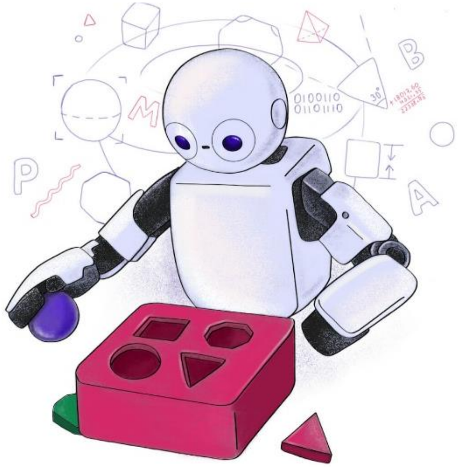
T319 - Introdução ao Aprendizado de Máquina: *Regressão Linear (Parte III)*



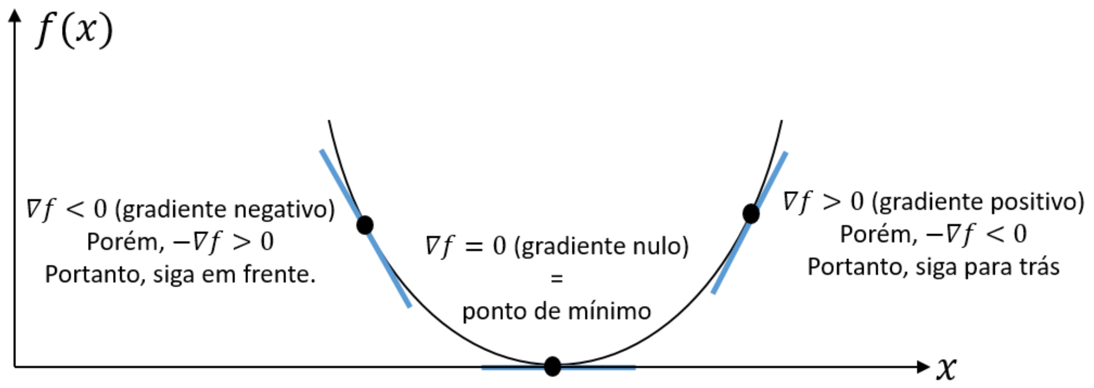
Inatel

Felipe Augusto Pereira de Figueiredo
felipe.figueiredo@inatel.br

Recapitulando

- No tópico anterior, discutimos o vetor gradiente.
- Aprendemos dois algoritmos que usam o vetor gradiente para a resolução de problemas de otimização.
  - ***Gradiente ascendente*** para problemas de ***maximização***.
  - ***Gradiente descendente*** para problemas de ***minimização***.
- Falamos sobre as três versões do gradiente descendente e as comparamos:
  - Batelada
  - Estocástico
  - Mini-batch
- Neste tópico, discutiremos o quão importante é o ajuste do passo de aprendizagem, α.

Escolha do passo de aprendizagem



- Conforme nós vimos, no gradiente descendente, o ***negativo*** do ***vetor gradiente***, $-\nabla f(x)$, dá a ***direção de decrescimento mais rápido de uma função*** a partir de um ponto e sua ***magnitude*** indica a ***taxa de variação da função*** nessa direção.
- Porém, ele não nos informa a ***distância*** até o ponto de máximo.

Escolha do passo de aprendizagem

$$\boldsymbol{a} \leftarrow \boldsymbol{a} - \alpha \frac{\partial J_e(\boldsymbol{a})}{\partial \boldsymbol{a}}$$

- Portanto, para ***andarmos na direção apontada pelo gradiente***, usamos uma ***porcentagem*** de seu valor.
- Essa porcentagem é dada pelo ***passo de aprendizagem***, α.
- O passo de aprendizagem ***controla o quão "grande" ou "pequena" é a atualização aplicada aos pesos*** do modelo em cada iteração do processo de treinamento.
- Ou seja, ele determina o ***tamanho do passo dado na direção oposta à indicada pelo vetor gradiente***.

Portanto, como veremos, a escolha do passo de aprendizagem é muito importante para o aprendizado de um modelo de ML.

Escolha do passo de aprendizagem

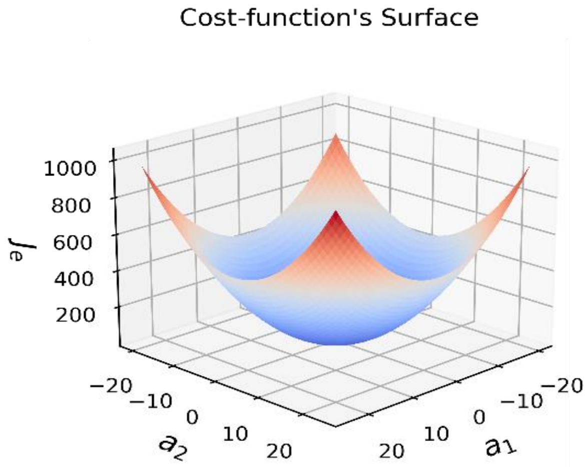
- O passo de aprendizagem é um ***hiperparâmetro*** que influencia diretamente o ***desempenho e a convergência*** do algoritmo do gradiente descendente.
  - **Hiperparâmetros**: são ***parâmetros que não são aprendidos durante o treinamento*** do modelo, mas que influenciam seu aprendizado.
- Valores ***muito pequenos*** podem resultar em ***treinamento lento***, enquanto valores ***muito grandes*** podem causar ***divergência***.
- Em geral, a escolha do passo é feita empiricamente por meio de experimentação.
- Uma regra empírica para ***exploração*** do passo de aprendizagem é usar a seguinte sequência (***ajuste manual***):

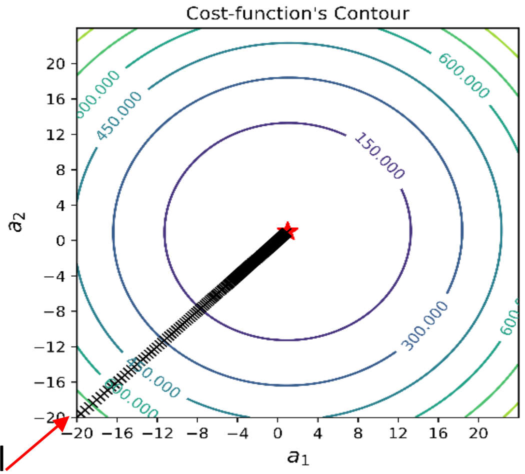
$$\ldots, 0.001 \xrightarrow{3\times} 0.003 \xrightarrow{\approx 3\times} 0.01 \xrightarrow{3\times} 0.03 \xrightarrow{\approx 3\times} 0.1 \xrightarrow{3\times} 0.3 \xrightarrow{\approx 3\times} 1.0, \ldots$$

Exemplo: selecionando_o_passo_de_aprendizagem.ipynb

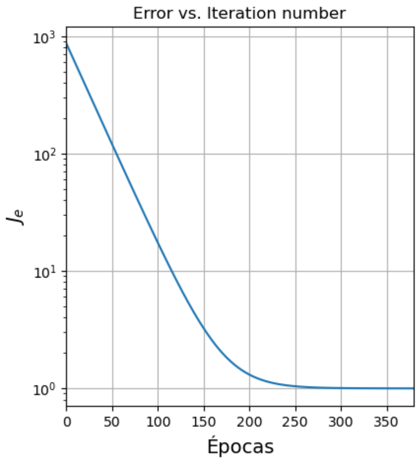
Passo de aprendizado pequeno

- Caso o passo de aprendizagem seja ***muito pequeno***, a ***convergência do algoritmo será muito lenta***.
- No exemplo abaixo, com $\alpha = 2 \times 10^{-6}$, o algoritmo atinge o ponto de mínimo, i.e., converge, após mais de 250 épocas.
  - Passos muito curtos, fazem com que o algoritmo caminhe vagarosamente em direção ao ***mínimo global*** da ***função de erro***.



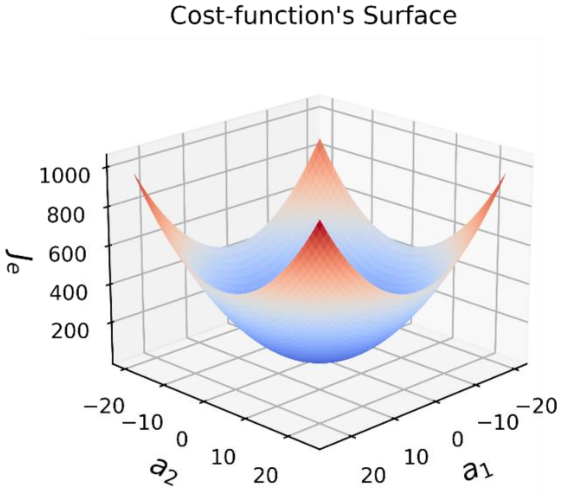


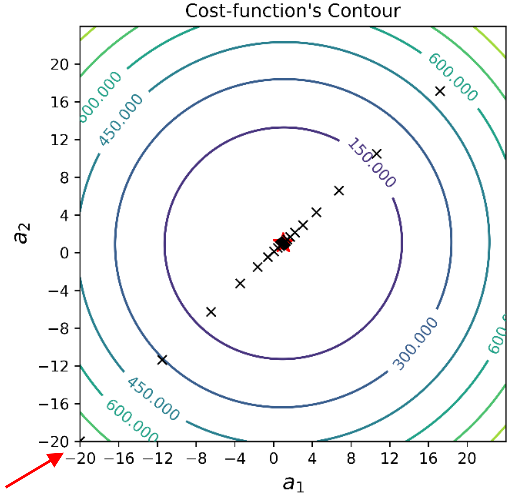
a inicial



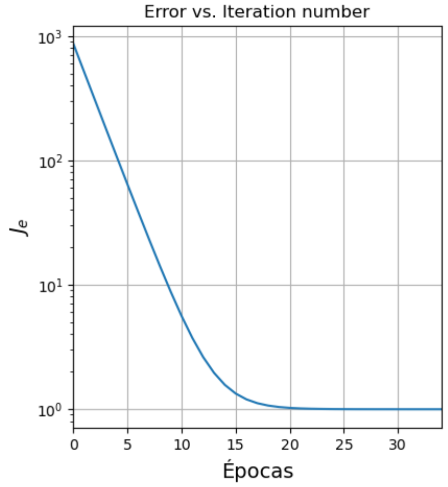
Passo de aprendizado grande

- Caso o passo seja grande, o algoritmo pode nunca convergir.
- Se o passo for grande, ***mas não tão grande assim***, o algoritmo pode ficar "***pulando***" ou "***oscilando***" ***de um lado para o outro da superfície de erro*** até que, por sorte, ele converge.
  - No exemplo abaixo, com $\alpha = 1.8 \times 10^{-4}$, o algoritmo oscila inicialmente, mas acaba convergindo após 20 épocas.



