Tutorial: Regressão Não Linear
(Modelo Fuzzy de Mamdani)

Prof. Dr. Guilherme de Alencar Barreto

Setembro /2021

Departmento de Engenharia de Teleinformática (DETI)
Curso de Graduação em Engenharia de Teleinformática (PPGETI)
Universidade Federal do Ceará (UFC), Campus do Pici, Fortaleza-CE

gbarreto@ufc.br

Objetivo - Apresentar uma sequência de passos e respectivos comandos do Matlab/Octave para construir um dos mais simples modelos de regressão não linear usando o modelo de inferência fuzzy de Mamdani e a validação do modelo de regressão encontrado através da análise dos resíduos e do cálculo do coeficiente de determinação R^2. A aplicação escolhida é a determinação da curva de potência de um aerogerador.

1 Construção do Model e Comandos Matlab/Octave

- **Passo 1**: Carregar as medidas de velocidade do vento e potência e plotar o gráfico de dispersão dos dados. O gráfico de dispersão está mostrado na Figura 1.

```
>> X=load('aerogerador.dat');
>> x=data(:,1); % medidas de velocidade
>> y=data(:,2); % medidas de potencia
>> [x I]=sort(x); y=y(I); % Ordena as medidas (x, y)
>> figure; plot(x,y,'b.'); grid; hold on
>> title('CURVA DE POTENCIA - AEROGERADOR')
```

- **Passo 2**: Definir o número de funções de pertinência (N_{mf}) e os pontos de suporte para funções de pertinência gaussianas.

```
>> Nmf=5; % Numero de funcoes de pertinencia
>> centers=[4 6.5 9.2 11.5 14]; % centros das funcoes de pertinencia
>> powerw= [20 134 270 480 508]; % potencias associadas
>> plot(centers,powerw,'ro','markersize',10,'linewidth',3)
```

Comentário 1: Os pontos de suporte são escolhidos manualmente por inspeção do diagrama de dispersão. Por isso, a escolha destes pontos pode variar de pessoa a pessoa. Os pontos de suporte escolhidos estão mostrados na Figura 2.

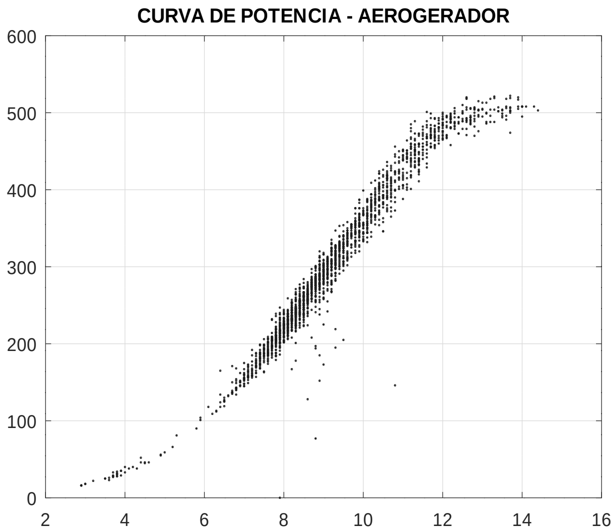

Figura 1: Gráfico de dispersão dos dados do aerogerador.

- **Passo 3**: Definir as funções de pertinência da variável de entrada (v, velocidade do vento). Para este problema em particular e pela facilidade de manuseio, escolheremos funções de pertinência gaussianas. Assim, a i-ésima função de pertinência, $i = 1, \ldots, N_{mf}$, é dada por

