



UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS SOBRAL

Curso: Engenharia da Computação
Disciplina: Inteligência Computacional
Prof. Jarbas Joaci de Mesquita Sá Junior
1º Trabalho

Aluno: Francisco Jeferson da Silveira Pontes

Matrícula: 397888

As questões foram elaboradas na linguagem de programação python, versão 3.6, e com o auxílio da IDE PyCharm 2019.2.2, e está disponível em : <https://github.com/Jefersonpontes13/Trabalho-01-IC>.

Se houver problemas ao executar, a pasta do projeto está no repositório, e contém o ambiente virtual do python3.6, que é adequado para execução.

1. Encontre o máximo da função $f(x,y) = x\sin(y\pi/4) + y\sin(x\pi/4)$ por meio do algoritmo hill-climbing. As variáveis x e y pertencem ao intervalo entre 0 e 20. Os vizinhos de determinado estado (x, y) são $(x, y \pm 0,01)$, $(x \pm 0,01, y)$ e $(x \pm 0,01, y \pm 0,01)$. Por exemplo, os vizinhos do estado $(1,00; 1,00)$ são $(1,00; 1,01)$, $(1,01; 1,01)$, $(0,99; 0,99)$, $(0,99; 1,00)$ etc.

O algoritmo gera uma posição dentro do intervalo $X = [0, 20]$, e $Y = [0, 20]$, e partindo da premissa do algoritmo Hillclimbing Original (maior passo), move-se para o estado vizinho que proporciona a maior melhoria, aumento no valor da função objetivo, no caso de $f(x, y)$, para no máximo local mais próximo.

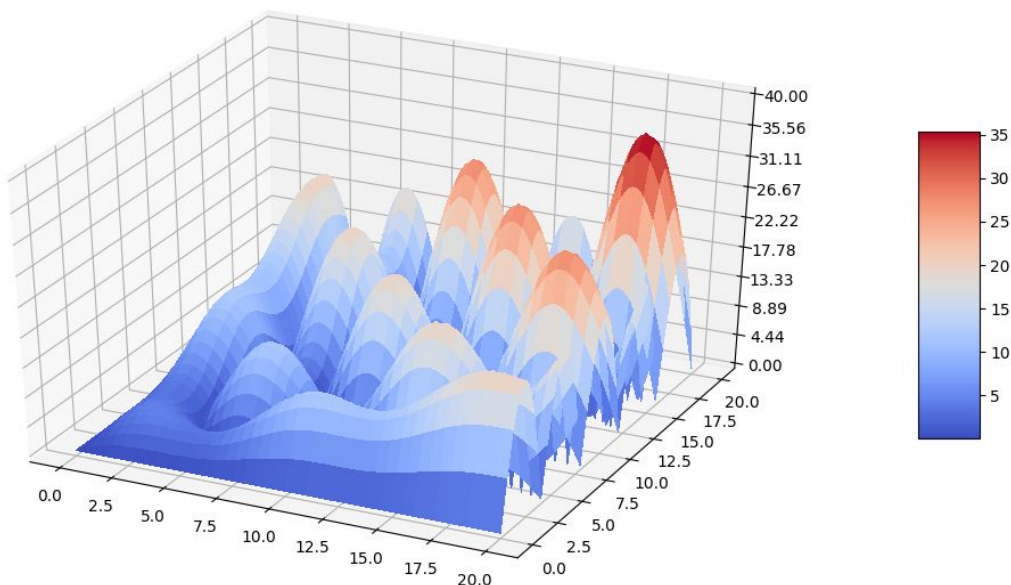


Figura - 01 : $f(x, y) = x\sin(y\pi/4) + y\sin(x\pi/4)$

Plotando o gráfico, é perceptível que a melhor solução do problema está entre 15 e 20 nos eixos X e Y, e $f(x, y)$ por volta de 35. A resolução está no arquivo Hillclimbing.py, e além dele, o arquivo Log-Hillclimbing.txt auxilia na compreensão da resolução do problema.