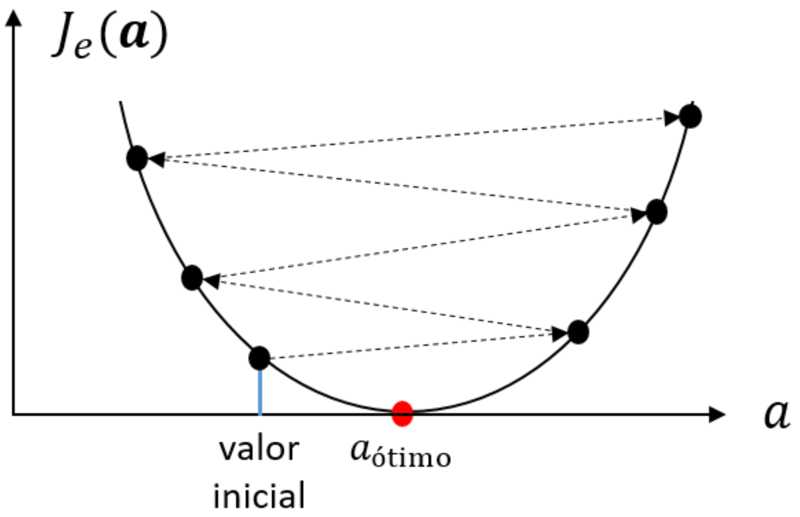


$\boldsymbol{a}$ inicial



Passo de aprendizado grande

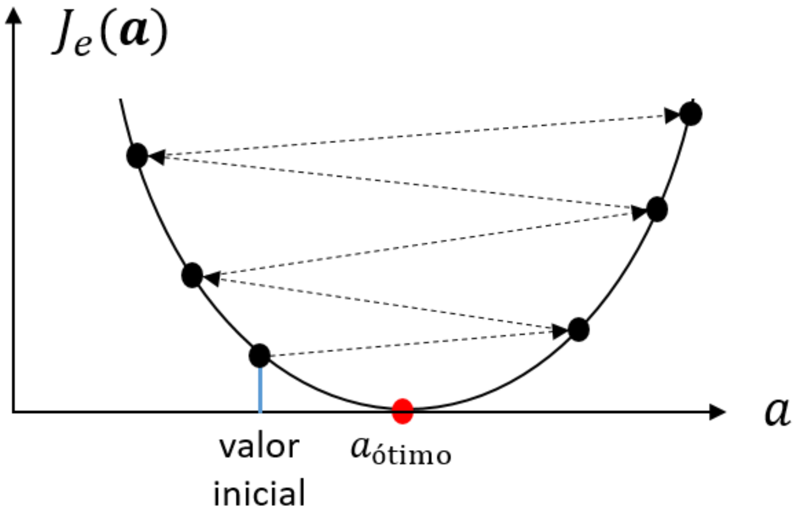
feedback positivo → estouro da precisão numérica



- Em outros casos, quando o passo é ***muito grande***, a cada época, o algoritmo "pula" para um valor mais alto do que o anterior e, assim, acaba divergindo.
- Ou seja, ao invés de se aproximar do ponto de mínimo a cada época, ele se distancia dele.

Passo de aprendizado grande

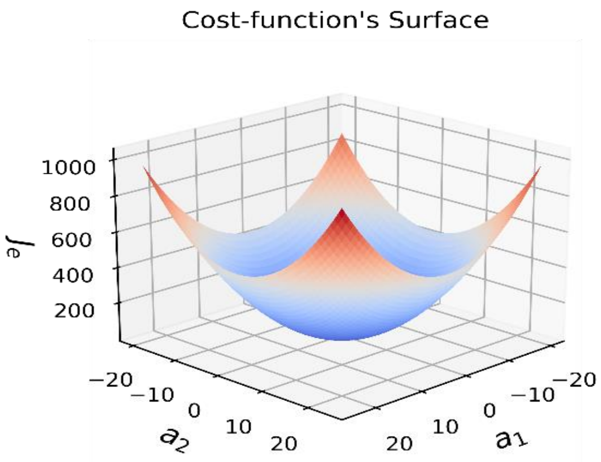
feedback positivo → estouro da precisão numérica

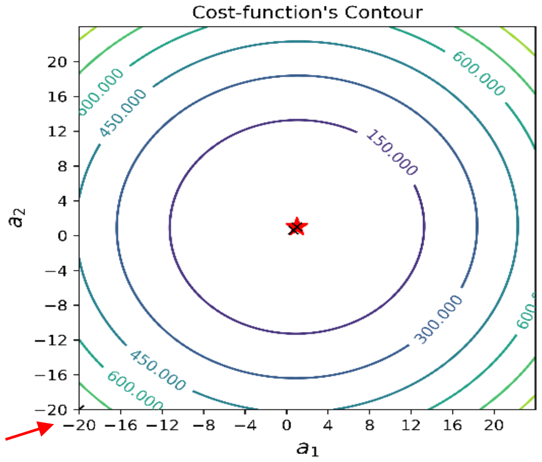


- Nesse caso ocorre um ***ciclo de feedback positivo*** onde ***a cada época*** os valores dos ***gradientes*** e, consequentemente, dos ***pesos se tornam maiores e maiores*** até que ocorra o ***estouro da representação numérica***.
  - Problema que ocorre quando uma variável não pode mais representar um valor, pois ele é maior do que o intervalo que ela pode armazenar.

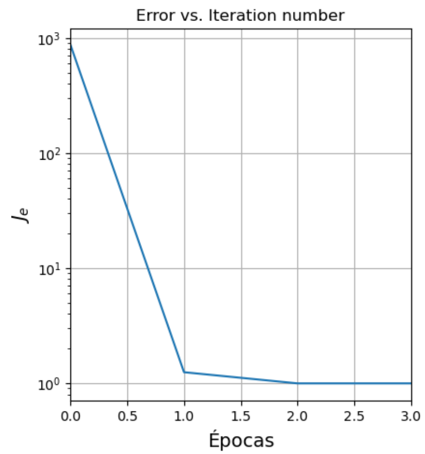
Passo de aprendizado ideal

- Portanto, o valor do passo de aprendizagem deve ser ***explorado*** para se encontrar um ***valor ideal*** que ***acelere a convergência*** de forma ***estável***, ou seja, sem oscilações.
- O exemplo abaixo, com $\alpha = 10^{-4}$, o algoritmo converge de forma estável para o ***mínimo global*** em apenas 3 épocas.

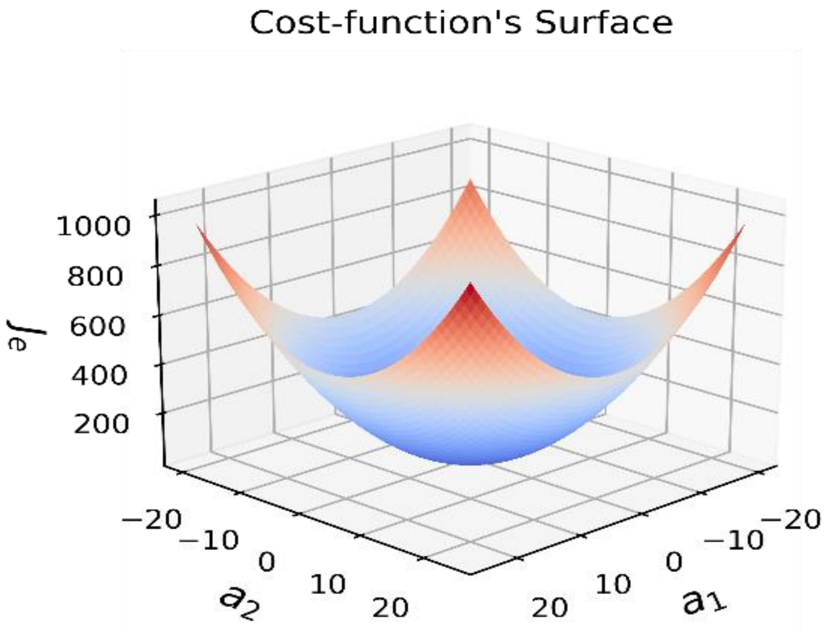




$\boldsymbol{a}$
inicial

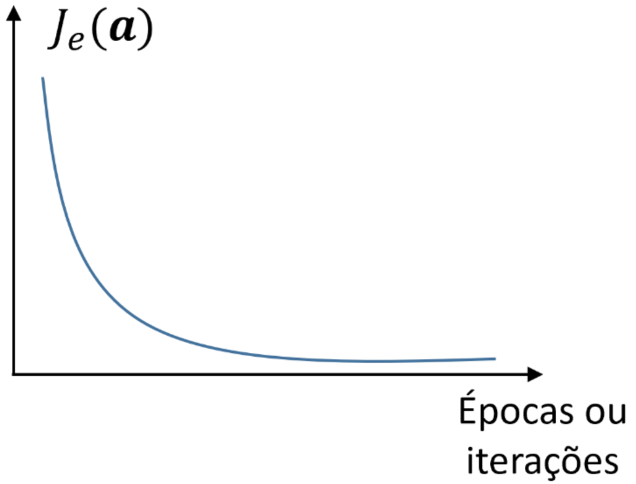


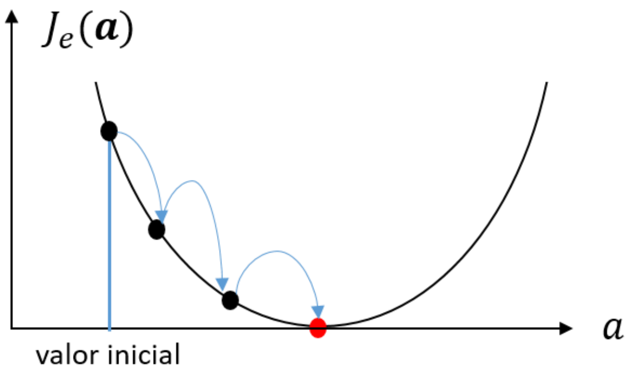
Analisando o treinamento de um modelo



- ***Nem sempre iremos conseguir plotar a superfície de erro e de contorno*** para analisarmos o treinamento e o desempenho de um modelo.
- Por exemplo, quando tivermos ***três atributos***, a ***superfície de erro terá quatro dimensões***, tornando sua análise mais difícil.
- Assim, em geral, usamos a ***curva do erro (i.e., EQM) em função das épocas*** (ou iterações) de treinamento para analisar o treinamento de um modelo.