$$\mu_i(v) = \exp\left\{-\frac{(v - c_i)^2}{2\sigma_i^2}\right\}, \tag{1}$$

em que c_i é o centro da i-ésima função e σ_i é o raio (*radius*) ou largura (*width*) desta função. Os centros das $N_{mf} = 5$ funções de pertinência são aqueles definidos na variável `centers`, aqui representados pelo vetor $\mathbf{c} = [c_1 \ \ c_2 \ \ c_3 \ \ c_4 \ \ c_5]^T$. Os gráficos destas funções estão mostrados na Figura 3, sendo gerados pelo código abaixo.

```
>> spread=1.35; % mesmo raio para todas as funcoes
>> xx=0.8*min(x):0.1:1.2*max(x); % define grid para velocidade do vento
>> for j=1:length(xx),
       for i=1:Nmf,
          mi(i,j) = exp(-(xx(j)-centers(i))^2/(2*spread^2));
       end
   end
>> K=200;
>> plot(xx,K*mi(1,:),'r-','linewidth',5);
>> plot(xx,K*mi(2,:),'g-','linewidth',5);
>> plot(xx,K*mi(3,:),'b-','linewidth',5);
>> plot(xx,K*mi(4,:),'k-','linewidth',5);
>> plot(xx,K*mi(5,:),'m-','linewidth',5);
```

Matematicamente, as $N_{mf} = 5$ funções de pertinência mostradas na Figura 3 são dadas

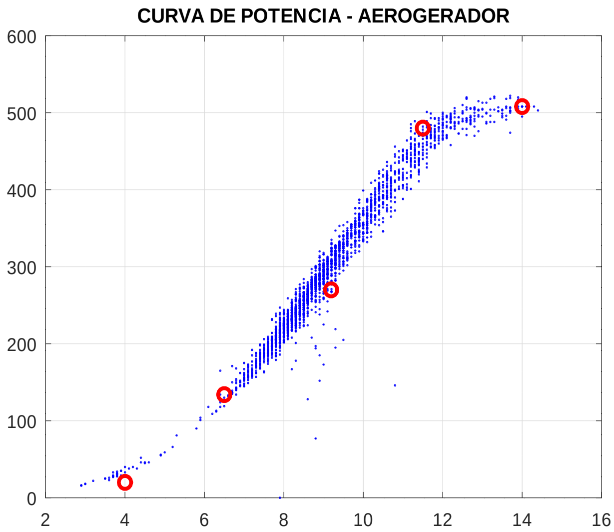

Figura 2: Gráfico de dispersão dos dados do aerogerador com os pontos de suporte destacados em vermelho.

por

$$\text{Velocidade muito baixa} : \mu_1(v) = \exp\left\{-\frac{(v-c_1)^2}{2\sigma^2}\right\} = \exp\left\{-\frac{(v-4)^2}{2(1,35)^2}\right\} \quad (2)$$

$$\text{Velocidade baixa} : \mu_2(v) = \exp\left\{-\frac{(v-c_2)^2}{2\sigma^2}\right\} = \exp\left\{-\frac{(v-6,5)^2}{2(1,35)^2}\right\} \quad (3)$$

$$\text{Velocidade média} : \mu_3(v) = \exp\left\{-\frac{(v-c_3)^2}{2\sigma^2}\right\} = \exp\left\{-\frac{(v-9,2)^2}{2(1,35)^2}\right\} \quad (4)$$

$$\text{Velocidade alta} : \mu_4(v) = \exp\left\{-\frac{(v-c_4)^2}{2\sigma^2}\right\} = \exp\left\{-\frac{(v-11,5)^2}{2(1,35)^2}\right\} \quad (5)$$

$$\text{Velocidade muito alta} : \mu_5(v) = \exp\left\{-\frac{(v-c_5)^2}{2\sigma^2}\right\} = \exp\left\{-\frac{(v-14)^2}{2(1,35)^2}\right\} \quad (6)$$

- **Passo 4**: A variável de saída (i.e., potência gerada) deve ser fuzzificada também. Para simplificar, vamos usar funções de pertinência do tipo *singleton*. Os pontos escolhidos para a potência são aqueles definidos na variável `powerw`, aqui representados pelo vetor $\mathbf{p}^{out} = [p_1^{out} \;\; p_2^{out} \;\; p_3^{out} \;\; p_4^{out} \;\; p_5^{out}]^T$. Os gráficos destas funções estão mostrados na Figura 4, sendo gerados pelo código abaixo.

```
>> figure;
>> stem(powerw,ones(1,Nmf),'k-','filled','linewidth',3,'markersize',10) ;
>> grid; title('PERTINENCIAS SINGLETON (SAIDA)');
```

