Introdução ao Processamento de Dados
Turma 3 (2020.1)

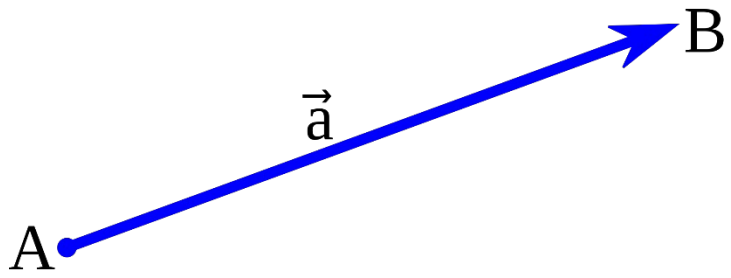
Introdução a Vetores

Gilson. A. O. P. Costa (IME/UERJ)

gilson.costa@ime.uerj.br

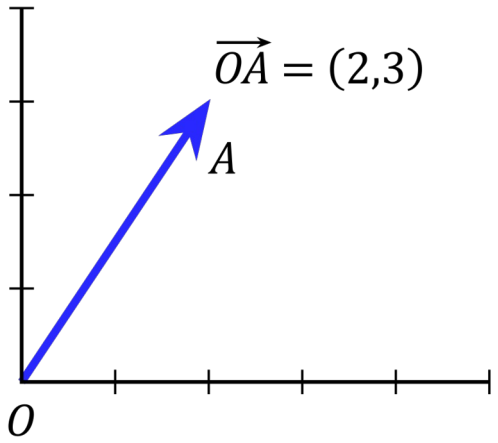
Vetores na Matemática

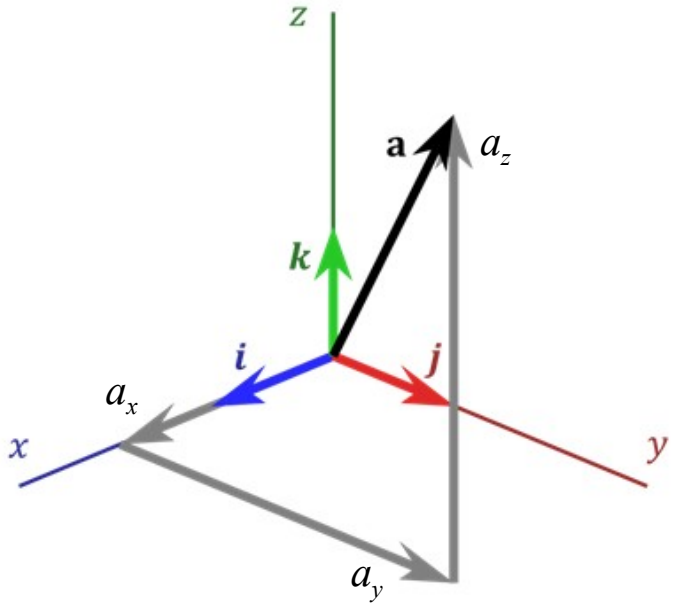
- Em matemática, física e engenharia, o conceito de vetor é comumente associado a um vetor Euclidiano (às vezes chamado de vetor geométrico ou vetor espacial).
- Um vetor Euclidiano é um objeto geométrico que tem magnitude (ou comprimento) e direção.
- Uma interpretação comum é que um vetor transporta um ponto no espaço (A) para outro ponto no espaço (B).



Vetores na Matemática

- Vetores em um espaço Euclidiano n-dimensional podem ser caracterizados pelas suas coordenadas, em algum sistema de coordenadas (e.g., Cartesianas).





Num espaço tridimensional ():

$\mathbf{a} = (a_x, a_y, a_z)$

$\mathbf{a} = (a_1, a_2, a_3)$

Num espaço *n*-dimensional ():

$\mathbf{a} = (a_1, a_2, a_3, \ldots, a_{n-1}, a_n)$

Vetores em Computação

- **Vetores em computação** também podem ser interpretados como coordenadas em um espaço qualquer.
- Mas eles têm uma definição mais geral: uma **lista ordenada de valores** (geralmente numéricos).
- Vetores básicos em Python são implementados por **listas**.
- Exemplos:

[7.5, 8.0, 5.5, 9.0, 8.0]

['Maria', 7.5, 'José', 8.0, 'Ana', 5.5, 'Ludimila', 9.0, 'Bento', 8.0]

Vetores/Listas em Python

- Cada elemento da vetor possui um **índice em sequência**.
- Exemplo: notaP1 = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]

Elemento:	**7.5**	**8.0**	**5.5**	**9.0**	**8.0**	**4.5**
Índice:	0	1	2	3	4	5

- notaP1[0] equivale a ao valor 7.5
- notaP1[2] equivale a ao valor 5.5
- notaP1[1:3] equivale ao vetor [8.0, 5.5, 9.0]
- notaP1[0:5:2] equivale ao vetor [7.5, 5.5, 8.0]