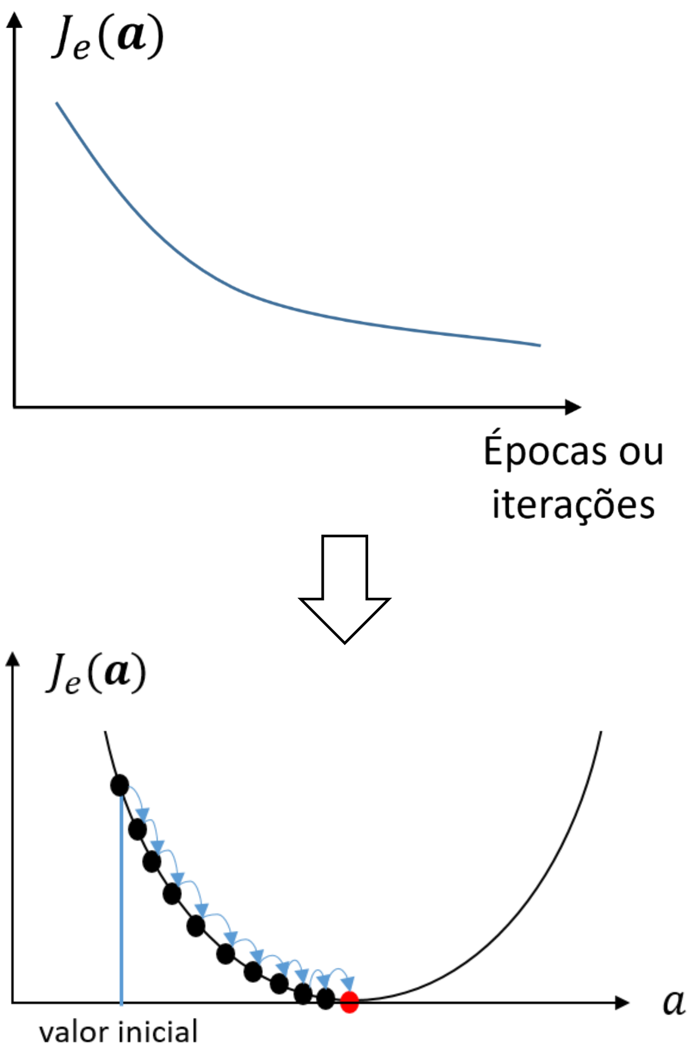
Analisando o treinamento de um modelo





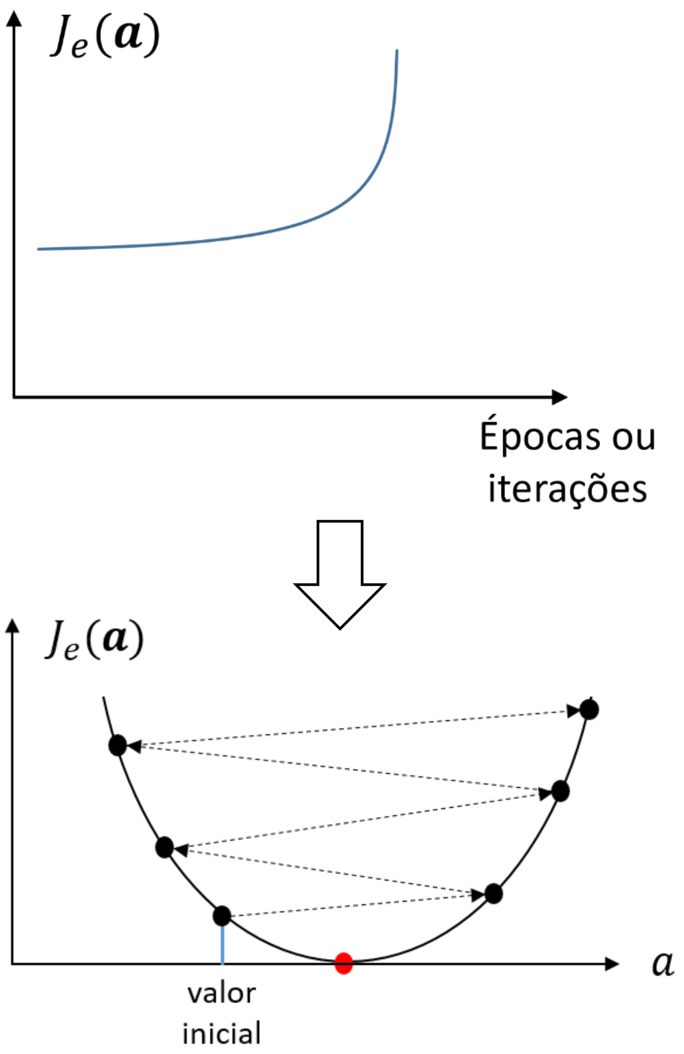
- A figura ao lado mostra o ***comportamento esperado*** quando o ***passo tem o tamanho ideal***.
- A ***convergência*** nesse caso é ***rápida***.
  - O ***erro diminui rapidamente nas primeiras épocas*** (ou iterações).
  - Conforme o treinamento continua, o ***erro se estabiliza e exibe uma redução suave*** (i.e., mais lenta).
  - A ***convergência é atingida quando o erro se torna praticamente constante*** ao longo das épocas, indicando que os ***pesos não são mais atualizados***, pois o mínimo da função foi atingido.
  - O treinamento pode ser encerrado quando o erro entre duas épocas consecutivas for menor do que um valor pré-definido (e.g., 1e-5).

Analisando o treinamento de um modelo



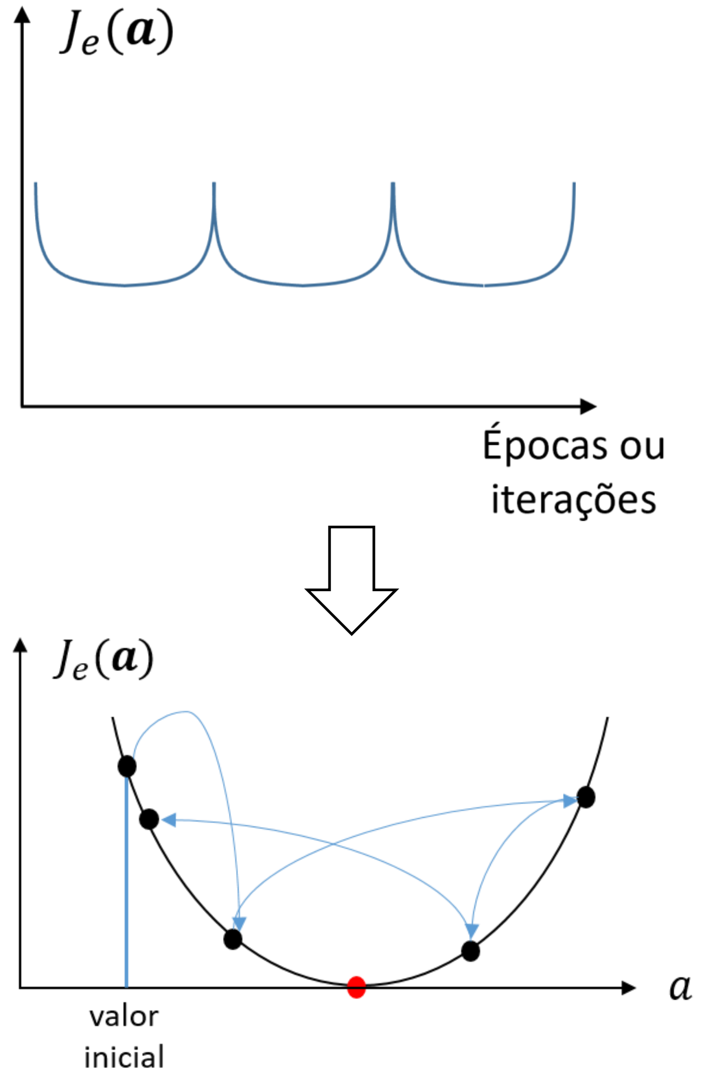
- A figura mostra o caso onde o ***passo de aprendizagem é muito pequeno***.
- Nesse caso, a ***convergência é muito lenta***.
- ***Após várias épocas*** de treinamento, o ***erro ainda não se estabilizou***.
- Levaria ***muito tempo*** para que o modelo atingisse o ***ponto de mínimo***.

Analisando o treinamento de um modelo



- A figura mostra o caso onde o ***passo de aprendizagem é muito grande***.
- Nesse caso, ocorre ***divergência***.
- Ou seja, o ***erro aumenta mais e mais ao longo do treinamento***, indicando que o algoritmo está se ***distanciando do ponto de mínimo***.
- Se o treinamento continuar, os gradientes e pesos podem se tornar tão grandes que ocorre o ***estouro da representação numérica***.

Analisando o treinamento de um modelo



- A figura mostra o caso onde o ***passo de aprendizagem é grande, mas não tão grande assim***.
- Nesse caso, o ***erro oscila*** entre valores grandes e pequenos.
- Por ventura, a ***convergência pode ocorrer após algumas épocas***.

Melhorando a convergência das versões estocásticas

- As versões estocásticas do gradiente descendente, i.e., SGD e mini-*batch* (principalmente quando MB é pequeno), têm um ***caminho irregular para o ponto de mínimo***.
- Além disso, quando as ***amostras*** do conjunto de treinamento estão ***contaminadas com ruido***, eles ***podem não convergir*** para o mínimo (oscilam ao redor dele).
- Esses problemas ***impactam*** o ***desempenho do modelo*** e deixam o ***treinamento lento*** e, possivelmente, ***instável***.
- Entretanto, existem ***técnicas para minimizar*** esses problemas, deixando essas versões do GD ***mais comportadas***.
- As mais conhecidas envolvem o ***ajuste do passo de aprendizagem*** e/ou do ***termo de atualização dos pesos***.

Ajuste do passo de aprendizagem

- ***Redução gradual*** (ou decaimento) do passo de aprendizagem ***diminui gradualmente o passo de aprendizagem*** ao longo do treinamento.
- A redução da taxa de aprendizagem faz com que as ***atualizações dos pesos se tornem cada vez menores*** à medida que o treinamento progride, o que pode ***melhorar (ou forçar) a convergência***.

$$\boldsymbol{a}(i+1) = \boldsymbol{a}(i) - \boxed{\alpha(i)} \nabla \widehat{J_e}\left(\boldsymbol{a}(i)\right),$$

onde i é número da iteração de atualização atual e $\nabla \widehat{J_e}\left(\boldsymbol{a}(i)\right)$ é a ***estimativa do vetor gradiente***.

- Essa é a técnica mais simples das que veremos, mas, precisamos encontrar os hiperparâmetros que dão a taxa ideal de redução do passo de aprendizagem.
- Veremos um exemplo de como ela funciona.

Técnicas mais comuns para a redução gradual

- As três técnicas mais comuns para a ***redução gradual*** do passo de aprendizagem são:
  - **Decaimento por etapas ou degraus**: reduz o passo de aprendizagem inicial, α_0, de um fator, τ, a cada número pré-definido de iterações, β. Um valor típico para reduzir a taxa de aprendizado é de $\tau = 0.5$ a cada β de iterações.
  - **Decaimento exponencial**: é dado pela equação $\alpha(i) = \alpha_0 e^{-ki}$, onde α_0, k e i são passo de aprendizagem inicial, a taxa de decrescimento e o número da iteração de atualização atual, respectivamente.
  - **Decaimento temporal**: é dado pela equação $\alpha(i) = {}^{\alpha_0}/_{(1+ki)}$ onde α_0, k e i tem o mesmo significado que no decaimento exponencial.
- Entretanto, percebam que ainda temos que encontrar os valores ideais para os ***hiperparâmetros*** α_0, τ, β e k.

