Prof Dr P Kantek (pkantek@gmail.com)
Matriz Inversa em Python VIVO158p, V: 1.02 1 - 1 ______ / ______ / ______

Matrizes e Equações Lineares

O sistema

$$\begin{cases} a_{11}\ x_1 + a_{12}\ x_2 + \ldots + a_{1n}\ x_n & = & b_1 \\ a_{21}\ x_1 + a_{22}\ x_2 + \ldots + a_{2n}\ x_n & = & b_2 \\ \ldots \\ a_{m1}\ x_1 + a_{m2}\ x_2 + \ldots + a_{mn}\ x_n & = & b_m \end{cases}$$

é equivalente à seguinte equação matricial

$$\begin{pmatrix} a_{11}\ a_{12}\ \ldots\ a_{1n} \\ a_{21}\ a_{22}\ \ldots\ a_{2n} \\ \ldots \\ a_{m1}\ a_{m2}\ \ldots\ a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ . \\ . \\ . \\ b_m \end{pmatrix}$$

no sentido de que toda solução do primeiro é igualmente solução do segundo. Dessa maneira pode-se resumir qualquer dos sistemas acima na forma matricial escrevendo

$$Ax = B$$

Matrizes Inversíveis

Diz-se que uma matriz quadrada é inversível se existe uma matriz B com a propriedade de que $AB = BA = I$ onde I é a matriz unidade.

A matriz B inversa de A é única se existir. A matriz inversa de A é indicada por A^{-1}. Note-se que a relação acima é simétrica, pois se B é a inversa de A então A também é inversa de B.

Mecanismos de cálculo da inversa

Usando a teoria: Usando a teoria, dada uma matriz como por exemplo $\begin{pmatrix} 3 & 5 \\ 2 & 3 \end{pmatrix}$

Uma possibilidade é querer encontrar escalares x, y, z e w para os quais $\begin{pmatrix} 3 & 5 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} x & y \\ z & w \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ ou $\begin{pmatrix} 3x+5z & 3y+5w \\ 2x+3z & 2y+3w \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ ou ainda equações que satisfaçam $\begin{cases} 3x+5z=1 \\ 2x+3z=0 \end{cases}$ e $\begin{cases} 3y+5w=0 \\ 2y+3w=1 \end{cases}$ Resolvendo estas equações, acha-se a matriz inversa que é $\begin{pmatrix} -3 & 5 \\ -2 & 3 \end{pmatrix}$.

Usando determinantes: Outros métodos (extraídos do excelente site `http://mathworld.wolfram.com`) vão a seguir descritos: Dada uma matriz 2×2 representada por $A = \begin{vmatrix} a & b \\ c & d \end{vmatrix}$, a inversa é dada por

$A^{-1} = \frac{1}{|A|} \begin{vmatrix} d & -b \\ -c & a \end{vmatrix} = \frac{1}{ad-bc} \begin{vmatrix} d & -b \\ -c & a \end{vmatrix}$.

Para uma matriz 3×3 veja lá no site, que o espaço aqui é muito pequeno.

Usando Gauss-Jordan: Neste método escreve-se a matriz da qual se quer achar a inversa ao lado esquerdo da matriz identidade, formando este arranjo (matriz escalonada reduzida por linhas).

$$[AI] = \begin{vmatrix} a_{11}\ \ldots\ a_{1n} & 1\ 0\ \ldots\ 0 \\ a_{21}\ \ldots\ a_{2n} & 0\ 1\ \ldots\ 0 \\ \ldots & \ldots \\ a_{n1}\ \ldots\ a_{nn} & 0\ 0\ \ldots\ 1 \end{vmatrix}$$

Aplica-se agora a este arranjo as técnicas de eliminação de Gauss, sobre o conjunto todo, de maneira a ficar com o seguinte arranjo $\begin{vmatrix} 1\ 0\ \ldots\ 0 & b_{11}\ \ldots\ b_{1n} \\ 0\ 1\ \ldots\ 0 & b_{21}\ \ldots\ b_{2n} \\ \ldots & \ldots \\ 0\ 0\ \ldots\ 1 & b_{n1}\ \ldots\ b_{nn} \end{vmatrix}$

Neste ponto, a matriz (b_{ij}) é a matriz inversa de A. Ou seja $A^{-1} = B$ e também $AB = I$.

A vantagem deste último método é que ele vale para qualquer dimensão de matriz. É este método que vai ser programado em Python a seguir.

Para que serve a matriz inversa ?