Executando o script Hillclimbing.py: O algoritmo implementa o Random-Restart: Múltiplas buscas hill-climbing a partir de diferentes estados iniciais gerados aleatoriamente. Foi definido 10 tentativas, e os resultados são gravados no arquivo Log-Hillclimbing.txt, logo, é possível identificar se houve êxito na busca do máximo global, que no problema é 36,088, nas coordenadas $X = 18,09$, e $Y = 18,09$.

```
[19, 17, 28.45584412271573]
Interação: 274
[18,9, 17,1, 27.374614761601144]
Interação: 275
[18.799999999999997, 17.200000000000003, 29.124611797498144]
Interação: 276
[18.699999999999996, 17.300000000000004, 30.695045916747397]
Interação: 277
[18.599999999999994, 17.400000000000006, 32.07623487078131]
Interação: 278
[18.499999999999993, 17.500000000000007, 33.25966317040639]
Interação: 279
[18.399999999999999, 17.600000000000001, 34.2380345866256]
Interação: 280
[18.299999999999999, 17.700000000000001, 35.005317134316414]
Interação: 281
[18.199999999999999, 17.800000000000001, 35.55678026142502]
Interação: 282
[18.099999999999987, 17.900000000000013, 35.88902401439263]
Interação: 283
[18.099999999999987, 18.000000000000014, 36.04451200719632]
Interação: 284
[18.099999999999987, 18.100000000000016, 36.08840748113923]
Fim Tentativa 10
jeferson@jeferson:~/PycharmProjects/Trabalho01-IC$
```

Fig - 2 : Terminal

```
log-Hillclimbing.txt
1
2 Tentativa: 1 X: 18.200000000000024 | Y: 10.099999999999996 | R: 28.119547713953814
3
4 Tentativa: 2 X: 14.300000000000002 | Y: 6.099999999999925 | R: 20.18737438680956
5
6 Tentativa: 3 X: 10.200000000000001 | Y: 10.2 | R: 20.148842148140808
7
8 Tentativa: 4 X: 2.1000000000000005 | Y: 10.8 | R: 12.465642892565173
9
10 Tentativa: 5 X: 18.800000000000033 | Y: 2.099999999999999 | R: 20.440981562370197
11
12 Tentativa: 6 X: 18.200000000000024 | Y: 10.099999999999996 | R: 28.119547713953814
13
14 Tentativa: 7 X: 6.100000000000003 | Y: 14.299999999999992 | R: 20.18737438680956
15
16 Tentativa: 8 X: 18.800000000000004 | Y: 2.099999999999983 | R: 20.440981562370197
17
18 Tentativa: 9 X: 18.100000000000016 | Y: 18.1 | R: 36.08840748113923
19
20 Tentativa: 10 X: 18.099999999999987 | Y: 18.100000000000016 | R: 36.08840748113923
21
```

Fig - 3 : Log-Hillclimbing.txt

Fig - 2: Mostra as interações e os valores obtidos.

Fig - 3: Mostra os resultados obtidos em cada tentativa.

2. Construa um programa baseado em lógica fuzzy (inferência de Mamdani) que receba três valores: pressão no pedal, velocidade da roda e velocidade do carro e que devolva a pressão no freio. Siga as regras disponibilizadas nos slides sobre Lógica Fuzzy.

Com base nas informações dos slides sobre Lógica fuzzy, a solução foi modelada no arquivo Fuzzy.py. Ao executar, é preciso definir os valores de entrada no terminal do python, e logo após, imprime os gráficos das funções de pertinência, e o resultado final da desnebulização e o gráfico de corte das funções de pertinência das variáveis (Apertar e Liberar freio) nos valores obtidos no processo de nebulização.

```
jeferson@jeferson:~/PycharmProjects/Trabalho01-IC$ python3 Fuzzy.py
Entre com os valores de pressão no pedal, velocidade do carro e velocidade das r
odas

Pressão no pedal: 60
Velocidade do carro: 80
Velocidade das rodas: 55
```

Fig - 4 : Inserção dos valores de entrada.