Ajuste do termo de atualização dos pesos

- O ***termo momentum*** adiciona a ***média do histórico de estimativas do vetor gradientes***, $\boldsymbol{v}$, à equação de atualização dos pesos, ***tornando as atualizações menos ruidosas***, e, consequentemente, ***acelerando a convergência*** do algoritmo.

$$\boldsymbol{v}(i) = \mu\boldsymbol{v}(i-1) + (1-\mu)\nabla\widehat{J_e}\,(\boldsymbol{a}(i)),$$
$$\boldsymbol{a}(i+1) = \boldsymbol{a}(i) - \alpha\boldsymbol{v}(i).$$

onde $\nabla\widehat{J_e}\,(\boldsymbol{a}(i))$ é a ***estimativa do vetor gradiente*** e μ, chamado de ***coeficiente de momentum***, determina a quantidade de estimativas anteriores que são consideradas no cálculo da média.

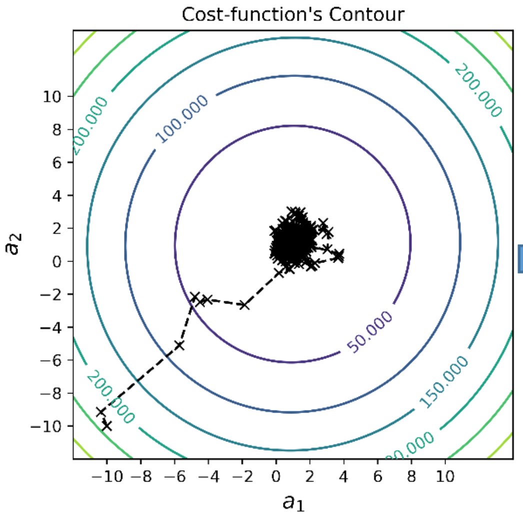
- O passo de aprendizagem é constante.
- A ***desvantagem*** é que nós precisamos encontrar as valores ideais dos ***hiperparâmetros*** α e μ.

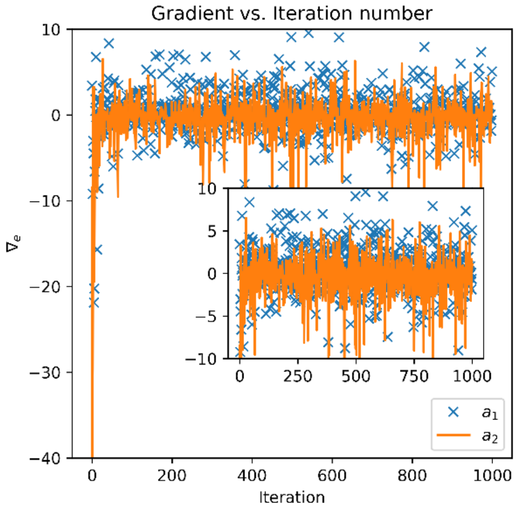
Ajuste dos pesos e de seu termo de atualização

- Na ***variação adaptativa***, o passo de aprendizagem é ***ajustado adaptativamente*** de acordo com a ***inclinação da superfície de erro***.
- Além disso, usa ***passos de aprendizagem diferentes para cada peso*** do modelo, ***os atualizando de forma independente*** de acordo com a inclinação da superfície na direção dos pesos.
- Pode ser ***combinado com o termo momentum*** para ajustar o termo de atualização dos pesos, melhorando ainda mais a convergência.
- Uma ***vantagem*** é que na maioria dos casos, ***não é necessário se ajustar manualmente nenhum hiperparâmetro*** como no caso das técnicas de redução gradual e termo momentum.
- As técnicas mais conhecidas são RMSProp, AdaGrad e Adam.

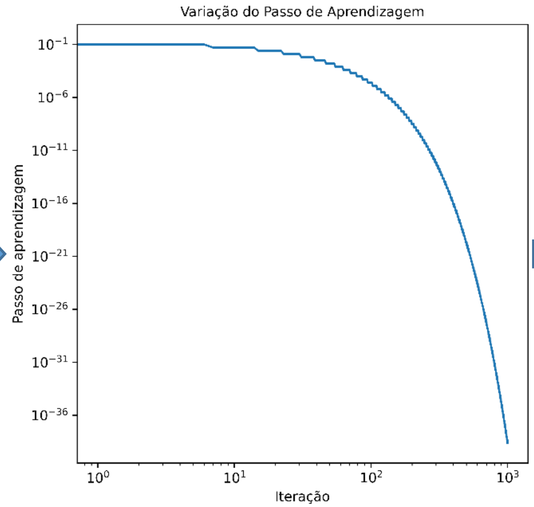
Exemplo de redução programada com GDE

Sem redução gradual



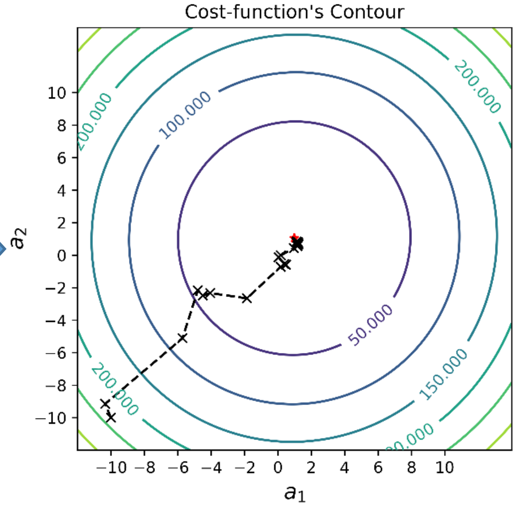


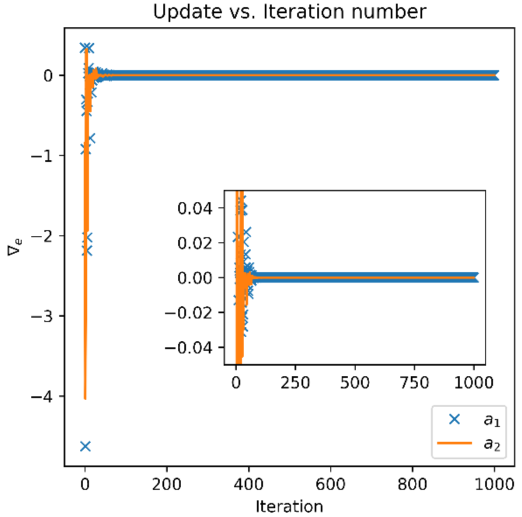
Redução gradual por degraus




```
# learning schedule: Gradual decay.
def stepDecay(alpha_init, t, beta=6.0, drop=0.5):
    '''Gradual decay.'''
    alpha = alpha_init * math.pow(drop, math.floor((1+t)/beta))
    return alpha
```

Com redução gradual



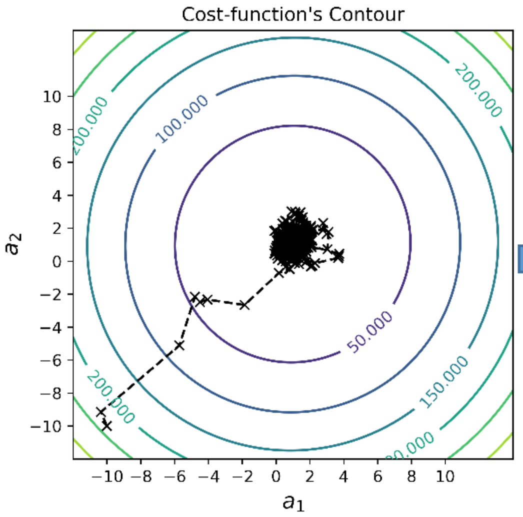


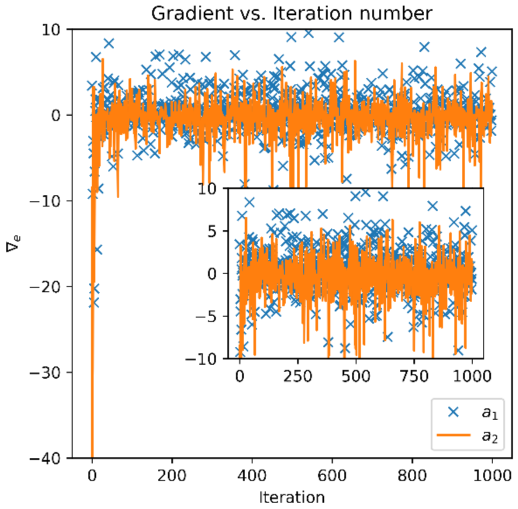
- O caminho com ***decaimento gradual*** também não é regular para o ponto de mínimo.
- Ele apresenta ***algumas mudanças de direção*** ao longo do caminho.
- O passo não influencia na direção, apenas no tamanho do deslocamento.

[Exemplo: gde_com_redução_gradual.ipynb]

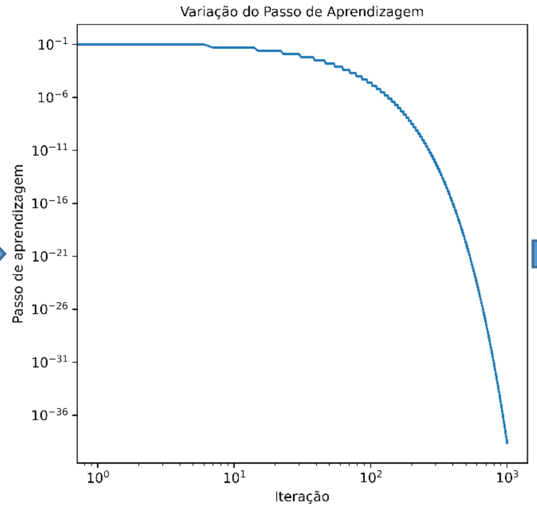
Exemplo de redução programada com GDE

Sem redução gradual



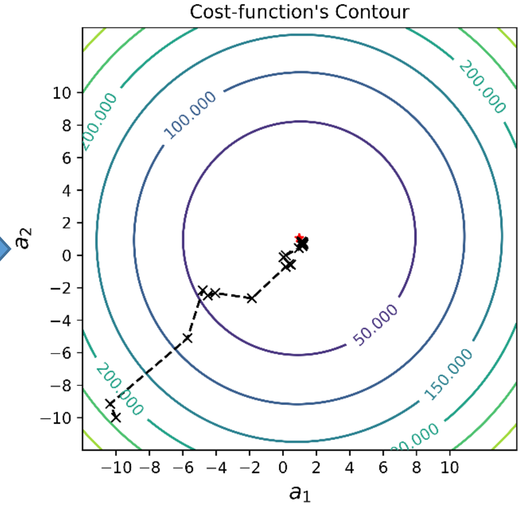


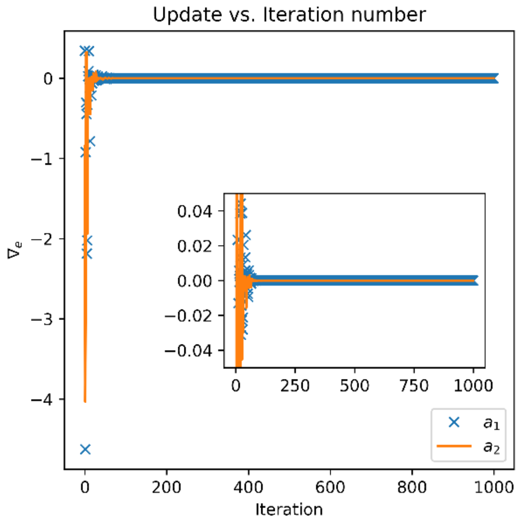
Redução gradual por degraus




```python
# learning schedule: Gradual decay.
def stepDecay(alpha_init, t, beta=6.0, drop=0.5):
    '''Gradual decay.'''
    alpha = alpha_init * math.pow(drop, math.floor((1+t)/beta))
    return alpha
```

