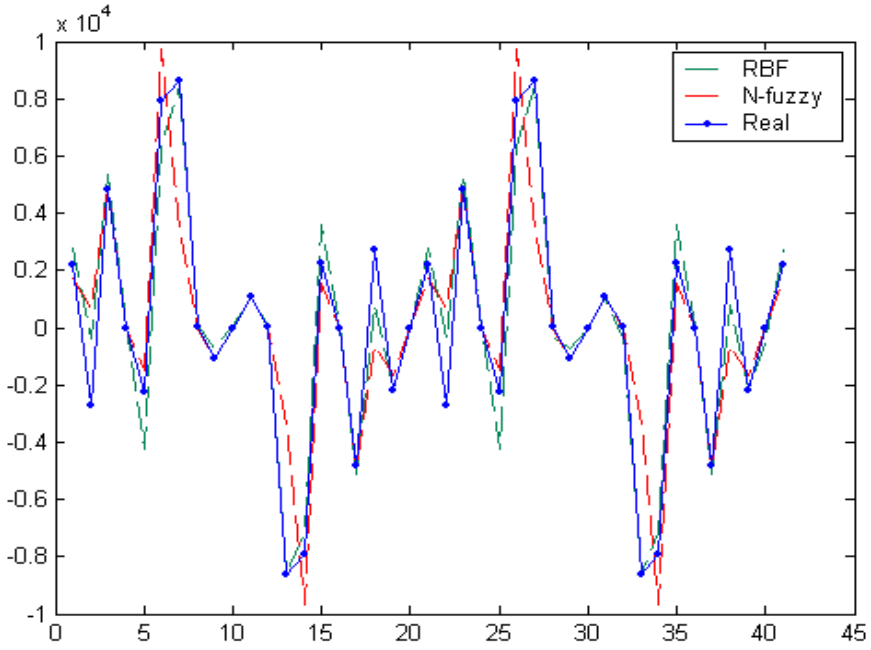

Figura 34 Comparação saídas estimadas

Erro quadrático médio RBF	**Erro quadrático médio Neuro-fuzzy**
982.2742	2.0957e+003

Tabela 1 Erros quadráticos médios RBF e Neuro-fuzzy

Cada uma das técnicas usadas (RBF e Neuro-fuzzy) usaram suas próprias partições, ajustando seus parâmetros automaticamente, mas é importante ressaltar que estes resultados podem ser melhorados ajustando os parâmetros manualmente num processo de tentativa – erro, o que foge ao escopo deste trabalho.

Preenchendo as matrizes de dados correspondentes a cada uma das saídas estimadas e à saída real temos:

$$M\_dados\_rbf = 1000 * \begin{bmatrix} 2.7526 & -0.3622 & 5.3537 & 0.0851 & \cdots & & & \cdots & -2.1576 & -0.0746 \\ -0.3622 & 5.3537 & 0.0851 & \cdots & & & \cdots & -2.1576 & -0.0746 & 2.7526 \\ 5.3537 & 0.0851 & & & & & -2.1576 & -0.0746 & 2.7526 & \vdots \\ 0.0851 & \cdots & & & & & & & & \\ \vdots & & & & \ddots & \ddots & \ddots & & & \\ \cdots & & & & & & & & \cdots & 3.5853 \\ \cdots & & & -2.1576 & -0.0746 & & & \cdots & 3.5853 & 0.0227 \\ \cdots & & -2.1576 & -0.0746 & 2.7526 & & \cdots & 3.5853 & 0.0227 & -5.1388 \\ -2.1576 & -0.0746 & 2.7526 & & & \cdots & 3.5853 & 0.0227 & -5.1388 & 0.8052 \\ -0.0746 & 2.7526 & & & \cdots & 3.5853 & 0.0227 & -5.1388 & 0.8052 & -2.1576 \end{bmatrix}$$

$$M\_dados\_Nfuzzy = 1000 * \begin{bmatrix} 1.7124 & 0.6567 & 4.8833 & -0.0014 & \cdots & & & \cdots & -1.6609 & -0.0014 \\ 0.6567 & 4.8833 & -0.0014 & \cdots & & & \cdots & -1.6609 & -0.0014 & 1.7124 \\ 4.8833 & -0.0014 & & & & & -1.6609 & -0.0014 & 1.7124 & \vdots \\ -0.0014 & \cdots & & & & & & & & \\ \vdots & & & & \ddots & \ddots & \ddots & & & \\ \cdots & & & & & & & & \cdots & \cdots \\ \cdots & & & -1.6609 & -0.0014 & & & \cdots & & -0.0014 \\ \cdots & & -1.6609 & -0.0014 & 1.7124 & & \cdots & & -0.0014 & -4.8794 \\ -1.6609 & -0.0014 & 1.7124 & & & \cdots & & -0.0014 & -4.8794 & -0.6022 \\ -0.0014 & 1.7124 & & & \cdots & & -0.0014 & -4.8794 & -0.6022 & -1.6609 \end{bmatrix}$$

$$M\_dados\_Real = 1000 * \begin{bmatrix} 2.2069 & -2.7239 & 4.8179 & 0 & \cdots & & & \cdots & -2.2069 & 0 \\ -2.7239 & 4.8179 & 0 & \cdots & & & \cdots & -2.2069 & 0 & 2.2069 \\ 4.8179 & 0 & & & & & -2.2069 & 0 & 2.2069 & \vdots \\ 0 & \cdots & & & & & & & & \\ \vdots & & & & \ddots & \ddots & \ddots & & & \\ \cdots & & & & & & & & \cdots & \\ \cdots & & & -2.2069 & 0 & & & \cdots & & 0 \\ \cdots & & -2.2069 & 0 & 2.2069 & & & \cdots & 0 & -4.8179 \\ -2.2069 & 0 & 2.2069 & & & \cdots & & 0 & -4.8179 & 2.7239 \\ 0 & 2.2069 & & & \cdots & & 0 & -4.8179 & 2.7239 & -2.2069 \end{bmatrix}$$

Aplicando o EIG a cada matriz de dados obtida anteriormente temos:

$$[\text{autovetores}_{\text{real}}, \text{autovalores}_{\text{real}}] = \text{EIG}(\text{M\_dados\_real})$$

$$[\text{autovertores}_{\text{Rbf}}, \text{autovalores}_{\text{Rbf}}] = \text{EIG}(\text{M\_dados\_rbf})$$

$$[\text{autovetores}_{\text{Nfuzzy}}, \text{autovalores}_{\text{Nfuzzy}}] = \text{EIG}(\text{M\_dados\_nfuzzy})$$

Na Tabela 2 podem-se observar os erros quadráticos médios dos autovetores obtidos da técnica RBF e do modelo Neuro-fuzzy, respeito dos originais.

Autovetor	**Erros Quadráticos Médios Autovetores RBF**	**Erros Quadráticos Médios Autovetores Neuro-fuzzy**
1	0.3179	0.4472
2	0.3158	0.3162
3	0.3052	3.2380e-4
4	0.0054	0.3162
5	0.0188	0.3162
6	0.3161	0.3162
7	0.3080	0.3165
8	0.3172	0.3162
9	0.3158	0.3214
10	0.4467	0.3278
11	0.4455	0.3043
12	0.3165	0.3109
13	0.3132	0.3162
14	0.3239	0.3165
15	0.3167	0.3162
16	0.4468	0.3162
17	0.4472	0.3162
18	0.3271	0.4472
19	0.3154	0.3162
20	0.3143	1.8126e-4

Tabela 2 Erros quadráticos médios dos autovetores RBF e Neuro-fuzzy

Na Figura 35 e 36 podem-se visualizar os resultados da Tabela 2.

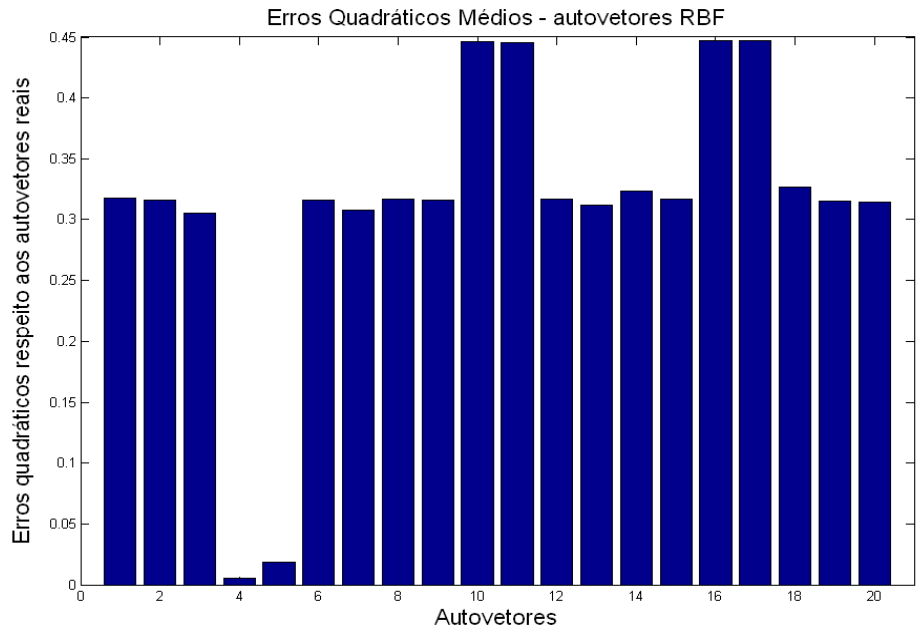

Figura 35 Erros quadráticos dos autovetores da saída estimada RBF com respeito aos autovetores reais