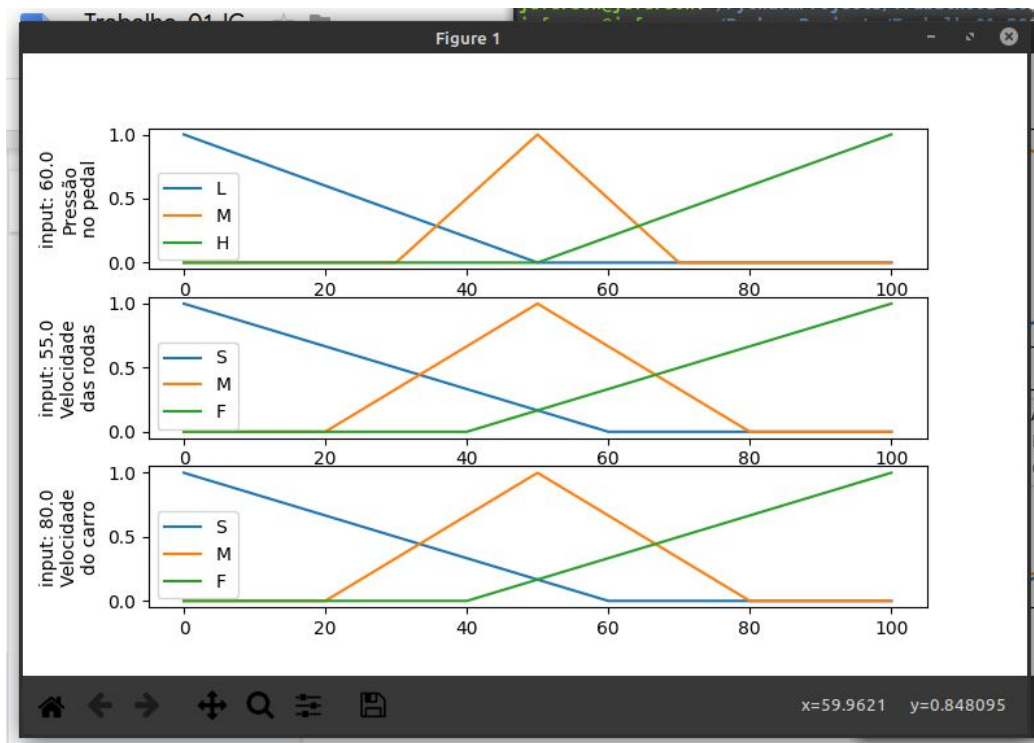


Fig - 5 : Gráfico das funções de pertinência.