Com redução gradual

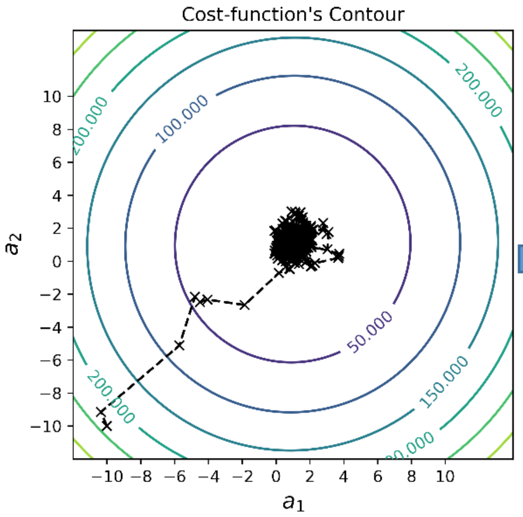


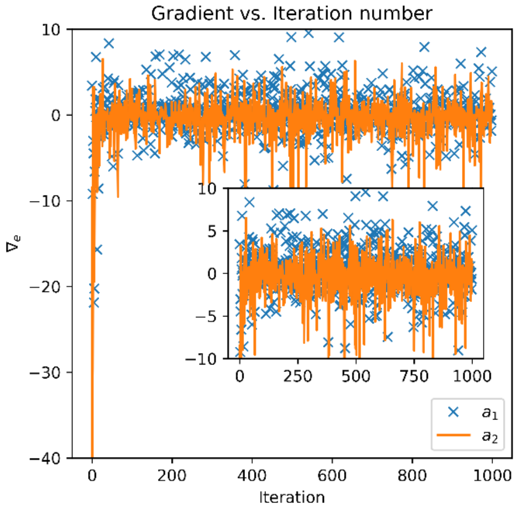


- Porém, a ***oscilação em torno do mínimo é bastante reduzida*** devido à ***diminuição gradual*** do passo de aprendizagem, α.
- Conseguimos visualizar melhor o efeito da redução de α nas figuras que mostram os elementos do vetor gradiente.

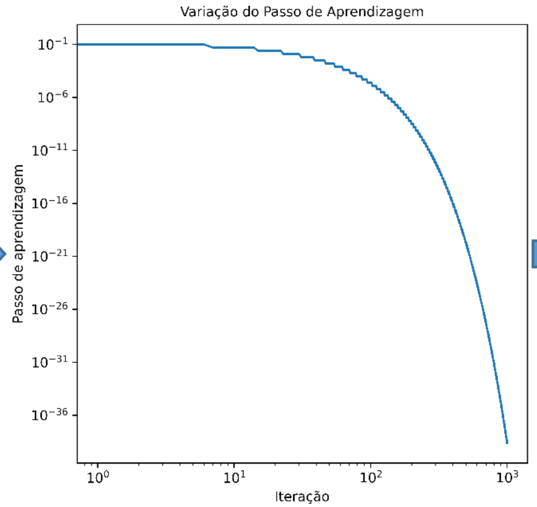
Exemplo de redução programada com GDE

Sem redução gradual



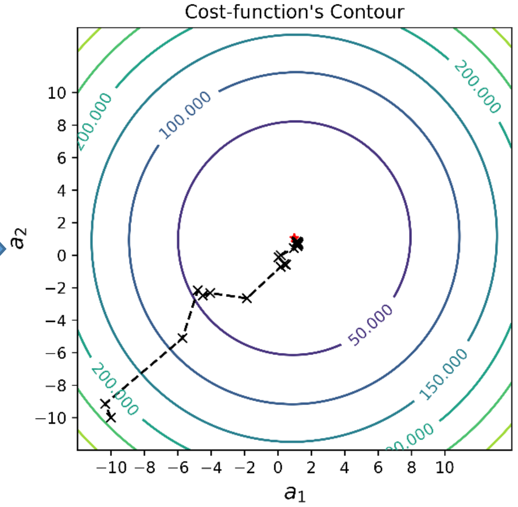


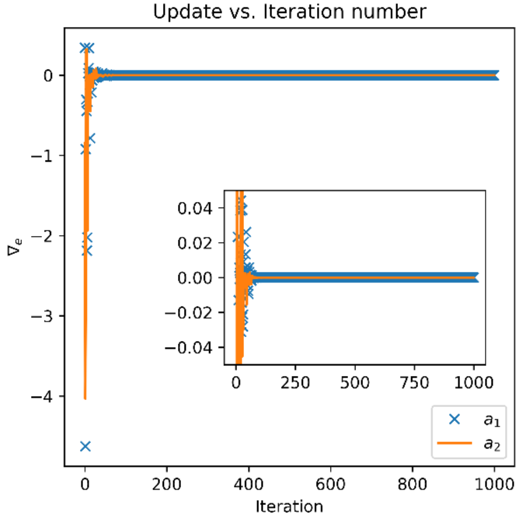
Redução gradual por degraus




```python
# learning schedule: Gradual decay.
def stepDecay(alpha_init, t, beta=6.0, drop=0.5):
    '''Gradual decay.'''
    alpha = alpha_init * math.pow(drop, math.floor((1+t)/beta))
    return alpha
```

Com redução gradual



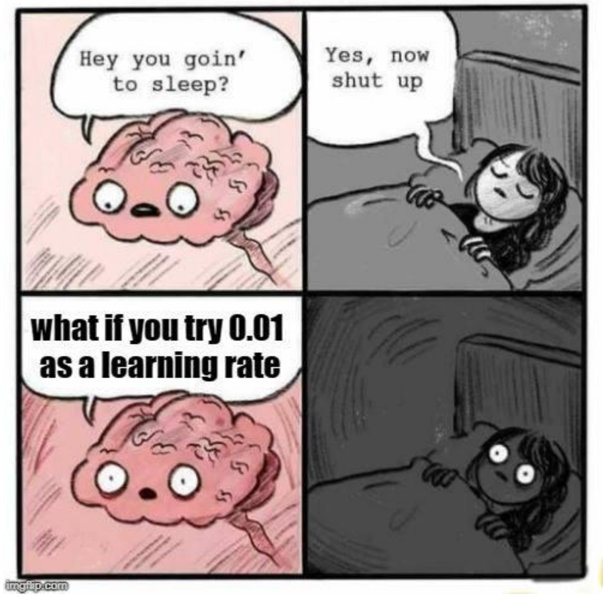


- **Conclusão**: um passo de aprendizagem que tem seu valor reduzido ao longo das iterações de treinamento permite que que versões estocásticas do gradiente descendente se estabilizem próximo ao ponto de mínimo global.

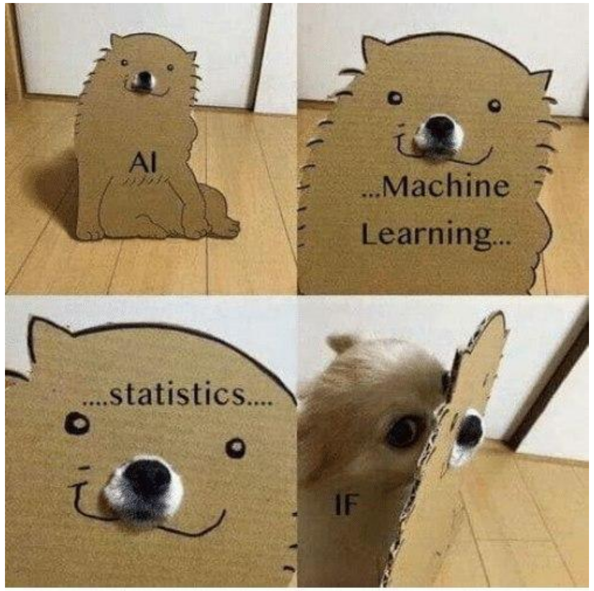
Tarefas

- **Quiz**: "*T319 - Quiz - Regressão: Parte III*" que se encontra no MS Teams.
- **Exercício Prático**: **Laboratório #4**.
  - Pode ser acessado através do link acima (Google Colab) ou no GitHub.
  - Vídeo explicando o laboratório: Arquivos -> Material de Aula -> Laboratório #4
  - Se atentem aos prazos de entrega.
  - Instruções para resolução e entrega dos laboratórios.

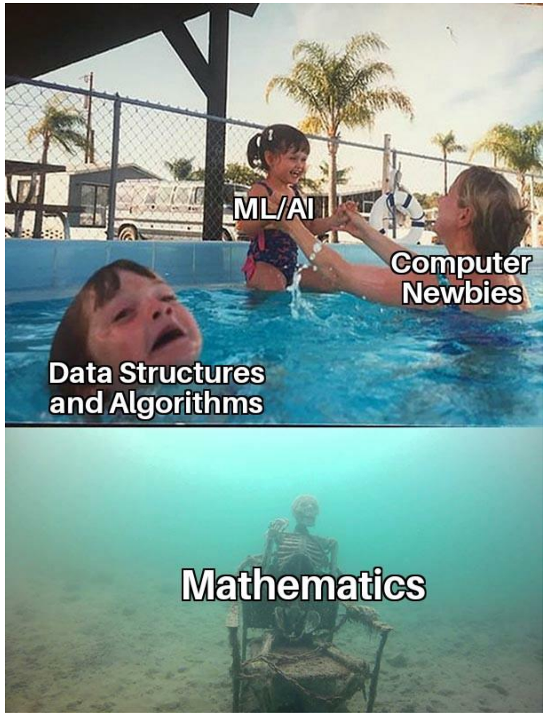
Obrigado!