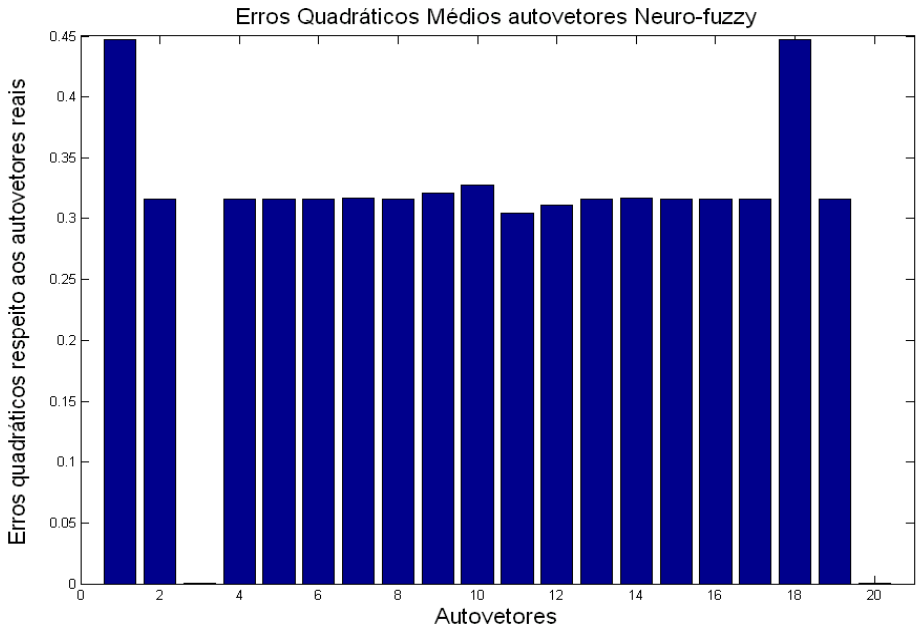

Figura 36 Erros quadráticos dos autovetores da saída estimada Neuro-fuzzy com respeito aos autovetores reais

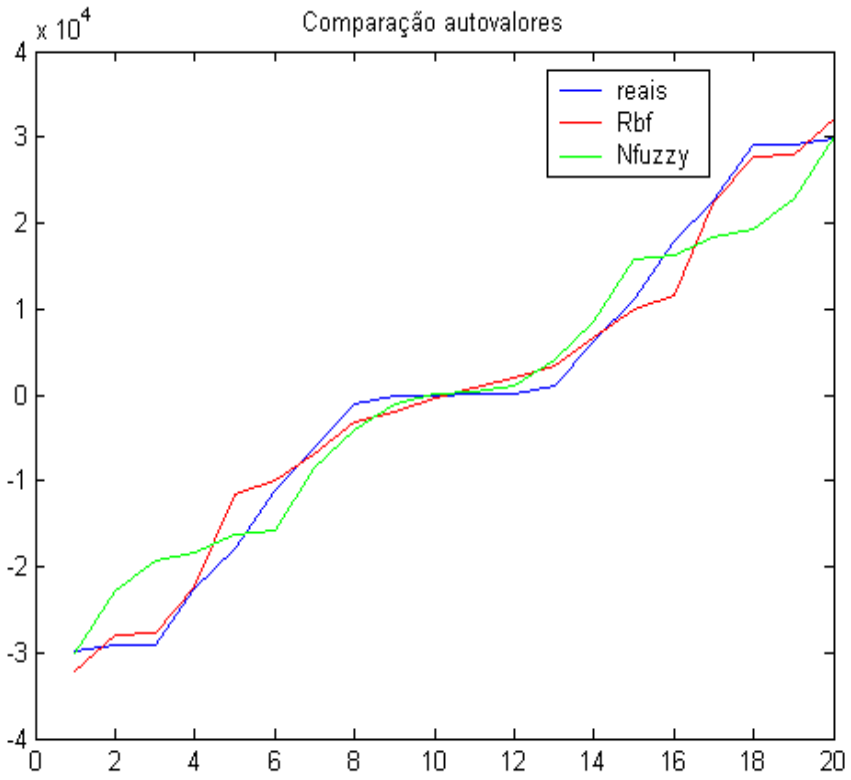

Figura 37 Comparação autovalores

Erro quadrático médio RBF	**Erro quadrático médio Neuro-fuzzy**
2.4607e+003	4.4110e+003

Tabela 3 Erros quadráticos médios RBF e Neuro-fuzzy

As combinações utilizadas para os autovetores e seus respectivos pesos são mostrados a seguir:

pesos		autovetores RBF		pesos		autovetores Neuro- fuzzy		autovetores Combinados	erros
0.0000	*	autovetor1	+	−1.0000	*	autovetor1	=	autovetor1	0.0008
0.0026	*	autovetor2	+	0.0000	*	autovetor2	=	autovetor2	1.4124
0.0000	*	autovetor3	+	1.0000	*	autovetor3	=	autovetor3	0.0014
0.9997	*	autovetor4	+	−0.0052	*	autovetor4	=	autovetor4	0.0234
0.9965	*	autovetor5	+	0.0008	*	autovetor5	=	autovetor5	0.0840
0.0010	*	autovetor6	+	0.0000	*	autovetor6	=	autovetor6	1.4135
0.0514	*	autovetor7	+	−0.0010	*	autovetor7	=	autovetor7	1.3774
−0.0064	*	autovetor8	+	0.0000	*	autovetor8	=	autovetor8	1.4097
0.0028	*	autovetor9	+	−0.0331	*	autovetor9	=	autovetor9	⇒ 1.3905
−0.9962	*	autovetor10	+	−0.0800	*	autovetor10	=	autovetor10	0.0448
−0.9854	*	autovetor11	+	0.0799	*	autovetor11	=	autovetor11	0.1540
−0.0019	*	autovetor12	+	0.0332	*	autovetor12	=	autovetor12	1.3905
0.0189	*	autovetor13	+	−0.0001	*	autovetor13	=	autovetor13	1.4008
−0.0492	*	autovetor14	+	−0.0015	*	autovetor14	=	autovetor14	1.3790
−0.0027	*	autovetor15	+	0.0000	*	autovetor15	=	autovetor15	1.4123
−0.9997	*	autovetor16	+	0.0005	*	autovetor16	=	autovetor16	0.0833
0.0000	*	autovetor17	+	−0.0024	*	autovetor17	=	autovetor17	0.0233
0.0051	*	autovetor18	+	−1.0000	*	autovetor18	=	autovetor18	0.0014
0.0000	*	autovetor19	+	0.0000	*	autovetor19	=	autovetor19	1.4106
0.0000	*	autovetor20	+	1.0000	*	autovetor20	=	autovetor20	0.0008

A coluna “erros” mostra a norma euclidiana do vetor diferença entre o autovetor resultante da combinação e seu respectivo autovetor real. Lembremos que a Tabela anterior pode-ser representada pela equação

$$\begin{bmatrix} p1_j & p2_j & \cdots & pk_j \end{bmatrix} \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \\ \vdots \\ \mathbf{u}k_j \end{bmatrix} = \begin{bmatrix} \mathbf{u}^e_j \end{bmatrix}$$

onde :

j: índice do autovetor

k=2 (número de técnicas)

p1,p2,..,pk: pesos

u: autovetores das técnicas (técnicas 1,2,..,k)

$\mathbf{u}^{r}_{j}$: autovetor verdadeiro

Logo os pesos são calculados aplicando a inversa generalizada, como segue (Considerando o número de técnicas "k=2"):

$$\begin{bmatrix} p1_j & p2_j \end{bmatrix}[\mathbf{I}] = \begin{bmatrix} \mathbf{u}^{r}_{j} \end{bmatrix} * \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \end{bmatrix}^t * \left(\begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \end{bmatrix} * \begin{bmatrix} \mathbf{u}1_j \\ \mathbf{u}2_j \end{bmatrix}^t \right)^{-1}$$

Neste caso de duas técnicas, devido ao processo de inversa generalizada, pode-se ter um peso de uma técnica igual a "0" e o pesos da outra técnica igual a "1", o que indicaria que o autovetor da técnica de peso "1" estaria representando o 100 por cento do autovetor real, enquanto o autovetor da outra técnica não estaria contribuindo ao processo, ou como poe-se se ver na Tabela mostrada anteriormente, pode-se ter pesos quase "1" e pesos quase "0", o que indicaria que o autovetor de uma técnica estaria contribuindo em maior porcentagem na combinação lineal que o autovetor da outra técnica, mas ambas técnicas estariam contribuindo em determinadas porcentagens. Mas este procedimento para calcular os pesos, usando a inversa generalizada não assegura zerar o erro entre o vetor resultante de combinação lineal e o autovetor real correspondente, por tal motivo na coluna de erros da Tabela anterior não temos nenhum erro de valor "zero", mas se erros próximos de zero, como o caso dos autovetores de posição 1,3,18 e 20. O processo de inversa generalizada não zera os erros, mas calcula os pesos de forma a minimiza-los.