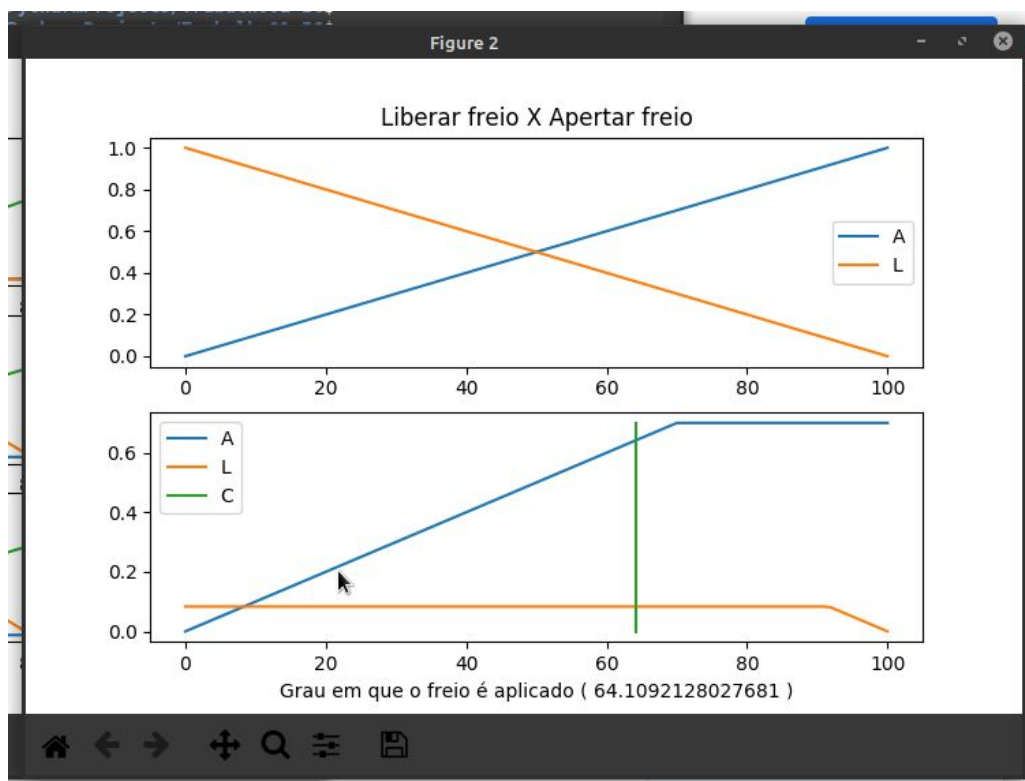


Fig - 6 : Resultado final da desnebulização.

3. Usando o conjunto de dados do aerogerador (variável de entrada: velocidade do vento – m/s, variável de saída: potência gerada – kWatts), determine os modelos de regressão polinomial (graus 2, 3, 4 e 5) com parâmetros estimados pelo método dos mínimos quadrados. Avalie a qualidade de cada modelo pela métrica R^2 e R^2_{aj} (equações 48 e 49, slides sobre Regressão Múltipla).

A solução está no arquivo `Regressao.py`, ao executar o algoritmo, deve-se inserir o grau do polinômio, e automaticamente será gerado o gráfico da distribuição dos dados e a função de grau pré-definido.

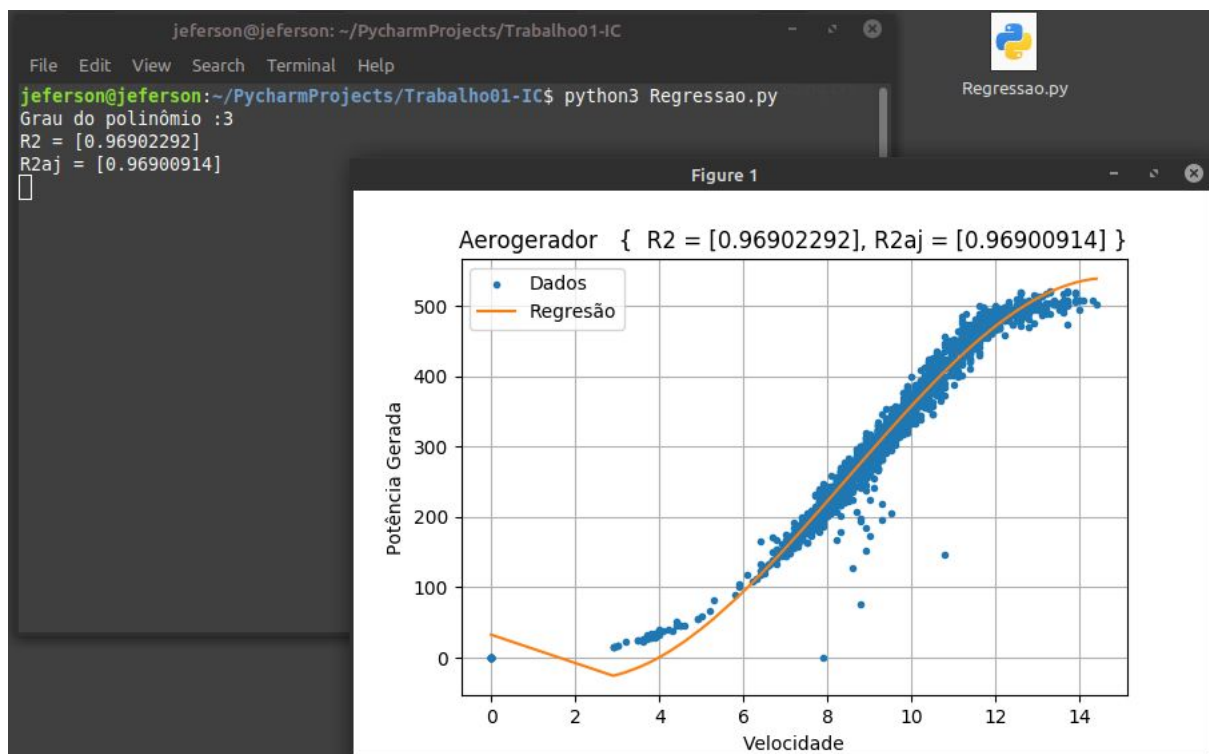


Fig - 7 : Execução do algoritmo.