Vetores/Listas em Python

- Pode-se **criar um vetor** de tamanho arbitrário, com valores arbitrários.
- Exemplo:

```
notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
```

- Pode-se criar (inicializar) um vetor de tamanho arbitrário, com os mesmos valores para todos seus elementos.
- Exemplo:

```
>>> notas = [0.0]*6
>>> print(notas)
>>> [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

Vetores/Listas em Python

- Pode-se **criar um vetor** de tamanho arbitrário, com valores arbitrários.
- Exemplo:

```
notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
```

- Pode-se criar (inicializar) um vetor de tamanho arbitrário, com os mesmos valores para todos seus elementos.
- Exemplo:

```
>>> notas = [1.0]*6
>>> print(notas)
>>> [1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
```

Vetores/Listas em Python

- Pode-se **criar um vetor** de tamanho arbitrário, com valores arbitrários.
- Exemplo:

```
notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
```

- Pode-se criar (inicializar) um vetor de tamanho arbitrário, com os mesmos valores para todos seus elementos.
- Exemplo:

```
>>> nomes = ['X']*6
>>> print(nomes)
>>> ['X', 'X', 'X', 'X', 'X', 'X']
```

Vetores/Listas em Python

- Para **modificar um elemento** do vetor, basta fazer uma atribuição àquele elemento.
- Exemplo:

```
>>> notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
>>> notas[1]= 4.0
>>> print(notas)
>>> [7.5, 4.0, 5.5, 9.0, 8.0, 4.5]
```

Vetores/Listas em Python

- Quando se tenta **acessar um elemento** do vetor que não existe (através de um **índice inválido**) o programa vai dar um **erro fatal!**
- Exemplo:

```
>>> notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
>>> notas[6] = 4.0 # esta instrução vai dar erro!
>>> print(notas[6]) # esta também!
```

Vetores/Listas em Python

- Para saber o **número de elementos** do vetor, pode-se usar o comando ***len(nome_do_vetor)***.
- Exemplo:

```
>>> notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
>>> elementos = len(notas)
>>> print(elementos)
>>> 6
```

Exercício

Criar um programa que lê as notas de uma turma de N alunos, e indica o número de alunos acima da média.

Exercício

Criar um programa que lê as notas de uma turma de N alunos, e indica o número de alunos acima da média.

```
N = int(input("Quanto alunos há na turma? "))
notas = [0.0]*N
soma = cont = 0
for i in range(0,N):
    notas[i] = float(input("Entre com a nota #{}: ".format(i+1)))
    soma += notas[i]
media = soma/N
print("Média:", media)
for j in range(0,N):
    if notas[j] > media:
        cont += 1
print("Acima da média:", cont)
```

Vetores/Listas em Python

- Um vetor/lista em Pyhton **pode ser vazio** (análogo a um conjunto vazio).
- Exemplo:

```
>>> notas = []
>>> elementos = len(notas)
>>> print(elementos)
>>> 0
```

Métodos de Vetores/Listas

- Pode-se **acrescentar um elemento** ao final do vetor/lista usando-se o método ***append(valor_do_elemento)***.
- Exemplos:

```
>>> notas = [7.5, 8.0, 5.5, 9.0, 8.0, 4.5]
>>> notas.append(9.5)
>>> print(notas)
>>> [7.5, 4.0, 5.5, 9.0, 8.0, 4.5, 9.5]
```

Métodos de Vetores/Listas

- Pode-se **acrescentar um elemento** ao final do vetor/lista usando-se o método ***append(valor_do_elemento)***.
- Exemplos:

```
>>> notas = []
>>> notas.append(9.5)
>>> notas.append(1.0)
>>> notas.append(5.5)
>>> print(notas)
>>> [9.5, 1.0, 5.5]
```

Exercício

Criar um programa que lê as notas de uma turma de N alunos, e indica o número de alunos acima da média, utilizando o método ***append(...)***.

Exercício

Criar um programa que lê as notas de uma turma de N alunos, e indica o número de alunos acima da média, utilizando o método ***append(...)***.

```
N = int(input("Quanto alunos há na turma? "))
notas = []
soma = cont = 0
for i in range(0,N):
    nota = float(input("Entre com a nota #{}: ".format(i+1)))
    soma += nota
    notas.append(nota)
media = soma/N
print("Média:", media)
for j in range(0,N):
    if notas[j] > media:
        cont += 1
print("Acima da média:", cont)
```

Exercício

Criar um programa que lê dois vetores e produz um terceiro vetor que é a interseção dos dois primeiros. Os elementos dos vetores são lidos até que se digite um elemento negativo.

Exercício

Criar um programa que lê dois vetores e produz um terceiro vetor que é a interseção dos dois primeiros. Os elementos dos vetores são lidos até que se digite um elemento negativo.