A combinação dos autovetores da posição 1 está mais próxima do vetor autovetor real correspondente (maior importância aos autovetores de maior energia que tiveram menor erro). Logo em função dele e seguindo as condições de ortonormalização são calculados novamente os pesos dos autovetores das posições restantes, obtendo-se os novos pesos e autovetores:

pesos		autovetores RBF		pesos		autovetores Neuro - fuzzy		autovetores Combinados	erros
0.0000	*	autovetor 1	+	– 1.0000	*	autovetor 1	=	autovetor 1	0.0008
0.0066	*	autovetor 2	+	0.0048	*	autovetor 2	=	autovetor 2	0.1160
0.0134	*	autovetor 3	+	0.8456	*	autovetor 3	=	autovetor 3	0.0003
0.9000	*	autovetor 4	+	– 0.6672	*	autovetor 4	=	autovetor 4	0.1158
0.5600	*	autovetor 5	+	0.0231	*	autovetor 5	=	autovetor 5	0.1153
0.0423	*	autovetor 6	+	0.0082	*	autovetor 6	=	autovetor 6	0.1160
0.0320	*	autovetor 7	+	– 0.0058	*	autovetor 7	=	autovetor 7	0.1165
– 0.0034	*	autovetor 8	+	0.0022	*	autovetor 8	=	autovetor 8	0.1161
0.0045	*	autovetor 9	+	– 0.0452	*	autovetor 9	=	autovetor 9	⇒ 0.1214
– 0.7890	*	autovetor 10	+	– 0.0760	*	autovetor 10	=	autovetor 10	0.1281
– 0.6926	*	autovetor 11	+	0.1400	*	autovetor 11	=	autovetor 11	0.1038
– 0.0069	*	autovetor 12	+	0.0690	*	autovetor 12	=	autovetor 12	0.1109
0.0048	*	autovetor 13	+	– 0.0018	*	autovetor 13	=	autovetor 13	0.1162
– 0.0441	*	autovetor 14	+	– 0.0022	*	autovetor 14	=	autovetor 14	0.1159
– 0.0310	*	autovetor 15	+	0.0062	*	autovetor 15	=	autovetor 15	0.1160
– 0.4845	*	autovetor 16	+	0.0400	*	autovetor 16	=	autovetor 16	0.1171
0.7644	*	autovetor 17	+	– 0.4568	*	autovetor 17	=	autovetor 17	0.1163
0.0100	*	autovetor 18	+	0.6693	*	autovetor 18	=	autovetor 18	0.0005
0.0055	*	autovetor 19	+	0.0064	*	autovetor 19	=	autovetor 19	0.1162
0.0012	*	autovetor 20	+	0.0523	*	autovetor 20	=	autovetor 20	0.0014

Conhecendo a matriz de dados do sinal real e os autovetores achados anteriormente (resultados da combinação linear), é aplicado o processo inverso do EIG para obter os respectivos novos autovalores:

$$[\text{autovetores}, \text{autovalores}] = \text{EIG}(\text{Matriz\_de\_dados})$$

$$\text{Matriz\_de\_dados} * \text{autovetores} = \text{autovetores} * \text{autovalores}$$

$$\Rightarrow \text{autovalores} = \text{autovetores}^{-1}_{\text{combinação}} * \text{Matriz\_de\_dados} * \text{autovetores}_{\text{combinação}} \tag{36}$$

Da equação (36) se obtém a matriz de autovalores, mas esta matriz apresenta elementos diferentes de zero (valores pequenos comparados com os elementos da diagonal principal), fora da diagonal principal, estes pequenos valores são conseqüência do uso da inversa generalizada, usada no cálculo dos pesos dos autovetores. Para os cálculos a seguir, só foram considerados os elementos da diagonal principal, assumindo os restantes iguais a zero (por serem valores muitos pequenos) (37).

$$\text{matriz\_autovalores} = \text{autovalores}_{\text{combinação}}(f,c)$$

$$\text{onde} \begin{cases} \text{se } f = c \quad \text{autovalores}_{\text{combinação}}(f,c) = \text{autovalores } (f,c) \\ \text{se } f \neq c \quad \text{autovalores}_{\text{combinação}}(f,c) = 0 \end{cases} \tag{37}$$

Logo os autovalores da combinação são:

$$\text{autovalores}_{\text{combinação}}(f,c)_{f=c} = 1.0e+004 * \begin{bmatrix} -2.9776 \\ -2.2591 \\ -2.9136 \\ -2.9200 \\ -1.1077 \\ -0.1082 \\ 0.6103 \\ -1.7874 \\ 0.0184 \\ -0.0000 \\ -0.0000 \\ -0.0184 \\ 1.7874 \\ -0.6103 \\ 0.1082 \\ 1.1077 \\ 2.9200 \\ 2.9136 \\ 2.2591 \\ 2.9776 \end{bmatrix}$$

Expressando estes autovalores da combinação em função dos autovalores obtidos dos sinais estimados pelas respectivas técnicas, temos:

$$[K]*[\text{autovalores\_RBF}]+[\text{autovalores\_Nfuzzy}]=[\text{autovalores}_{\text{combinação}}]$$

Observe-se da equação anterior, que somente será preciso calcular o valor da variável de pesos "K", pois neste caso estão sendo usadas só duas técnicas.

No caso de trabalhar com mais de duas técnicas a equação seria:

$$[k1]*[\text{autovalores\_técnica}_1]+[k2]*[\text{autovalores\_técnica}_2]+\cdots$$

$$\ldots+[k_{n-1}]*[\text{autovalores\_técnica}_{n-1}]+[\text{autovalores\_técnica}_n]$$

onde precisa-se calcular os valores das variáveis $k1, k2, ..., k_{n-1}$.

Voltando ao nosso caso exemplo, temos:

$$[K] = ([\text{autovalores}_{\text{combinação}}] - [\text{autovalores\_Nfuzzy}]) * [\text{autovalores\_RBF}]^{-1}$$

Logo:

$$k(f,c)_{f=c} = [-0.0053 \; -0.0050 \;\; 0.3524 \;\; 0.4836 \;\; -0.4536 \; -1.4816 \;\; -2.1525 \;\; 4.2030$$
$$-0.5968 \;\; 0.4119 \; -0.2757 \;\; -0.5981 \;\; 4.1560 \;\; -2.1557 \; -1.4826 \; -0.4532$$
$$0.4835 \;\; 0.3527 \; -0.0050 \;\; -0.0053]$$

Tendo os autovetores e autovalores da combinação tem-se a matriz de dados como segue:

$$\text{Matriz\_dados\_combinação} =$$

$$\text{autovetores}_{\text{combinação}} * \text{autovelores}_{\text{combinação}} * \text{autovetores}^{-1}_{\text{combinação}}$$

$$\text{Matriz\_dados\_combinação} =$$

$$1000 * \begin{bmatrix} 2.2069 & -2.7239 & 4.8179 & 0 & \cdots & & & & \cdots & -2.2069 & 0 \\ -2.7239 & 4.8179 & 0 & \cdots & & & \cdots & & -2.2069 & 0 & 2.2069 \\ 4.8179 & 0 & & & & & & -2.2069 & 0 & 2.2069 & \vdots \\ 0 & \cdots & & & & & & & & & \\ \vdots & & & & \ddots & & \ddots & & \ddots & & \\ \cdots & & & & & & & & & \cdots & \\ \cdots & & -2.2069 & 0 & & & \cdots & & & 0 & \\ \cdots & -2.2069 & 0 & 2.2069 & & & & \cdots & & 0 & -4.8179 \\ -2.2069 & 0 & 2.2069 & & & \cdots & & & 0 & -4.8179 & 2.7239 \\ 0 & 2.2069 & & & \cdots & & & 0 & -4.8179 & 2.7239 & -2.2069 \end{bmatrix}$$

Da matriz anterior tem-se o sinal correspondente à combinação dos autovetores e autovalores das técnicas, cada diagonal paralela à diagonal secundária deverá ser formada por elementos iguais, representando o valor da amostra do sinal.