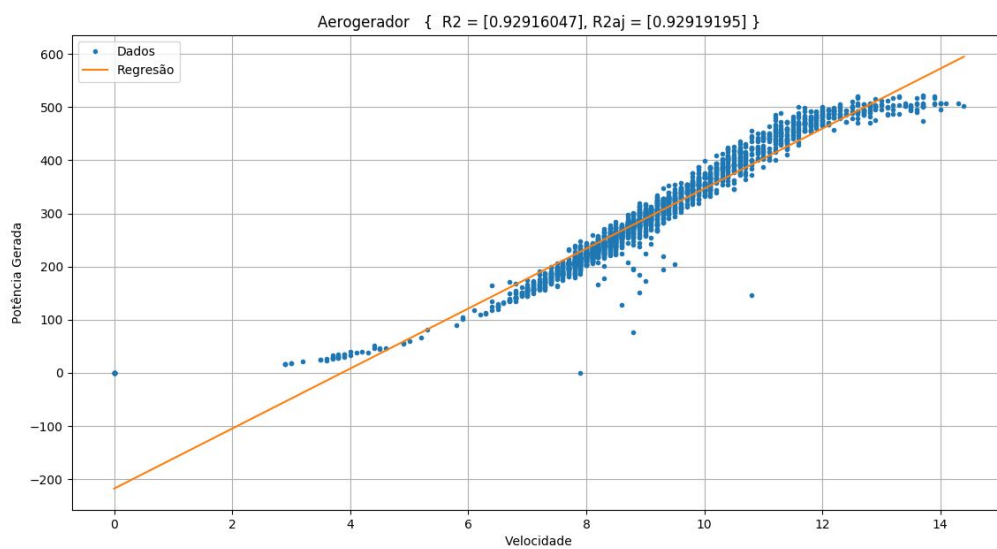


Fig - 8 : Regressão linear.

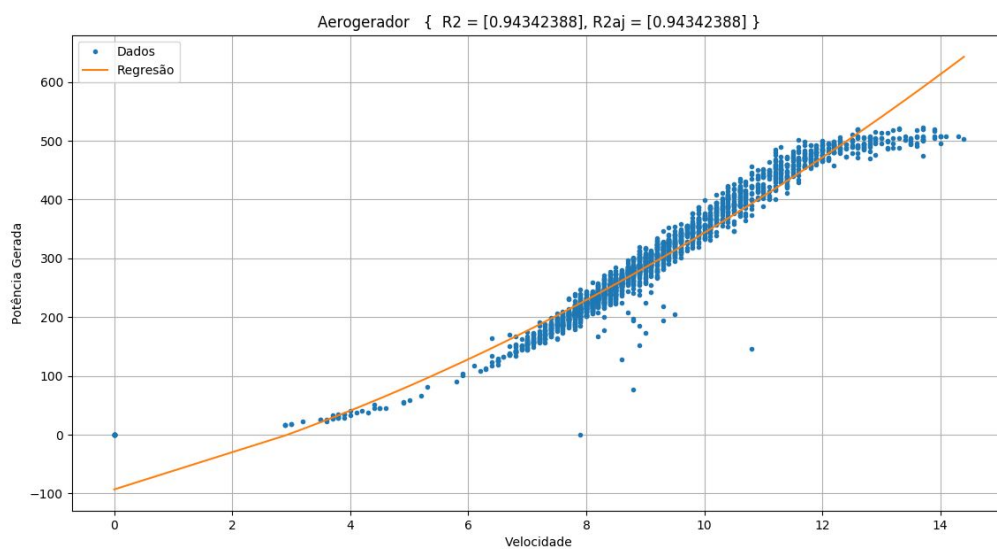


Fig - 9 : Regressão de grau 2.