```
vetA = []
vetB = []
vetC = []
num = int(input("Elemento do vetor A: "))
while num >= 0:
    vetA.append( num )
    num = int(input("Elemento do vetor A: "))
num = int(input("Elemento do vetor B: "))
while num >= 0:
    vetB.append( num )
    num = int(input("Elemento do vetor B: "))
...
```

```
...
for i in range(0,len(vetA)):
    for j in range(0,len(vetB)):
        if vetA[i] == vetB[j]:
            vetC.append(vetA[i])
print('Primeiro vetor:', vetA)
print('Segundo vetor:', vetB)
print('Vetor interseção:', vetC)
```

Métodos de Vetores/Listas

- O método ***count(valor_do_elemento)*** conta o número de ocorrências de ***valor_do_elemento*** no vetor/lista.
- Exemplos:

```
>>> notas = [7.5,  8.0, 5.5, 9.0, 8.0, 4.5]
>>> notas.count(8.0)
>>> 2
```

Métodos de Vetores/Listas

- O método ***insert(posicao, valor_do_elemento)*** é similar ao ***append***, mas acrescenta o elemento em uma posição específica do vetor/lista.
- Exemplos:

```
>>> notas = [7.5,  8.0, 5.5, 9.0, 8.0, 4.5]
>>> notas.insert(2, 0.0)
>>> print(notas)
>>> [7.5,  8.0, 0.0, 5.5, 9.0, 8.0, 4.5]
```

Métodos de Vetores/Listas

- O método ***insert(posicao, valor_do_elemento)*** é similar ao ***append***, mas acrescenta o elemento em uma posição específica do vetor/lista.
- Exemplos:

```
>>> nomes = ['Ana', 'Maria', 'Julieta']
>>> nomes.insert(0, 'Karla')
>>> print(nomes)
>>> ['Karla', 'Ana', 'Maria', 'Julieta']
```

Métodos de Vetores/Listas

- O método ***pop( )*** – sem parâmetros – executa duas ações: **remove o último elemento** do vetor e **retorna** o valor deste elemento.
- Exemplos:

```
>>> notas = [7.5,  8.0, 5.5, 9.0, 8.0, 4.5]
>>> nota = notas.pop()
>>> print(nota)
>>> 4.5
>>> print(notas)
>>> [7.5,  8.0, 5.5, 9.0, 8.0]
```

Métodos de Vetores/Listas

- O método ***pop( )*** – sem parâmetros – executa duas ações: **remove o último elemento** do vetor e **retorna** o valor deste elemento.
- Exemplos:

```
>>> nomes = ['Karla', 'Ana', 'Maria', 'Julieta']
>>> nome = nomes.pop()
>>> print(nome)
>>> 'Julieta'
>>> print(nomes)
>>> ['Karla', 'Ana', 'Maria']
```

Métodos de Vetores/Listas

- Com ***pop(posicao)*** pode-se remover e retornar um elemento numa posição arbitrária.
- Exemplo:

```
>>> nomes = ['Karla', 'Ana', 'Maria', 'Julieta']
>>> nome = nomes.pop(1)
>>> print(nome)
>>> 'Ana'
>>> print(nomes)
>>> ['Karla', 'Maria', 'Julieta']
```

Métodos de Vetores/Listas

- O método ***remove(valor_do_elemento)*** remove a primeira ocorrência de um elemento no vetor/lista.
- Exemplo:

```
>>> nomes = ['Karla', 'Ana', 'Maria', 'Julieta', 'Ana']
>>> nome = nomes.remove('Ana')
>>> print(nomes)
>>> ['Karla', 'Maria', 'Julieta', 'Ana']
```

Exercício

Criar um programa para acrescentar um elemento em uma lista se for digitado a letra 'A', remover se 'R', imprimir se 'I' e sair se 'F'.

.

Estatísticas

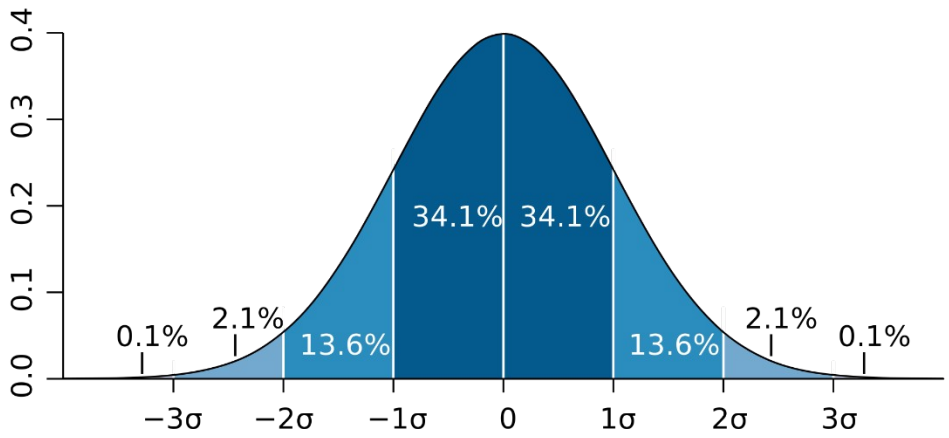
- Já tínhamos as ferramentas para calcular a média, máximo e mínimo de um conjunto de valores.
- Agora podemos calcular outra estatística importante: o **desvio padrão** destes valores.

Estatísticas

- O **desvio padrão** de um conjunto de valores é uma **medida da dispersão** destes valores no conjunto.
- Para uma distribuição normal (Gaussiana) em torno de 70% dos valores no conjunto estão contidos na faixa de a

$$= \sqrt{\frac{\sum_{=1} (\quad - \quad)^2}{}}$$

: desvio padrão
: média
: valor do elemento *i*
: número de elementos

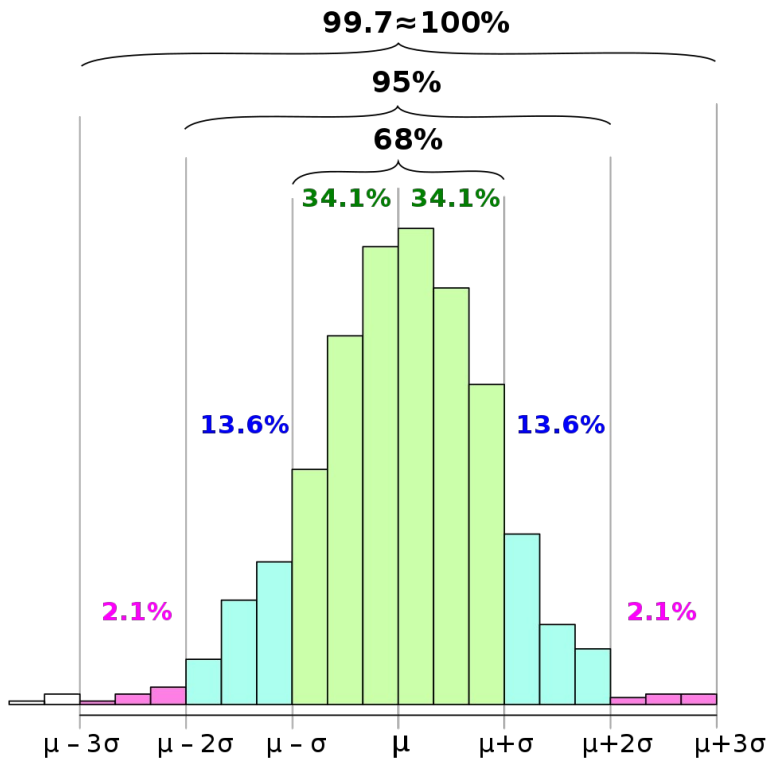


Estatísticas

- O **desvio padrão** de um conjunto de valores é uma **medida da dispersão** destes valores no conjunto.
- Para uma distribuição normal (Gaussiana) em torno de 70% dos valores no conjunto estão contidos na faixa de a

$$= \sqrt{\frac{\sum_{=1} (\quad - \quad)^2}{}}$$

: desvio padrão
: média
: valor do elemento *i*
: número de elementos



Exercício

Calcule as seguintes estatísticas para o conjunto de notas de uma determinada turma: máximo, mínimo, média e desvio padrão.

turma1 = [1.1, 7.5, 0.8, 1.8, 1.5, 1.9, 10.0, 10.0, 9.3, 10.0, 7.7, 0.6, 0.5, 8.7, 5.6, 7.0, 8.3, 7.0, 9.1, 7.4, 8.1, 7.0, 6.3, 0.6, 7.4, 2.8, 5.0, 1.4, 1.5, 0.5, 8.3, 7.0, 2.9, 7.6, 10.0, 3.3, 1.9, 5.1, 7.0]

turma2 = [10.0, 8.2, 8.7, 5.5, 6.8, 8.6, 8.5, 6.1, 6.2, 8.5, 7.7, 10.0, 10.0, 6.1, 8.4, 5.4, 5.6, 9.8, 2.1, 8.5, 3.3, 8.7, 8.5, 9.1, 9.7]

Introdução ao Processamento de Dados
Turma 3 (2020.1)

Introdução a Vetores

Gilson. A. O. P. Costa (IME/UERJ)

gilson.costa@ime.uerj.br