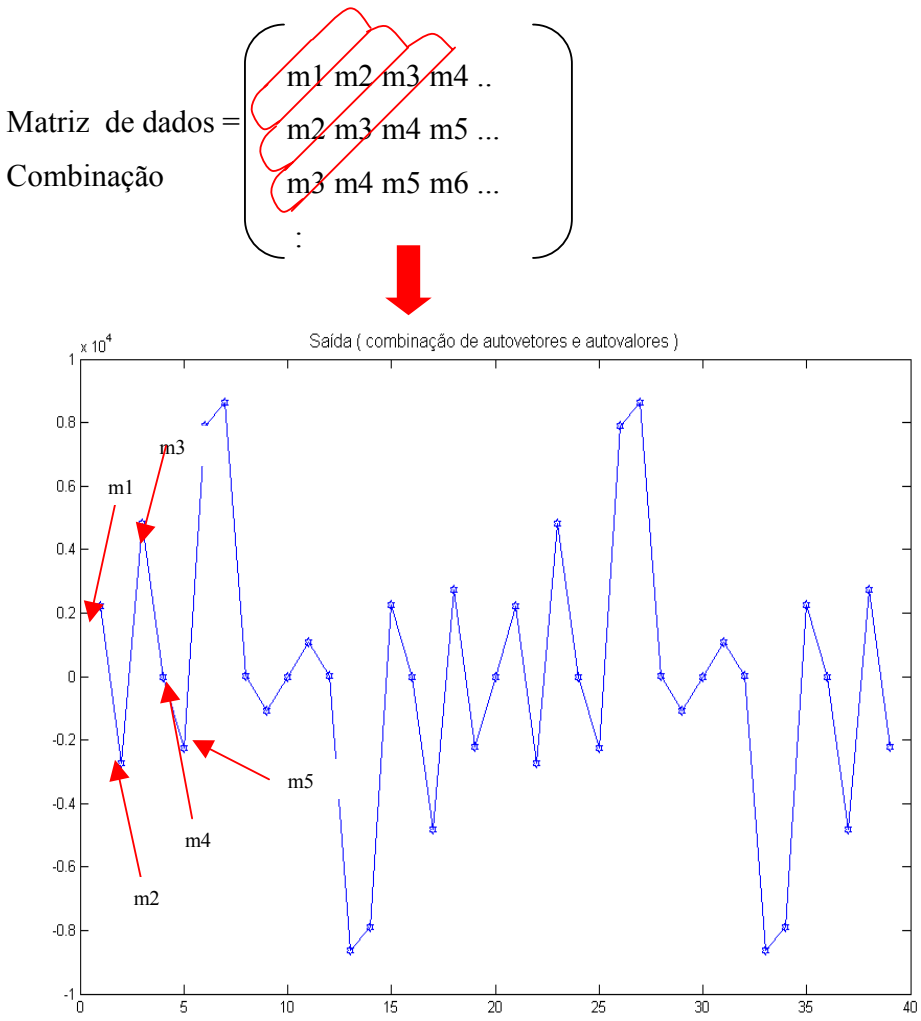

Figura 38 Novas saídas estimadas das recombinações

A Tabela 4 apresenta os erros quadráticos médios obtidos pelas técnicas RBF, Neuro-fuzzy e da combinação de autovetores e autovalores.

Erro quadrático médio RBF	**Erro quadrático médio Neuro-fuzzy**	**Erro quadrático médio Combinação**
982.2742	2.0957e+003	15.3849

Tabela 4 Erros quadráticos médios RBF, Neuro-fuzzy, combinação

A Figura 39 mostra a estimação das 40 amostras seguintes, pelas técnicas e pela combinação de autovetores e autovalores.

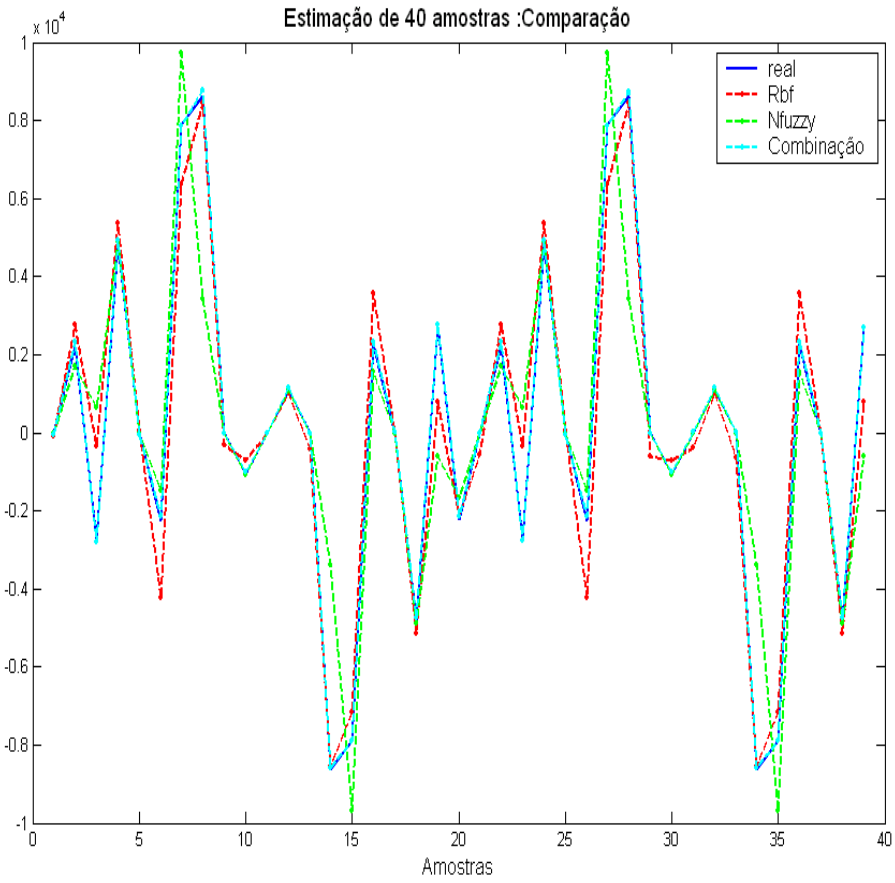

Figura 39 Comparação saídas estimadas

Erro quadrático médio RBF	Erro quadrático médio Neuro-fuzzy	Erro quadrático médio Combinação
998.5032	2.0938e+003	88.4260

Tabela 5 Erros quadráticos médios RBF, Neuro-fuzzy, combinação

Para cada uma das partições feitas, haverá uma saída constituída pela combinação de autovetores e autovalores das técnicas convencionais usadas, os pesos da combinação de autovetores e autovalores serão armazenados, para logo serem usados na etapa da síntese.

Os resultados alcançados mostram claramente a efetividade do processo proposto, em que a combinação de resultados de técnicas individuais supera o de cada uma de sua componentes. O caso da síntese será visto a seguir.

3.1.5.4.
Conferindo se o erro de modelagem foi satisfeito

O passo seguinte é identificar se alguma partição dos sinais satisfaz o erro de modelagem desejado, se isto acontece quer dizer que a modelagem da partição está concluída, tendo ao final do processo uma saída constituída pela superposição das saídas estimadas por diversas técnicas.
No caso que o erro de modelagem não tenha sido satisfeito em alguma das partições, será feita uma sub-partição como é explicada na continuação.

3.1.5.5.
Estrutura Piramidal

Se a modelagem de uma partição não satisfez o erro de modelagem desejado, então, o passo seguinte é modelar os respectivos resíduos. Para a modelagem destes resíduos será feita uma nova partição do sinal de entrada e dos resíduos respectivos, de tal forma que a modelagem conseguirá alcançar características cada vez mais específicas do sinal. A modelagem será feita de forma piramidal, começando das características gerais do sinal até as características cada vez mais específicas.

A Figura 40 mostra o esquema piramidal ao qual se fez referência.

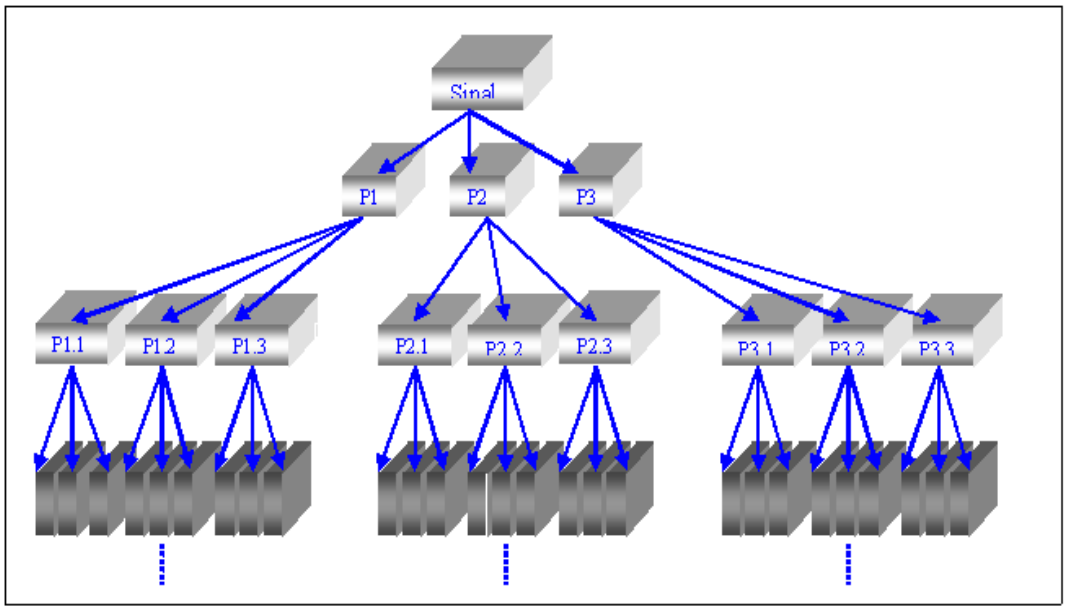

Figura 40 Esquema Piramidal

Observe-se que este procedimento é feito em cada uma das partições realizadas.

Na Figura 41 pode-se observar em mais detalhe a lógica de modelagem das partições e sub-partições.