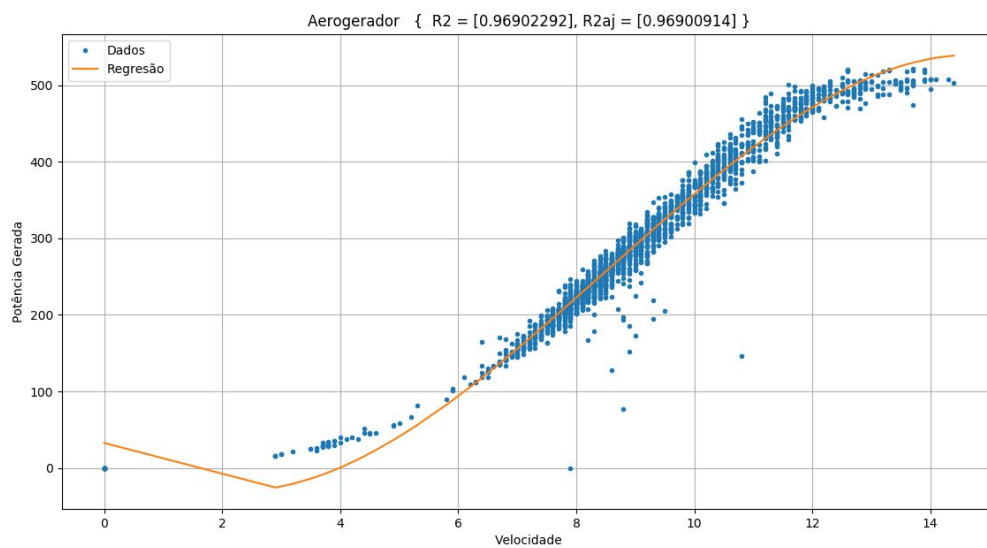


Fig - 10 : Regressão de grau 3.

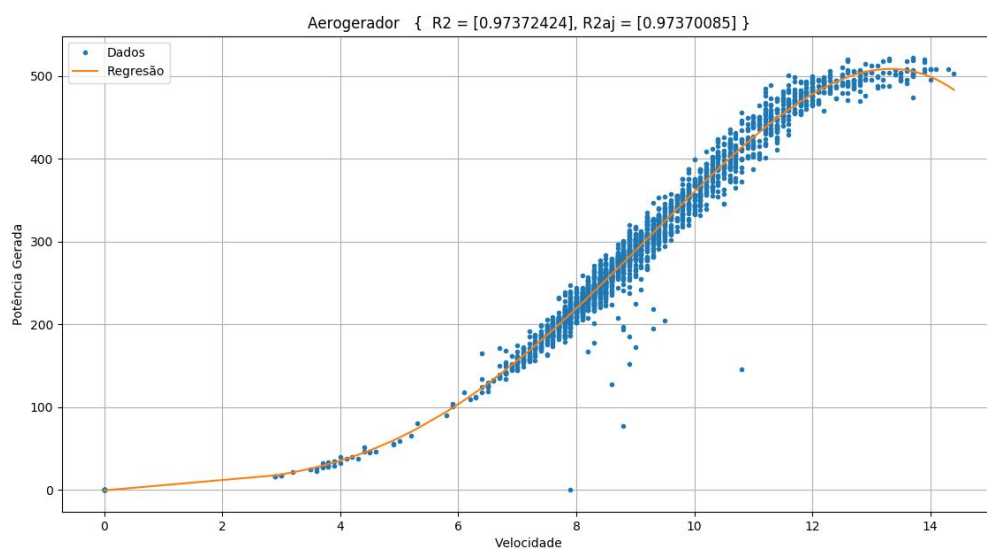


Fig - 11 : Regressão de grau 4.