When someone asks why you never stops talking about machine learning

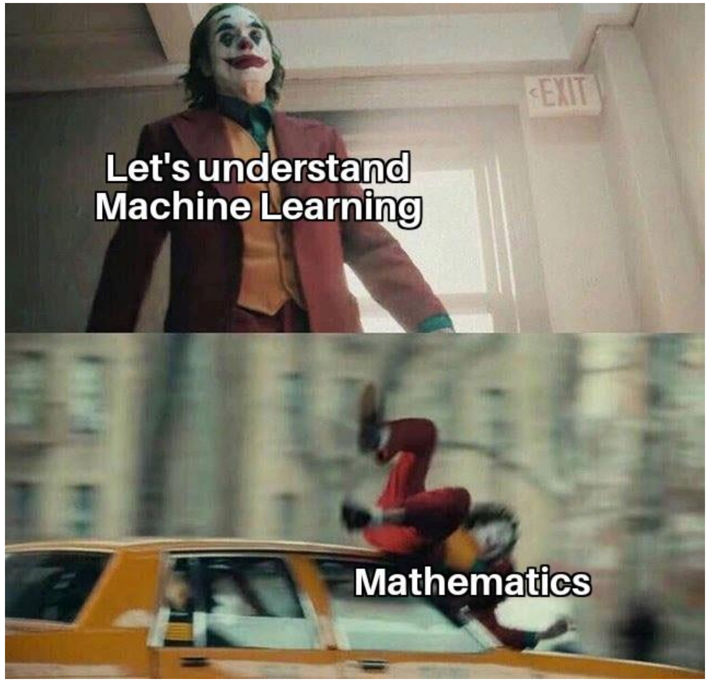


IF IF IF IF IF IF IF IF IF WE!

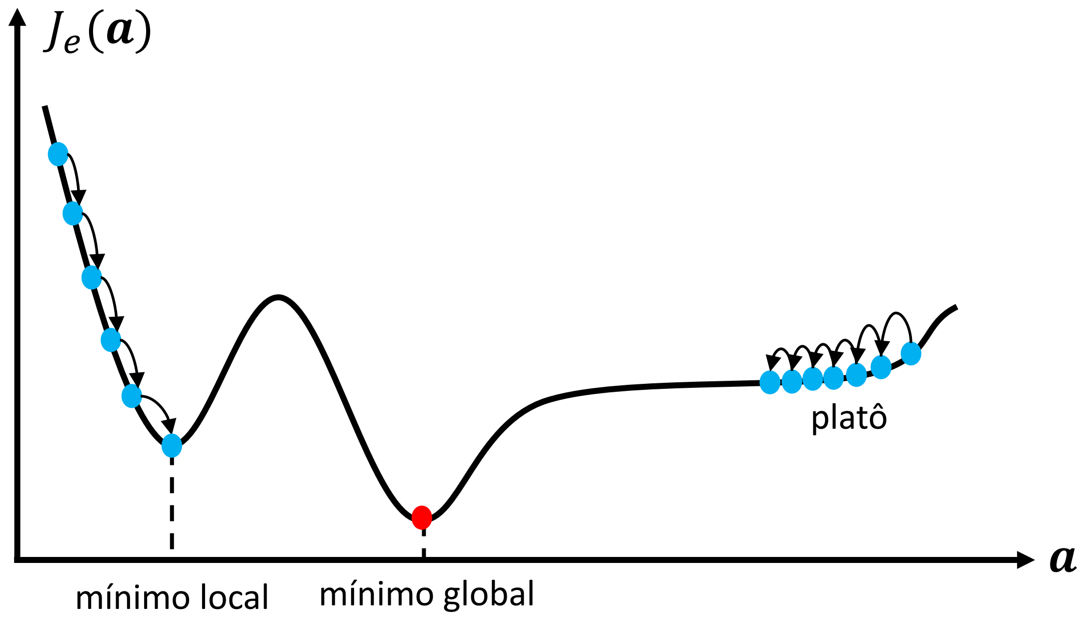


Albert Einstein: Insanity Is Doing the Same Thing Over and Over Again and Expecting Different Results

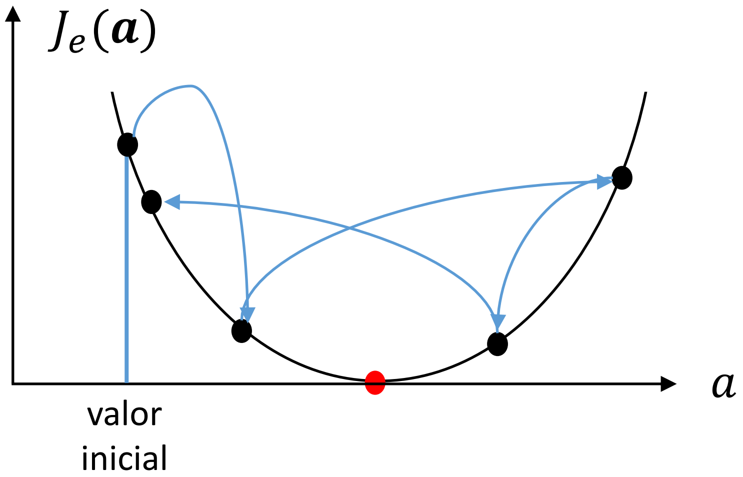
Machine learning:

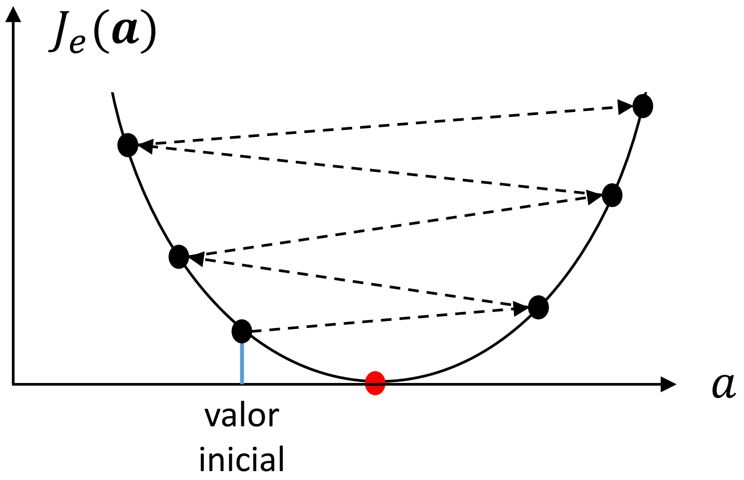


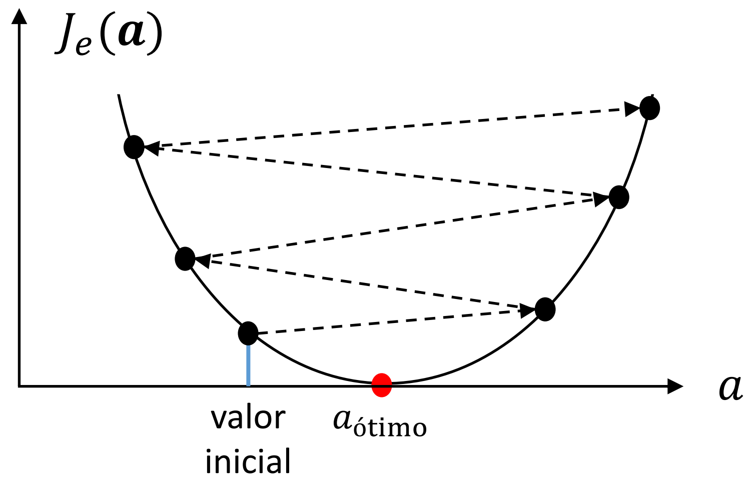
FIGURAS

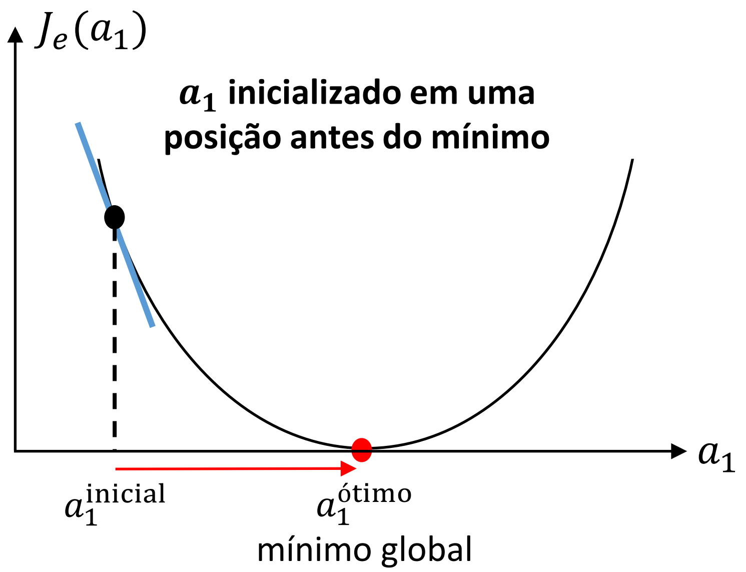


$J_e(\boldsymbol{a})$
$\boldsymbol{a}$
mínimo local
mínimo global
platô


$J_e(\boldsymbol{a})$
a
valor
inicial

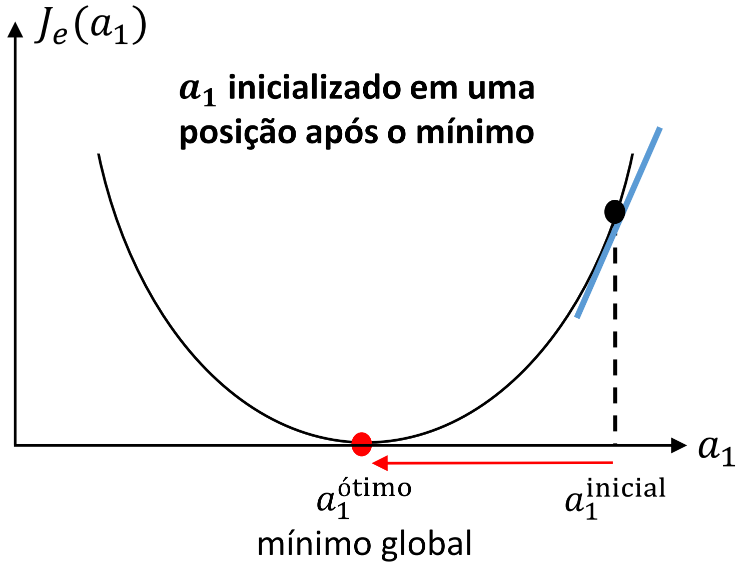

$J_e(\boldsymbol{a})$
a
valor
inicial


$J_e(\boldsymbol{a})$
a
valor inicial
$a_{\text{ótimo}}$



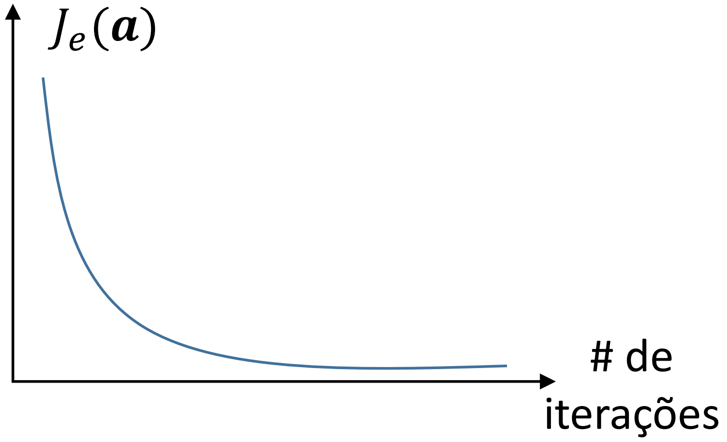
gradiente negativo: $a_1 = a_1^{\text{inicial}} + \alpha \nabla J_e(a_1)$