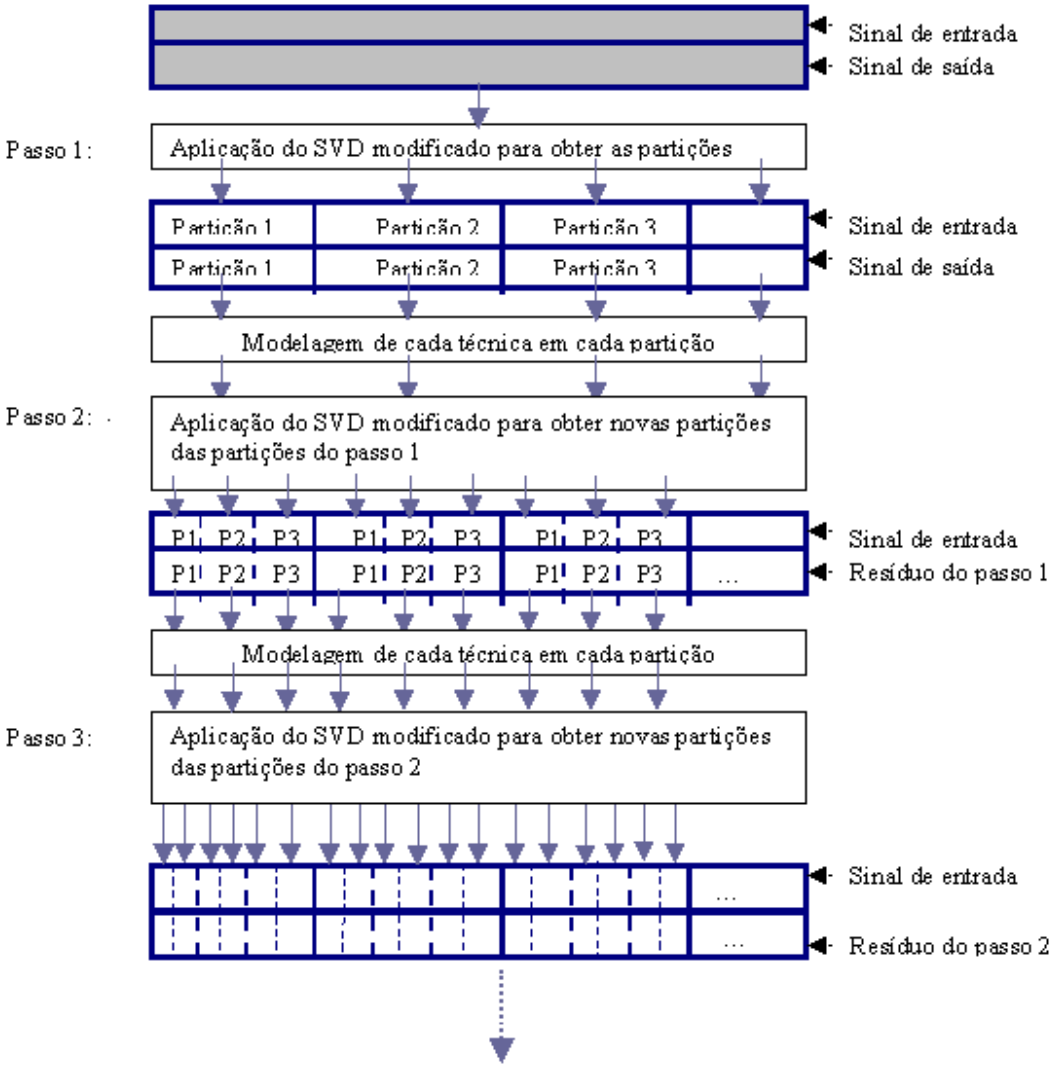

Figura 41 Processo iterativo de partições (estrutura piramidal)

Este procedimento será efetuado sucessivamente até satisfazer o erro esperado em cada uma das partições e/ou sub-partições feitas, ou até que não sejam mais possíveis sub-partições, caso quando existem partições com um só elemento (critério de parada).

A Figura 42 mostra as partições e sub-partições realizadas num sinal até satisfazer o erro de modelagem esperado.

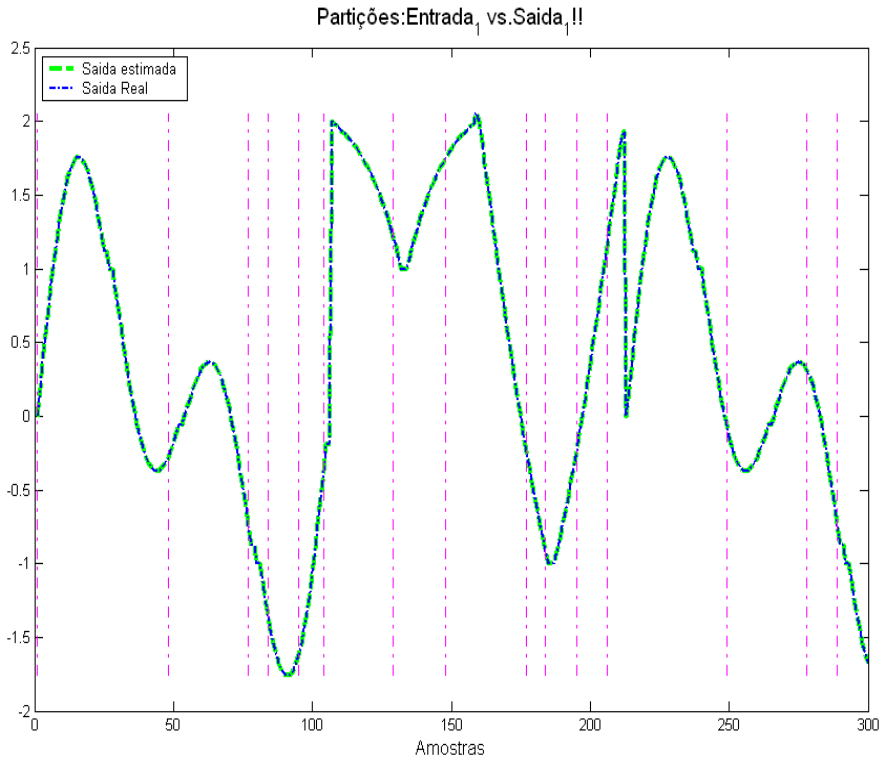

Figura 42 Partições e sub-partições

Quanto maior o nível de sub-partições, tanto menor é o erro final. Este maior número de níveis nas partições e sub-partições lembram a modelagem de sistemas não lineares através de linearizações por partes.

Um artifício utilizado na programação desenvolvida para evitar situações onde tenham-se partições de um só elemento, é gerar amostras por interpolação. Este procedimento de interpolação melhora a modelagem, mas não assegura totalmente que não se tenham partições de um elemento. Estas interpolações apresentam outras vantagens, por exemplo, quando não se conta com um número de amostras

significativo para realizar as partições. Na seção da síntese o assunto será discutido em detalhes.

O erro de modelagem esperado não pode ser muito grande, pois acarretaria baixa capacidade de síntese do sinal verdadeiro. Isto será detalhado na Seção 3.2, onde se trata da síntese do sinal de saída.

3.2.
Síntese: Descrição do Sistema

O modelo utilizado para a síntese das saídas será obtido de cada técnica em cada partição e sub-partições dos espaços estudados. O primeiro passo neste processo é a identificação a qual partição corresponde a amostra de entrada considerada. A notação considerada é ilustrada a seguir em um processo típico:

$Y = Y^{P1}$ → saída modelada da partiçao 1

$Y = Y^{P2}$ → saída modelada da partiçao 2

$Y = Y^{P3}$ → saída modelada da partiçao 3

$Y = Y^{P4}$ → saída modelada da partiçao 4

$Y = Y^{P5}$ → saída modelada da partiçao 5

.

.

.

O procedimento é naturalmente recursivo, sendo em seguida identificada a sub-partição para modelar o resíduo e assim sucessivamente em cada uma das sub-partições seguintes.

Cada uma das modelagens das partições contem informação das modelagens feitas por cada uma das técnicas usadas (rede RBF, Neuro-fuzzy,..), esta informação é representada pelos parâmetros respectivos e pelos pesos das combinações dos autovetores e autovalores.

$$Y^{P1.*} = Y_{Rbf} + Y_{Fuzzy} + \ldots$$

A identificação da partição é discutida na seção que segue.

3.2.1. Identificação da Partição

A identificação da partição é o ponto mais importante do processo de síntese e por isto será detalhado para sua melhor compreensão. Este processo segue o de análise, onde uma vez feito o treinamento dos dados do sinal de entrada e do de saída, temos armazenado em disco os modelos das técnicas e os pesos das combinações respectivas de autovetores e autovalores. O processo de síntese deve estimar a amostra de saída correspondente a cada nova amostra da entrada.

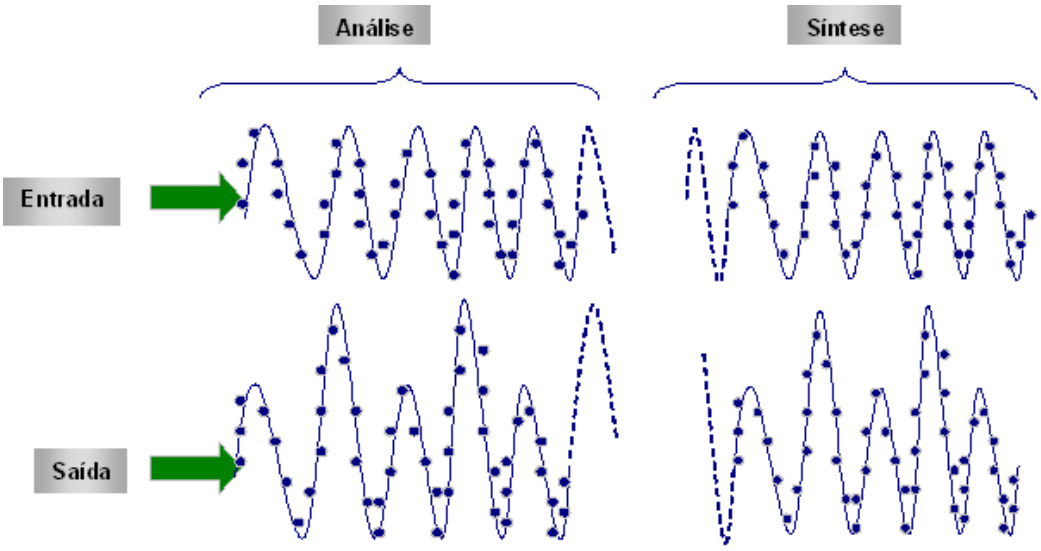

Figura 43 Etapas análise e síntese

Como indicado na Figura 43, mostramos as duas etapas do funcionamento do algoritmo. A primeira etapa (análise) foi detalhada anteriormente e a segunda etapa (síntese) baseia-se na identificação da partição, sendo discutida a seguir (para facilitar a compreensão do algoritmo os gráficos seguintes mostram ambas as etapas: análise e síntese).