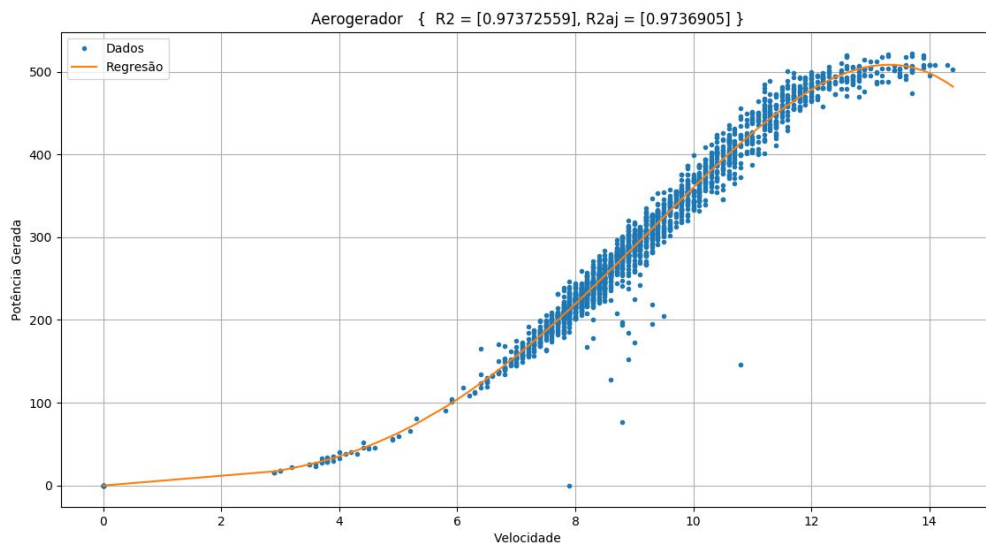


Fig - 12 : Regressão de grau 5.

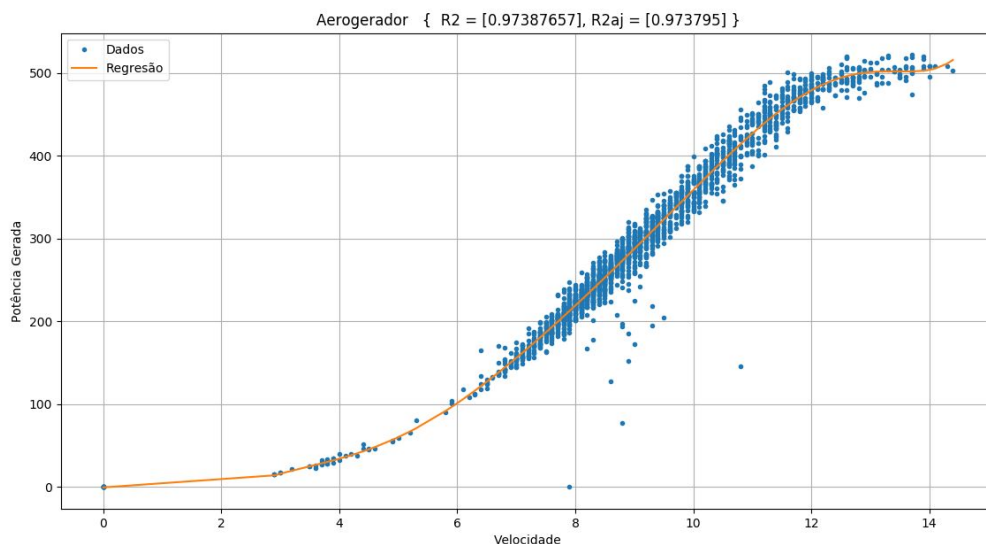


Fig - 13 : Regressão de grau 9.

Foram feitas regressões de grau 1 a 5, e uma especial de grau 9, a partir dos gráficos e de R^2 e R^2_{aj} , pode-se concluir que houve uma maior correspondência do comportamento do gráfico da regressão com o comportamento dos dados, conforme aumenta o grau do polinômio, mas nitidamente o ganho é reduzido drasticamente a partir da quarta regressão, e o polinômio de grau 9, obteve um ganho quase que desprezível, comparado com a de grau 5.

Além do mais, Analisando os resultados de R^2 , e R^2_{aj} , percebe-se que até grau 4 os valores são similares, mas a partir do grau 5, R^2_{aj} aponta uma diminuição no desempenho da regressão múltipla, embora R^2 insista em mostrar um ganho. Logo, é mais interessante usar R^2_{aj} como parâmetro de análise de desempenho de uma regressão polinomial, pois de maneira subjetiva aponta que o ganho não vale o custo computacional e de espaço, para um polinômio de grau muito elevado.