a_1 aumenta e se aproxima do mínimo

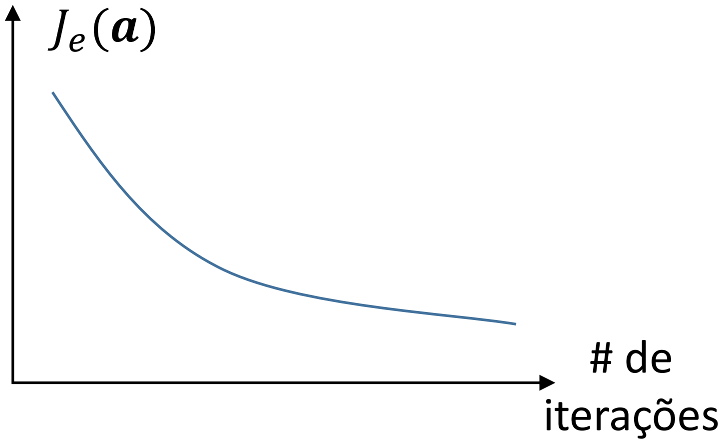


gradiente positivo: $a_1 = a_1^{\text{inicial}} - \alpha \nabla J_e(a_1)$

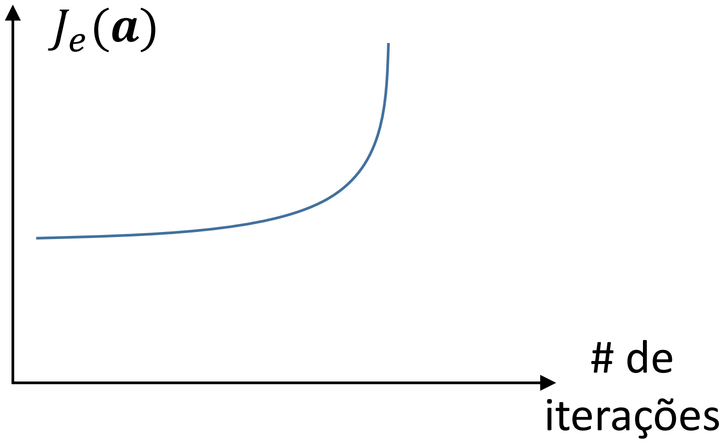
a_1 diminiu e se aproxima do mínimo



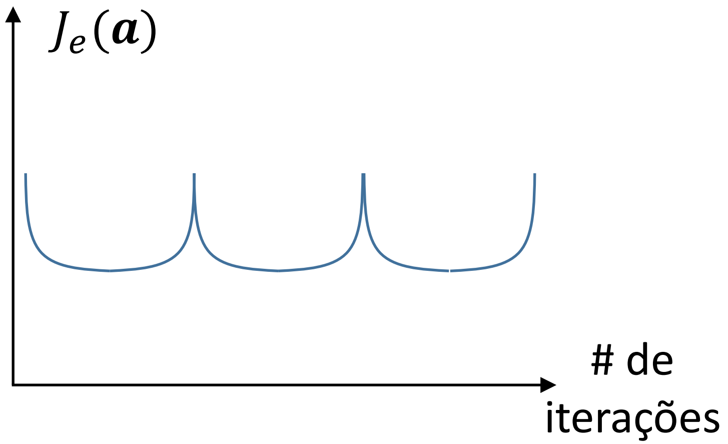
$J_e(\boldsymbol{a})$
de iterações



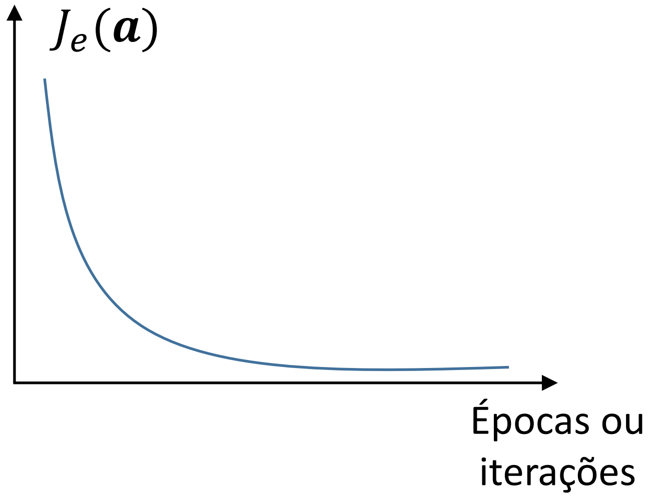
$J_e(\boldsymbol{a})$
de iterações

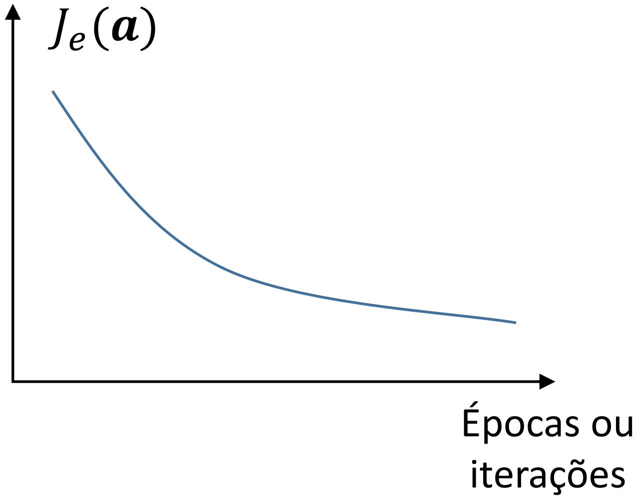


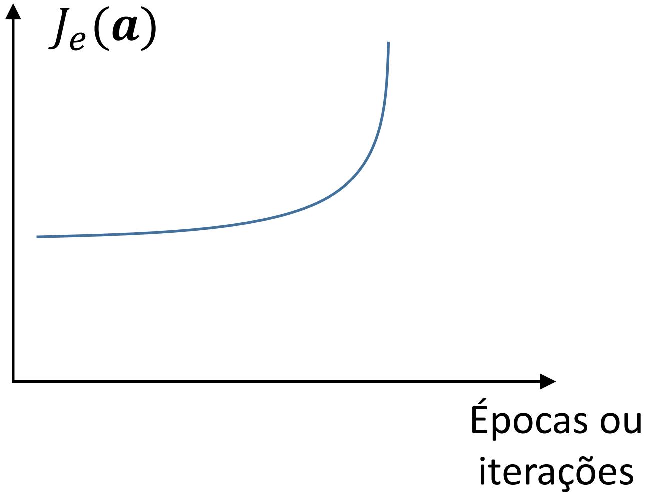
$J_e(\boldsymbol{a})$
de iterações

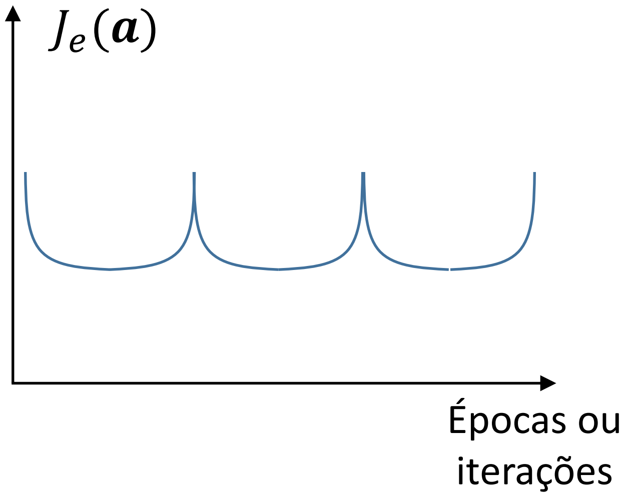


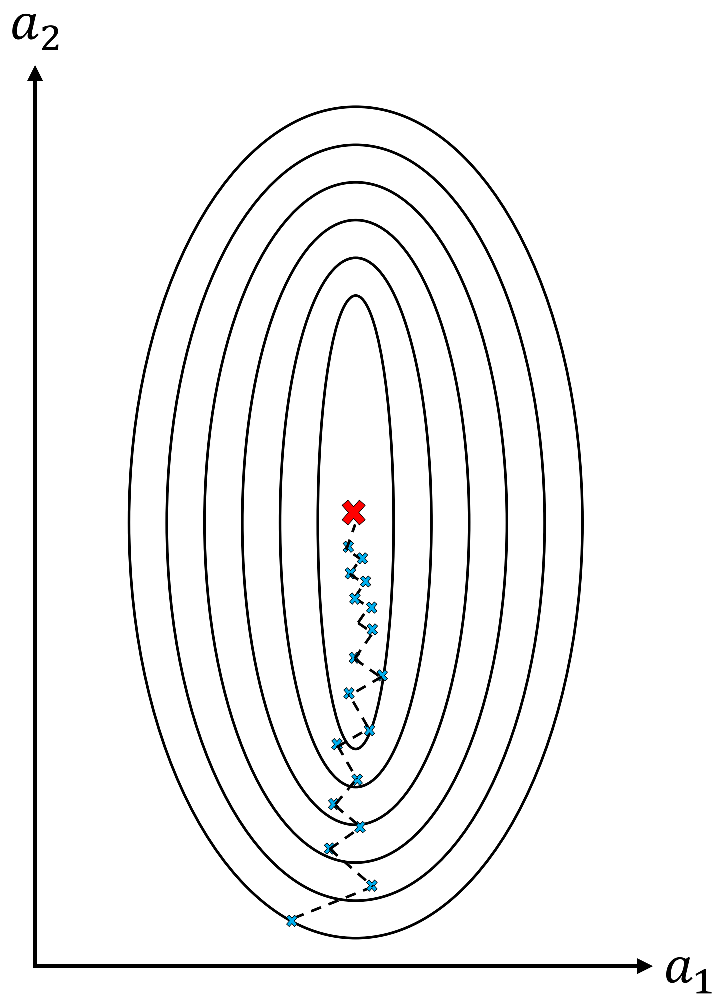
$J_e(\boldsymbol{a})$
de iterações

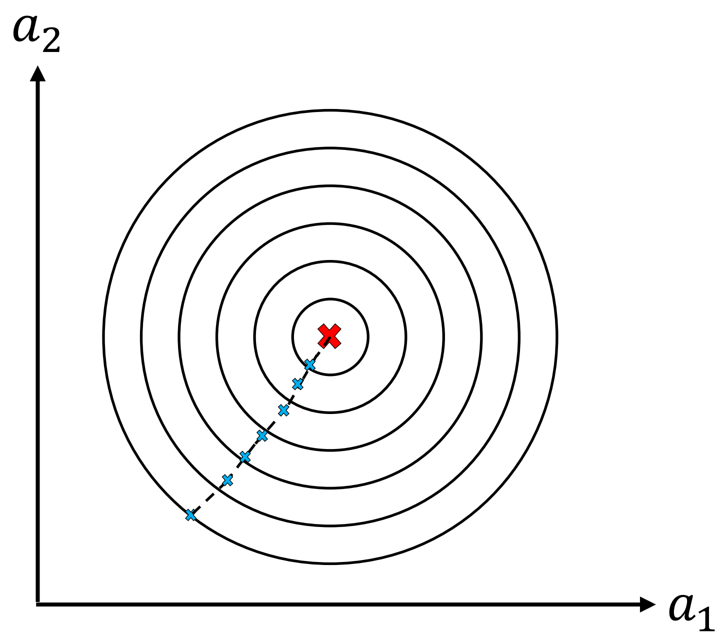