Lembremos que, quando foi feito o mapeamento dos sinais de entrada e saída dos dados de treinamento, os vetores característicos (autovetores e autovalores associados) obtidos foram armazenados assim como os parâmetros das técnicas obtidos para modelar cada partição.

Na etapa da síntese vamos preencher uma nova matriz de dados com o mesmo número de amostras que foi utilizado quando do mapeamento entrada-saída, isto é: se o número de amostras utilizado foi cinco, então vamos usar as cinco últimas amostras do sinal de entrada, onde a última amostra usada seria a amostra cuja saída deseja-se estimar. A Figura 44 resume este passo.

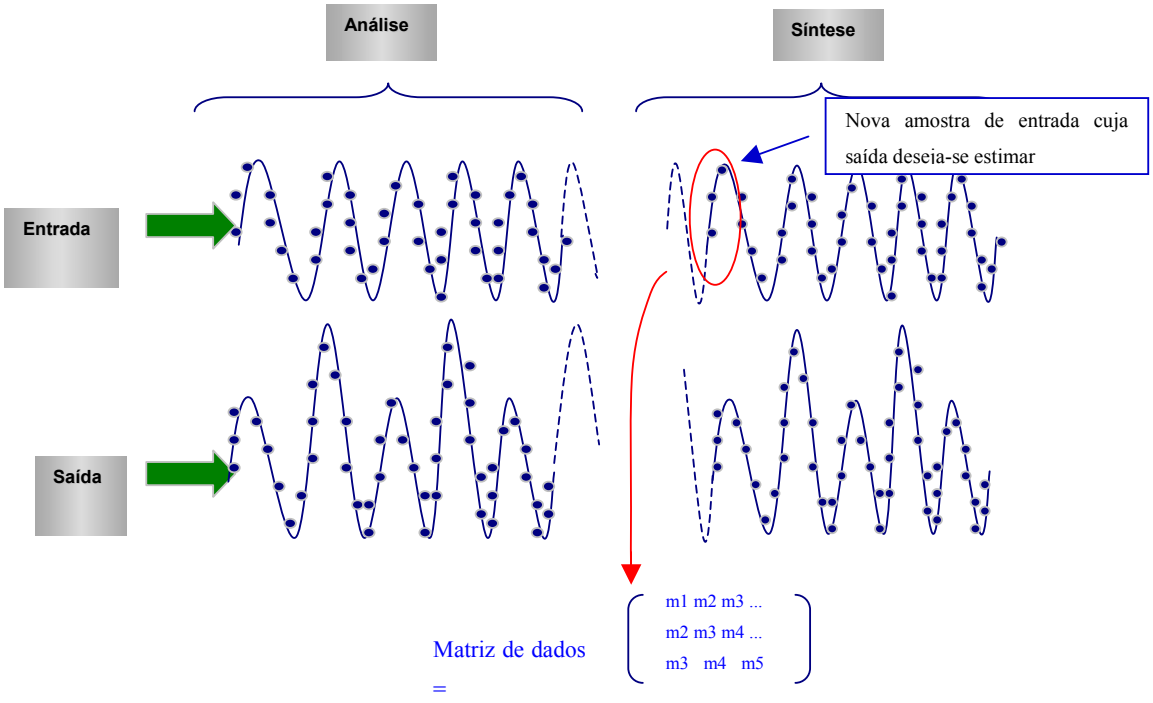

Figura 44 Preenchimento da matriz de dados com as "M-1" últimas amostras do sinal de entrada

Aplica-se o SVD a esta nova matriz de dados, gerando como resultado os vetores característicos representativos da matriz do sinal (Figura 45).

Figura 45 Vetor característico da matriz de dados das "M-1"últimas amostras do sinal de entrada

Com cada vetor característico da matriz de dados das últimas "M-1" amostras do sinal de entrada, procuramos no correspondente mapeamento do sinal respectivo (feito na etapa do treinamento) qual seria o vetor característico mais parecido com

este (para facilitar a programação os vetores característicos correspondentes a uma matriz de dados foram arrumados numa só linha: [autovetor1 autovalor1 autovetor2 autovalor3 autovetor4 autovalor4 ....]). Uma vez localizado, identifica-se a posição da amostra no sinal de entrada a que se refere. Isto indica a partição a que deve pertencer o vetor de entrada.

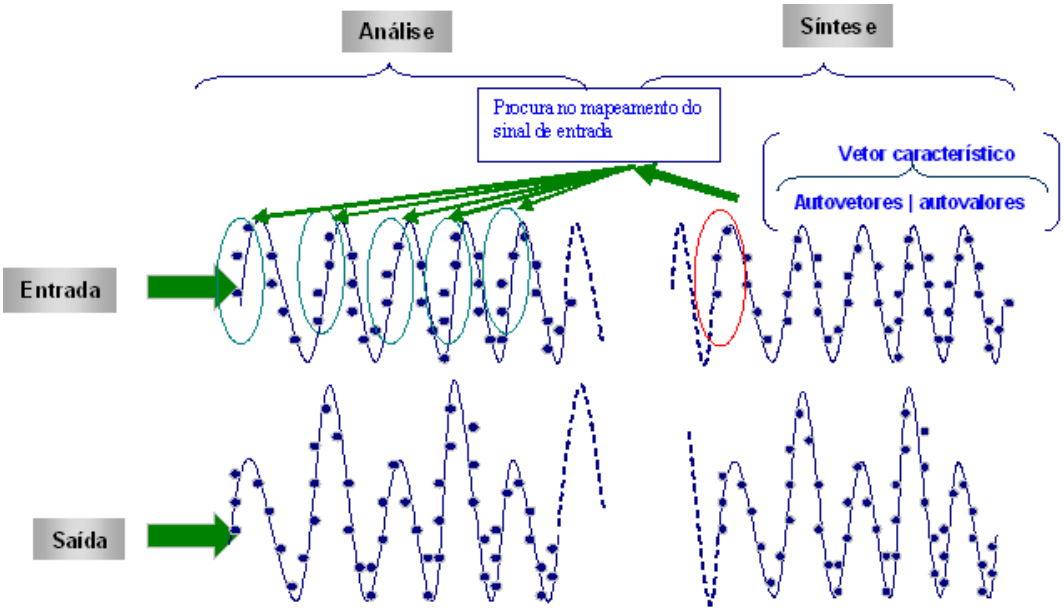

Figura 46 Partições identificadas nos dados de treinamento do sinal de entrada

O caso mais comum, entretanto, é que a identificação da partição correspondente tenha mais de uma solução (Figura 46). Isto exige identificar de todas as possíveis soluções anteriores qual delas é a mais apropriada para representar o comportamento das "M-1" últimas amostras do sinal de entrada. Observe-se que as correspondentes amostras do sinal de entrada, representam as possíveis partições candidatas a representar o conjunto de entrada (Figura 47).

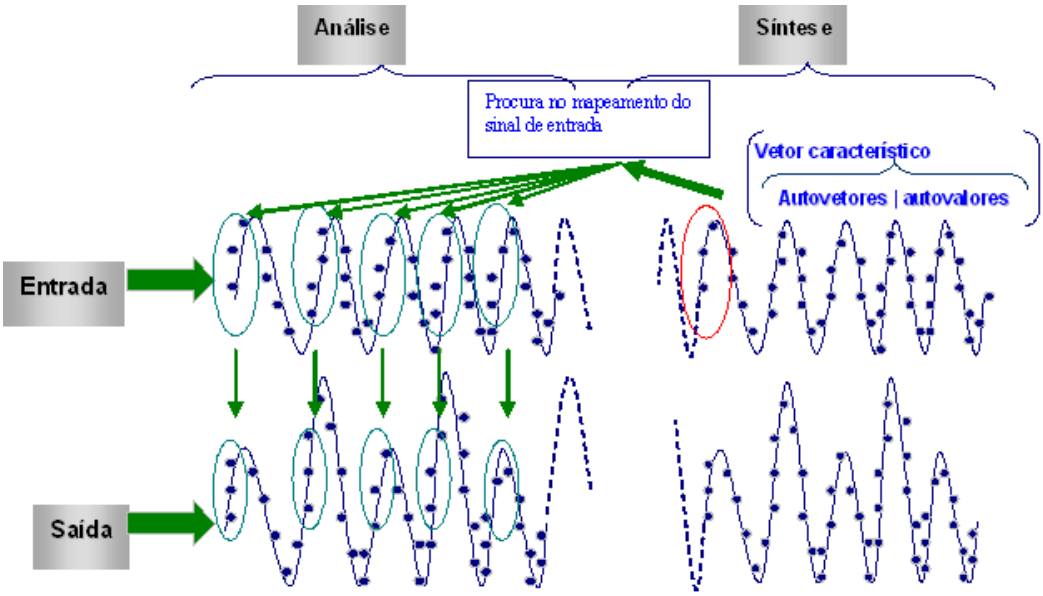

Figura 47 Respectivas amostras no sinal de saída das amostras identificadas no sinal de entrada

A seguir, aplica-se o SVD à matriz de dados correspondentes às "M-1" últimas amostras existentes do sinal de saída. Os correspondentes vetores característicos são comparados aos vetores característicos das amostras do sinal de saída anteriormente identificadas. O objetivo é reforçar o reconhecimento do vetor, agora visto como um dos elementos da trajetória do sinal. Pode acontecer que ainda assim exista mais de uma possível solução, pois a diferença entre as soluções é pequena, de tal forma que nenhuma delas está suficientemente próxima do ideal e significativamente afastada das demais (a solução escolhida será aquela que estiver a uma distância igual ou menor à metade da distância da solução que está mais afastada), exigindo que o algoritmo, recursivamente, passe a considerar progressivamente instantes anteriores da trajetória. Isto é feito até que se defina uma única solução (Figura 48). Prevendo que esta condição pode não ser satisfeita, o usuário determina um número máximo de iterações, ao final das quais será considerada como melhor solução aquela que apresentou o menor erro.