- **Passo 5**: Como só temos uma variável de entrada (velocidade do vento), o número de regras fuzzy será igual ao número de funções de pertinência. Os centros (c_i) são usados pelas funções de pertinência que compõem os antecedentes das regras, enquanto as potências p_i^{out}

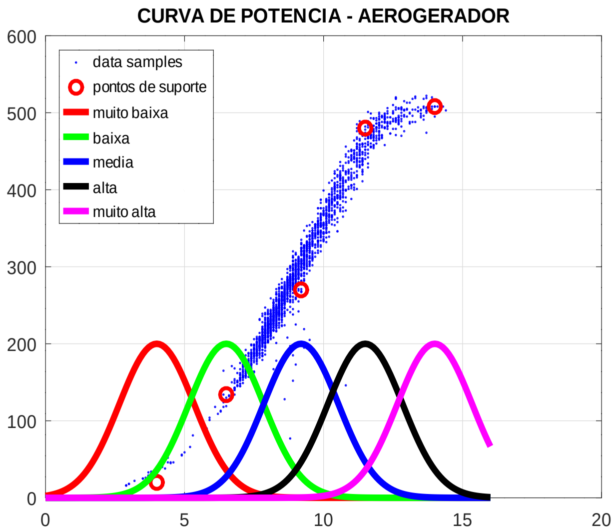

Figura 3: Gráfico de dispersão com $N_{mf} = 5$ funções de pertinência gaussianas de mesmo raio.

são usadas nos consequentes das regras. As $N_{mf} = 5$ regras fuzzy usadas neste tutorial são listadas abaixo.

$$\textbf{Regra 1:} \quad \text{Se } (v \text{ é Muito Baixa}), \text{ Então } y_1 = p_1^{out}. \tag{7}$$
$$\textbf{Regra 2:} \quad \text{Se } (v \text{ é Baixa}), \text{ Então } y_2 = p_2^{out}. \tag{8}$$
$$\textbf{Regra 3:} \quad \text{Se } (v \text{ é Média}), \text{ Então } y_3 = p_3^{out}. \tag{9}$$
$$\textbf{Regra 4:} \quad \text{Se } (v \text{ é Alta}), \text{ Então } y_4 = p_4^{out}. \tag{10}$$
$$\textbf{Regra 5:} \quad \text{Se } (v \text{ é Muito Alta}), \text{ Então } y_5 = p_5^{out}. \tag{11}$$

- **Passo 6**: As ativações m_i das regras mostradas no Passo 5 são dadas pelos próprios valores de pertinência da velocidade v medida, conforme mostrado no Passo 3; ou seja, $m_i = \mu_i(v)$. Assim, a saída predita $\hat{y}(v)$ do modelo fuzzy de Mamdani é a potência predita para a velocidade v medida, sendo calculada como

$$\hat{y}(v) = \hat{p}(v), \tag{12}$$
$$= \frac{\sum_{i=1}^{N_{mf}} m_i y_i}{\sum_{i=1}^{N_{mf}} m_i}, \tag{13}$$
$$= \frac{\sum_{i=1}^{N_{mf}} \mu_i(v) p_i^{out}}{\sum_{i=1}^{N_{mf}} \mu_i(v)}. \tag{14}$$

Em palavras, a potência predita pelo modelo, para uma dada velocidade v, é a média ponderada das potências de saídas (p_i^{out}) das N_{mf} regras, com os pesos dados pelas ativações das regras ($\mu_i(v)$).