$J_e(\boldsymbol{a})$
Épocas ou iterações

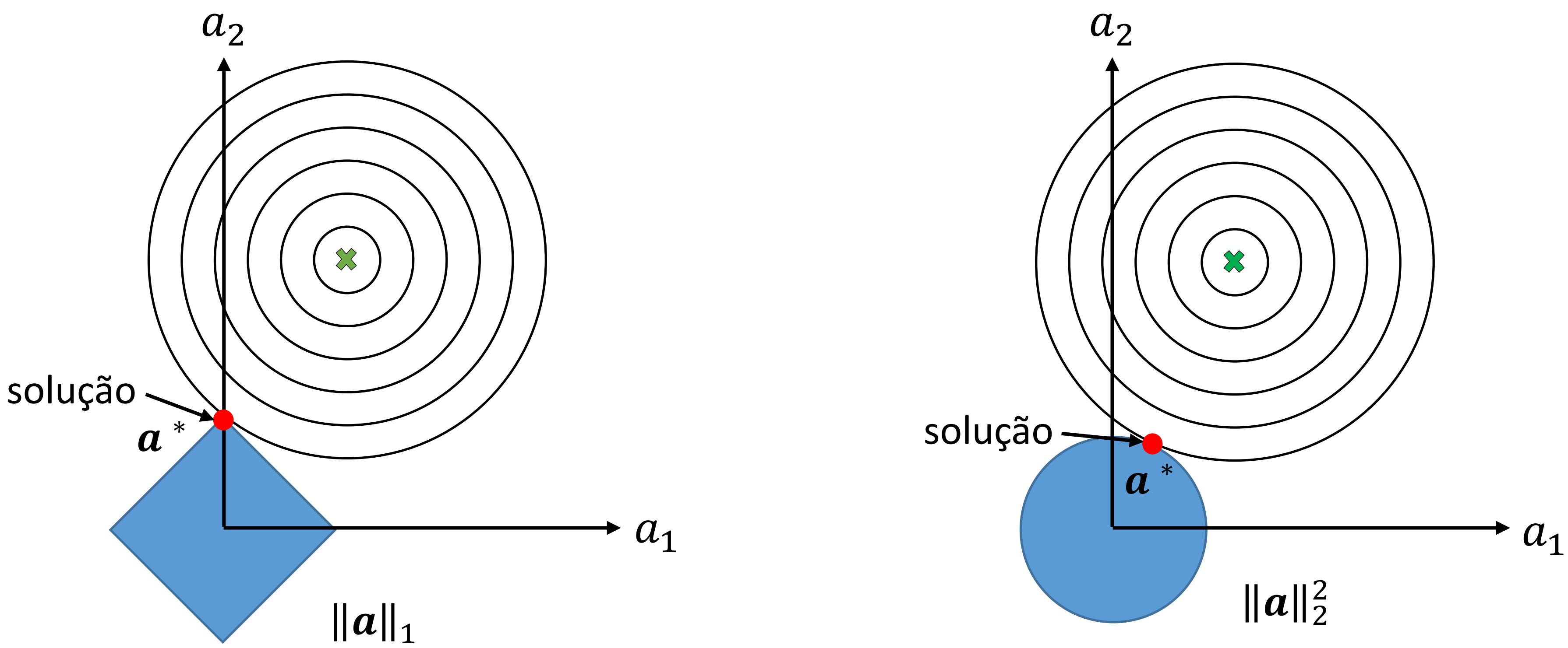

$J_e(\boldsymbol{a})$
Épocas ou iterações


$J_e(\boldsymbol{a})$
Épocas ou iterações


$J_e(\boldsymbol{a})$
Épocas ou iterações

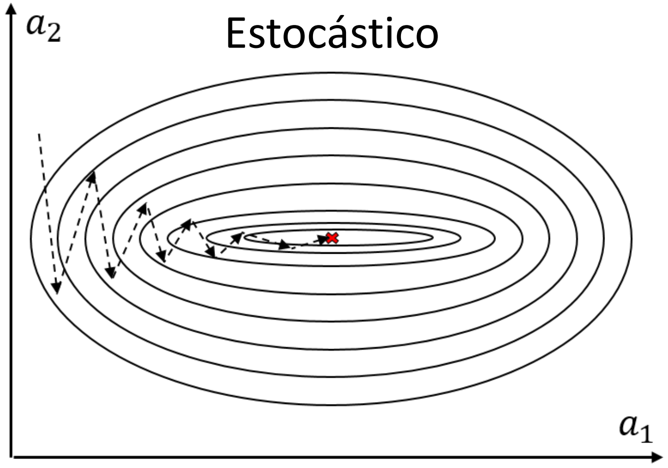

a_2
a_1

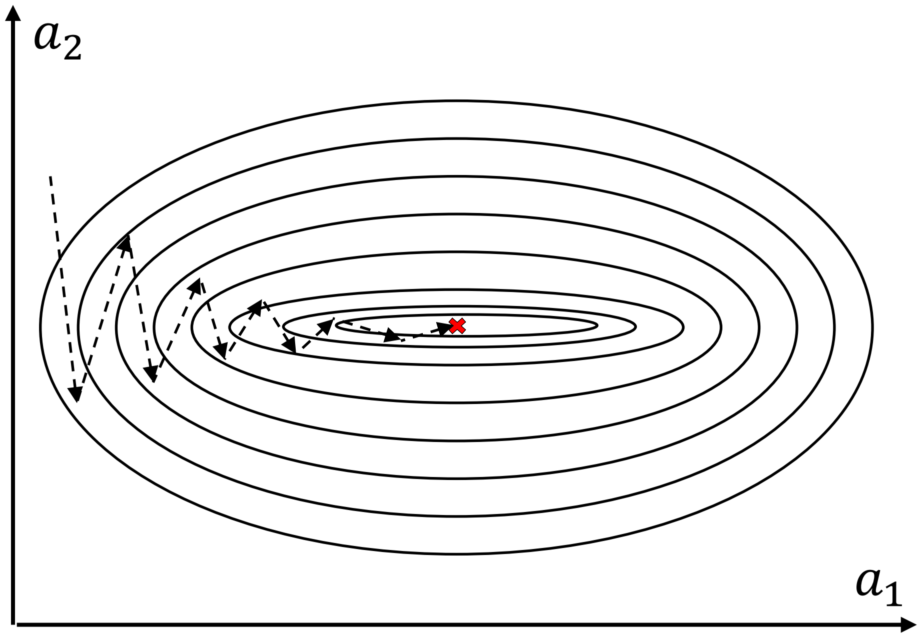

a_2
a_1



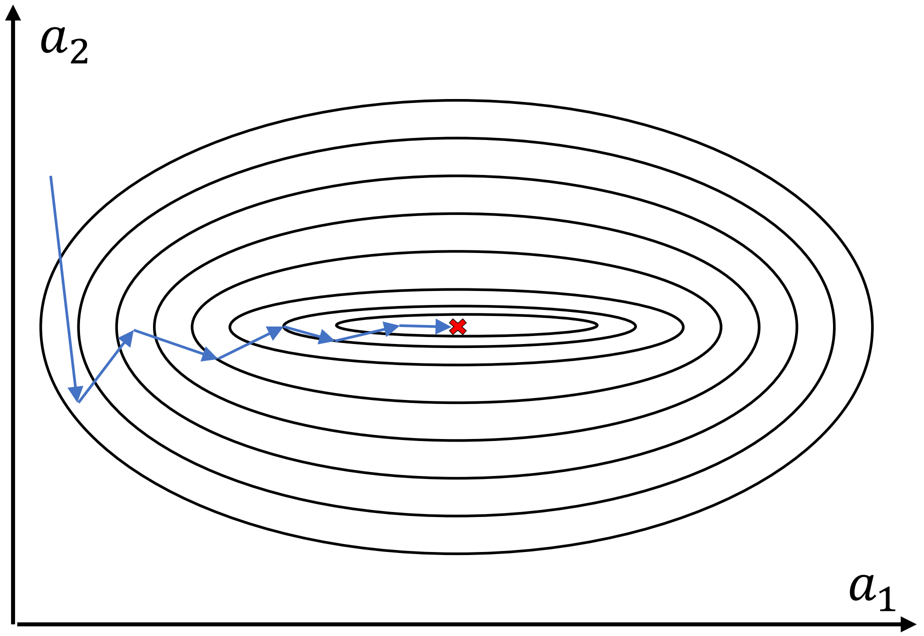
a_2
solução
$\boldsymbol{a}^*$
a_1
$\|\boldsymbol{a}\|_1$
a_2
solução
$\boldsymbol{a}^*$
a_1
$\|\boldsymbol{a}\|_2^2$

Gradiente Descendente Estocástico

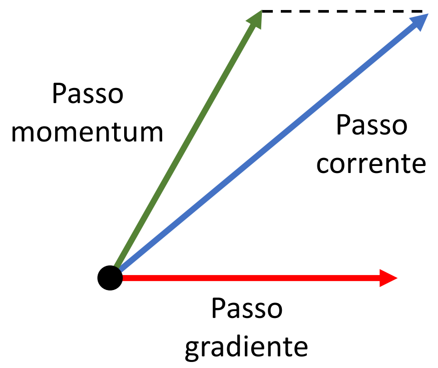




a_2
a_1



a_2
a_1


Passo
momentum
Passo
corrente
Passo
gradiente


THE RAVINE