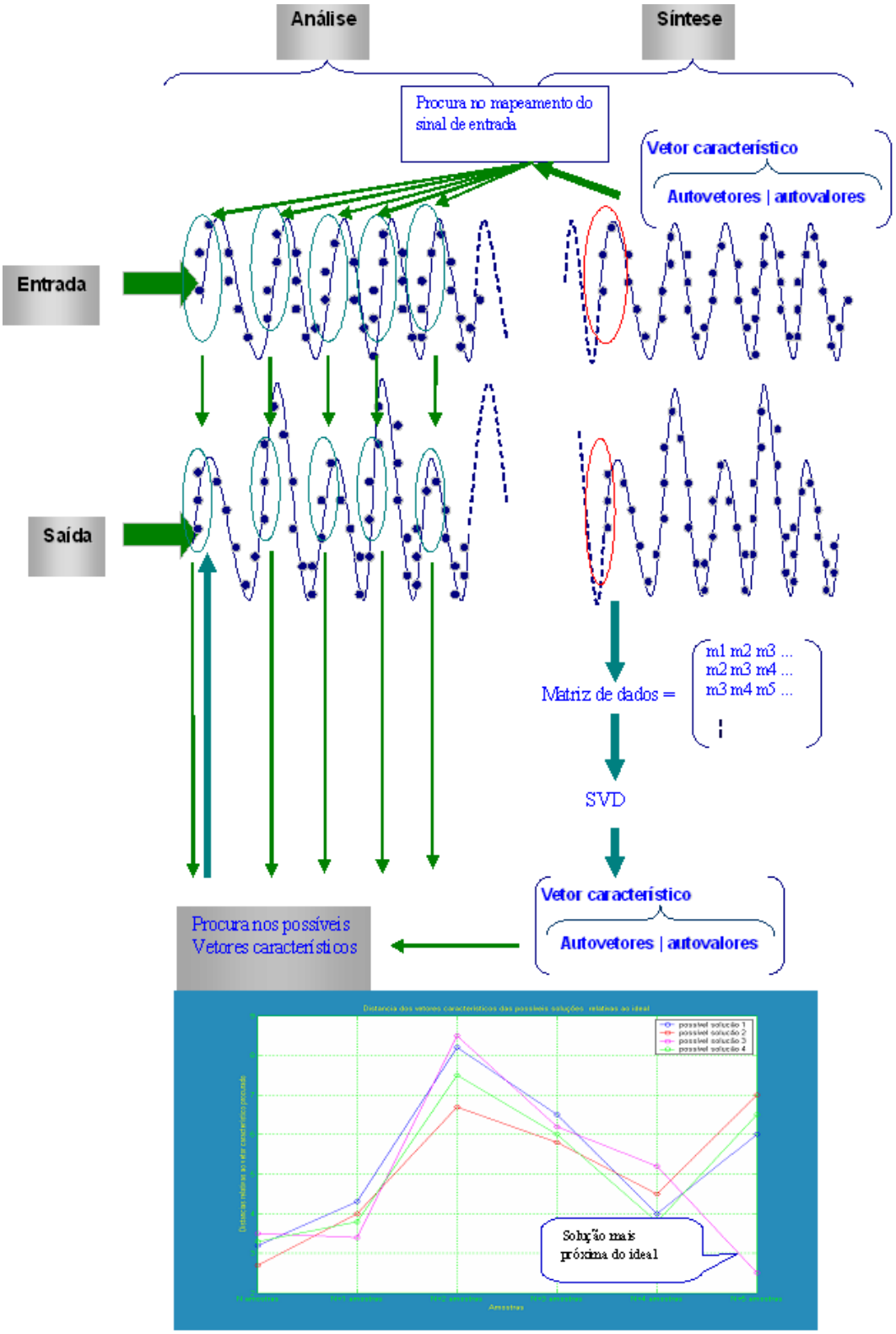

Figura 48 Processo recursivo para identificar a partição no sinal de saída

Ao final deste processo é identificada a partição mais apropriada a descrever o mapeamento entrada-saída e passa-se a estimar a amostra atual de saída respectiva. O procedimento adotado é descrito a seguir.

Tabela de Modelos

Modelo	Amostra_inicio	Amostra_final
1	1	20
2	21	40
3	41	60
4	61	70
5	1	10
6	5	10
⋮	⋮	⋮
N	1	2

Tabela 6 Tabela Código usado pelo programa para identificar os modelos das partições.

A Tabela 6 mostra o conjunto de parâmetros (resumido) usado pelo processo de identificação descrito para os modelos de partições. A primeira coluna, "Modelo", armazena os números índices atribuídos a cada modelo das diferentes técnicas usadas durante o treinamento. A segunda coluna, "Amostra_inicio", armazena a posição da amostra que inicia a partição. A coluna seguinte, "Amostra_final", armazena a posição da amostra com a qual termina a partição.

Note-se que na Tabela 6 pode-se chegar a muitos modelos que tenham a coincidência nas amostras de início e/ou fim de partições. Isto é devido a ter gerado, durante o treinamento, subpartições, cada qual dependendo de soluções provenientes

de sub-níveis dela originados. Podem existir também partições que tenham a mesma posição para amostra início e fim, indicando que a partição só têm uma amostra e que nela não podem ser feitas mais sub-partições ou interpolações.

Agora, como ilustração, vamos supor o caso onde se tenha identificado a amostra três nos dados de treinamento como a amostra início das "M-2" amostras seguintes com comportamento mais parecido às últimas "M-1" amostras de entrada e saída. Então, uma vez identificada a amostra três, vai se identificar as partições que contêm essa amostra, usando a Tabela 6. O procedimento pode ser observado na Figura 49.

Modelo	Amostra inicio	Amostra final
1	1	20
2	21	40
3	41	60
4	61	70
5	1	10
6	5	10
N	1	2

Modelos identificados

Figura 49 Identificação de modelos

O passo seguinte é sintetizar as saídas estimadas por cada uma das correspondentes técnicas, expressar estas saídas em função de seus autovetores e autovalores, e logo calcular os novos autovetores e autovalores aplicando os pesos achados na etapa da análise (Figura 50).