- **Passo 7**: Para desenhar a curva de potência, faz-se uso do grid (`xx`) definido no Passo 3. O código abaixo mostra como é o procedimento.

```
>> clear mi;
```

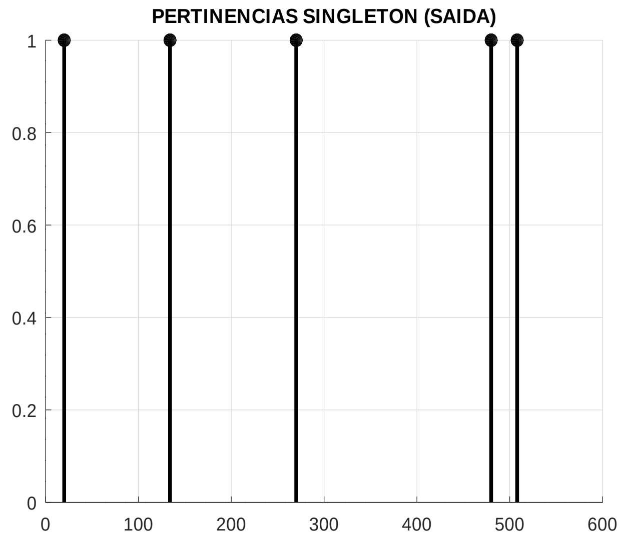

Figura 4: Funções de pertinência *singleton* da variável de saída (potência gerada).

```
>> for j=1:length(xx),
       for i=1:Nmf,
          mi(i,j) = exp(-(xx(j)-centers(i))^2/(2*spread^2));
          ypred(i) = powerw(i); % Saida predita pela i-esima regra
       end
       yhat(j) = sum(mi.*ypred)/sum(mi); % saida predita final
   end
>> plot(xx, yhat, 'k-', 'linewidth', 4);
>> xlabel('Wind speed [m/s]');
>> ylabel('Power output [KWatts]'); hold off
```

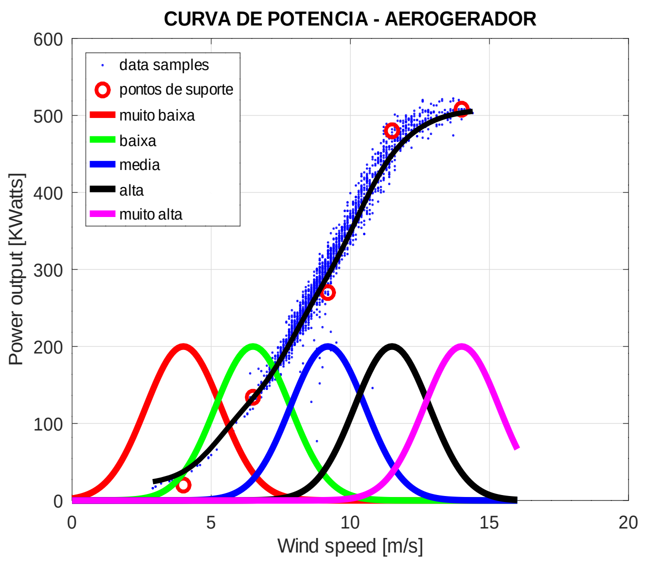

Figura 5: Curva de potência gerada pelo método de Mamdani.

- **Passo 8**: Para calcular o índice R^2, usamos as medidas da velocidade do vento e da potência nas variáveis `x` e `y`, respectivamente. O código abaixo mostra como é o procedimento.

```
>> clear mi;
>> Pmedia=mean(y); % valor medio da potencia
>> for j=1:length(x),
       for i=1:Nmf,
          mi(i,j) = exp(-(x(j)-centers(i))^2/(2*spread^2));
          ypred(i) = powerw(i); % Saida predita pela i-esima regra
       end
       yhat(j) = sum(mi.*ypred)/sum(mi); % saida predita final
       erronum(j) = y(j)-yhat(j); % erro de predicao modelo Mamdani
       erroden(j) = y(j)-Pmedia; % erro do modelo de predicao pela media
    end
>> NUM=sum(erronum.^2); % Numerador
>> DEN=sum(erroden.^2); % Denominador
>> R2 = 1 - NUM/DEN, % Indice R2
R2 = 0.96759
```