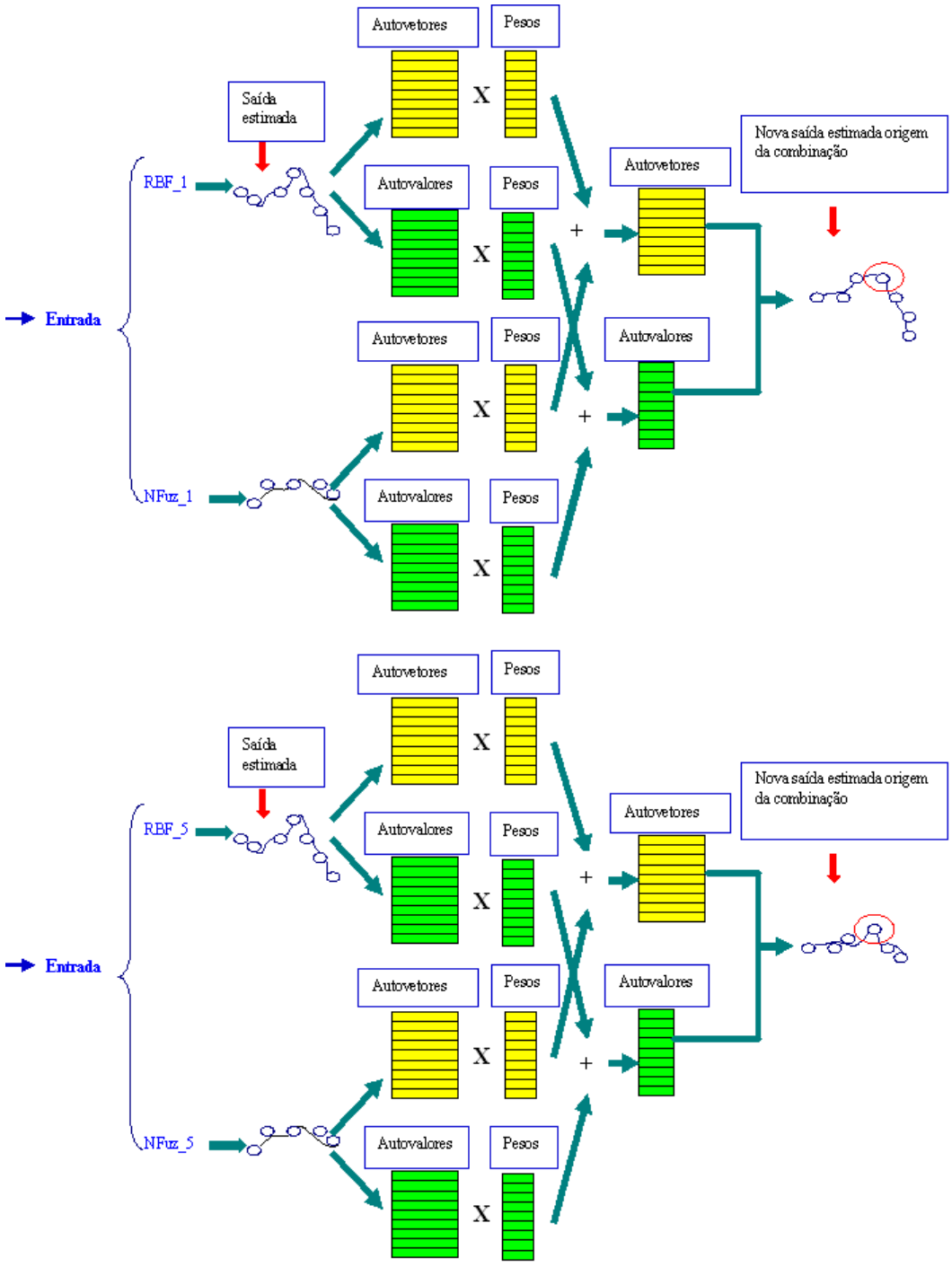

Figura 50 Recombinação de vetores característicos

A amostra estimada será resultado da soma das amostras obtidas pelos diferentes modelos da partição respectiva (Figura 51).

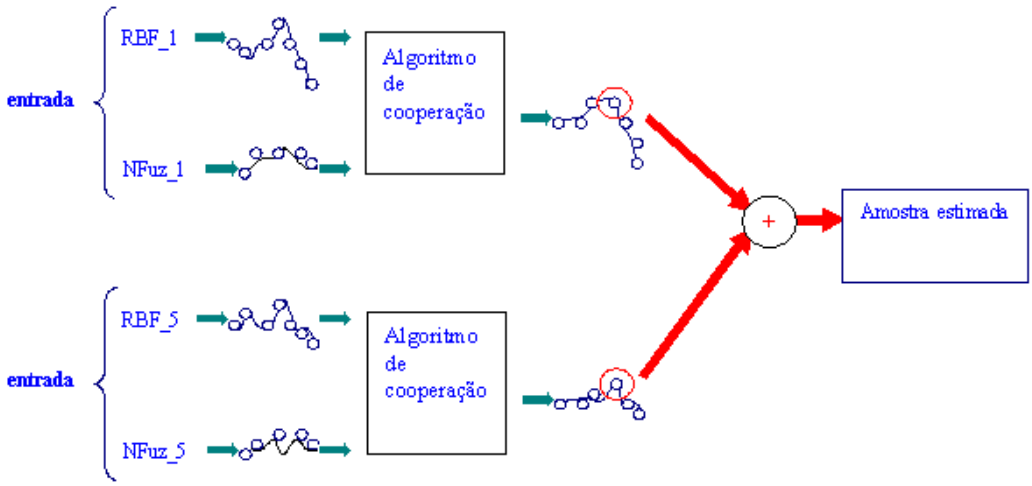

Figura 51 Amostra estimada

Pode-se observar na Figura 51 nas saídas estimadas da combinação uma amostra dentro de um círculo, que seria a posição da amostra estimada. É importante recordar que, quando foi feito o treinamento das partições pelas técnicas usadas, não necessariamente o número de amostras contidas em cada partição é o mesmo, ou seja, cada partição não necessariamente vai ter um mesmo número de vetores característicos que as outras partições. Observemos a Figura 52.

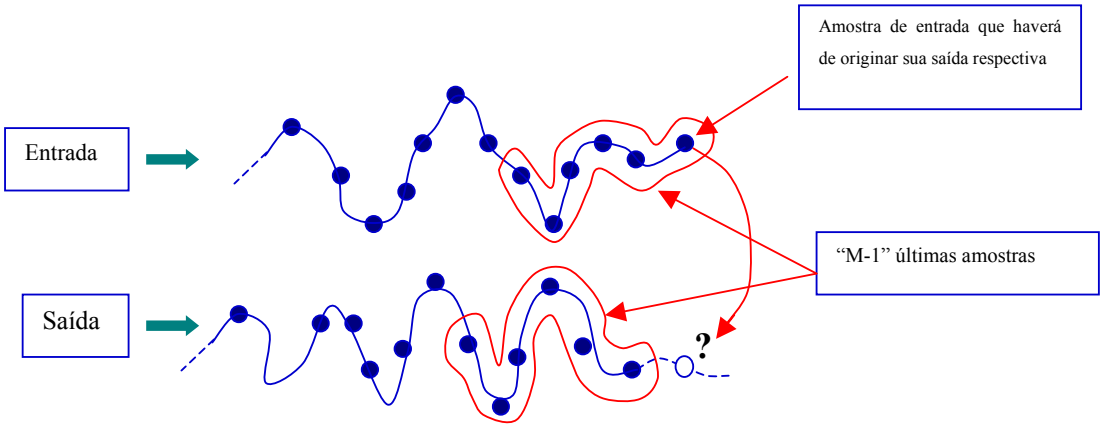

Figura 52 "M-1" últimas amostras

Com os vetores característicos representativos das “M-1” últimas amostras identificamos a partição de comportamento mais parecido, esta partição tem seus modelos respectivos, os quais foram obtidos com um número de amostras porem, um número de vetores característicos, mas as “M-1” últimas amostras podem ser poucas amostras para satisfazer o número de vetores característicos em cada modelo, pois para cada vetor característicos temos um peso para o autovetor e um outro peso para o autovalor. Para superar esta dificuldade é que interpolamos amostras dentro das “M-1” últimas amostras, criando amostras dentro das “M-1”disponíveis. A identificação da última amostra na seqüência de entrada permite selecionar a última amostra de saída, correspondendo a tal entrada (Figura 53).

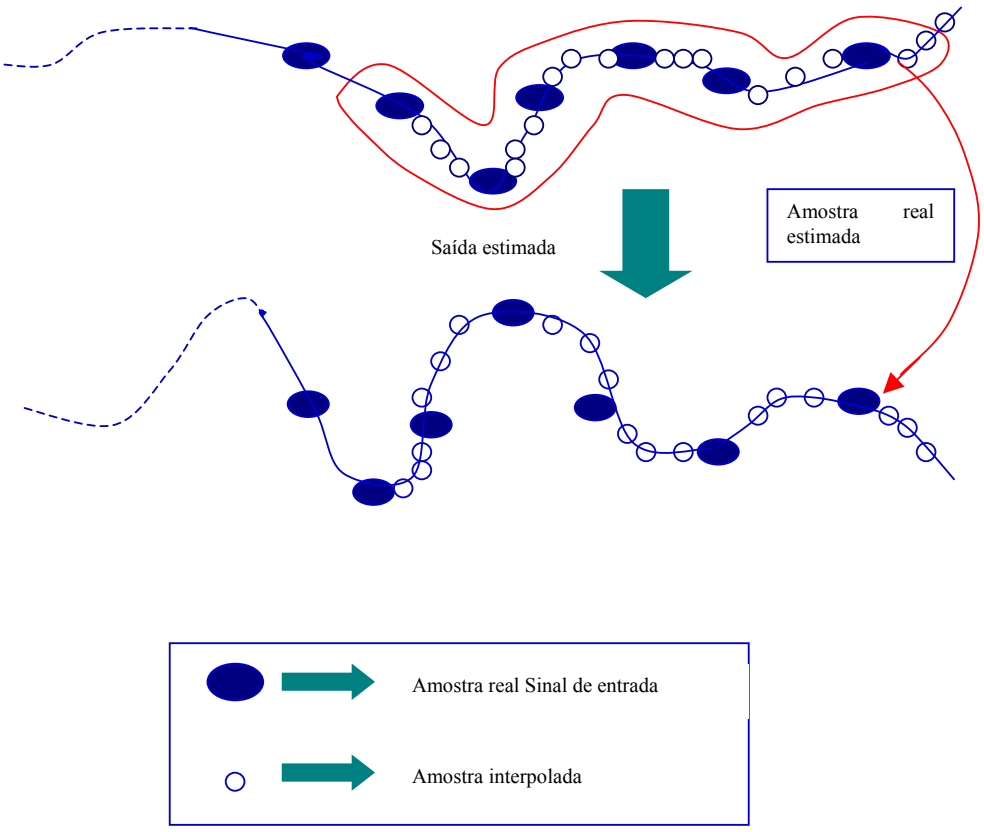

Figura 53 Amostras interpoladas

Sabendo a posição da amostra de entrada, saberemos a posição da amostra de saída correspondente.

Este procedimento de interpolação também é muito usado no processo iterativo de partições. Cada vez que é feita uma partição o número de amostras vai diminuindo, o que dificulta a utilização das técnicas de identificação. Para ter certo grau de eficiência é necessário ter um número significativo de amostras, que representem o comportamento do sinal. Além disto, a interpolação é feita com o propósito de atingir o erro de modelagem desejado antes de chegar a partições que não possam mais se sub-dividir.