Inúmeras aplicações, mas vamos focar aqui na solução de sistemas lineares. Dado o sistema $Ax = B$, a matriz A é a dos coeficientes. Se você calcular a inversa A^{-1}, poderá resolver inúmeras instâncias do mesmo sistema, sem outra preocupação, já que $A^{-1} \times B = x$. Vamos ver um exemplo do que se diz aqui. Seja um sistema, dado por

$$\begin{Bmatrix} 3x+2y+z & = \\ x-y+z & = \\ 5x+z & = \end{Bmatrix}$$

Escrito na forma matricial, o sistema fica $Ax = B$, onde A é a matriz

$$\begin{pmatrix} 3 & 2 & 1 \\ 1 & -1 & 1 \\ 5 & 0 & 1 \end{pmatrix}$$

Calculando a matriz inversa desta, obtém-se:

$$\begin{pmatrix} -0.1 & -0.2 & 0.3 \\ 0.4 & -0.2 & -0.2 \\ 0.5 & 1 & -0.5 \end{pmatrix}$$

Agora, a solução para a instância onde $B = \{10, 2, 8\}$ é obtida fazendo-se $A^{-1} \times B$ que dá os valores $x = 1, y = 2, z = 3$. Para outra instância cujo B seja $\{30, -1, 26\}$, ao fazer a mesma multiplicação, obtém-se $x = 5, y = 7, z = 1$ e assim por diante.

Este exercício pede que você implemente o algoritmo de Gauss-Jordan para cálculo da matriz inversa e depois o aplique a uma matriz de ordem suficientemente alta para desencorajar o cálculo manual da mesma. Relembrando as etapas, você deve:

1. Escrever a matriz A e ao seu lado a matriz I.
2. Aplicar operações elementares com linhas a A visando transformá-la em I.
3. Aplicar rigorosamente as mesmas operações sobre a matriz ao lado.
4. Quando A tiver se transformado em I, a matriz que era originalmente I se transformou em A^{-1}.
5. Para conferir, faça a multiplicação matricial entre A e A^{-1}. Tem que dar I.

Vamos aplicar este processo ao sistema acima descrito. O objetivo é produzir um zero no coeficiente do x na segunda equação: Para isso: $a[1][] = a[1][] + a[0] \times (-a[1][0]/a[0][0])$ onde a notação $a[1][]$ deve ser entendida como a aplicação da operaçao à toda a linha [1] da matriz a. Mas, atenção: isto não vale no compilador C++. Lá, tem que ser por extenso.

Note-se também que o fator $(-a[1][0]/a[0][0])$ é o mesmo para toda a linha e portanto deve ser calculado antes e preservado durante todo o ciclo. O algoritmo vai chamar este fator de *pivot* e ele vai ser previamente calculado e guardado.

para calcular o zero no coeficiente de x na terceira linha, o comando é $a[2][] = a[2][] + a[0] \times (-a[2][0]/a[0][0])$.

Para a segunda coluna, começa-se achando o zero na primeira linha. O comando agora é $a[0][] = a[0][] + a[1][] \times (-a[0][1]/a[1][1])$ e depois $a[2][] = a[2][] + a[1][] \times (-a[2][1]/a[1][1])$. A terceira coluna é $a[0][] = a[0][] + a[2][] \times (-a[0][2]/a[2][2])$ e $a[1][] = a[1][] + a[2][] \times (-a[1][2]/a[2][2])$. Com estas operações, os zeros foram alcançados. Agora basta transformar os elementos da diagonal principal na unidade. As operações são: $a[0][] = a[0][]/a[0][0]$, $a[1][] = a[1][]/a[1][1]$ e $a[2][] = a[2][]/a[2][2]$.

Note que não houve preocupação em determinar se a matriz A é inversível. Isto foi deixado para não complicar demais o algoritmo. Fica como desafio para os mais corajosos.

Eis o algoritmo:

```
import numpy as np
def cami(x):
  # x e a matriz de ordem ta de quem
  #      se quer obter a inversa
  ta=len(x[0])
  y=np.identity(ta,float) #tem 0, na dp tem 1
  i=0
  while(i<ta):
    j=0
    while(j<ta):
      if (j!=i):
        pivot=-x[j][i]/x[i][i]
        k=0
        while k<ta:
          x[j][k]=x[j][k]+x[i][k]*pivot
          y[j][k]=y[j][k]+y[i][k]*pivot
          k=k+1
      j=j+1
    alvo=x[i][i]
    k=0
    while k<ta:
      y[i][k]=y[i][k]/alvo
      x[i][k]=x[i][k]/alvo
      k=k+1
    i=i+1
  return y;
a=np.array([[2.1, 3.2],[4.3,5.4]],float)
b=cami(a)
print(b)
```

Um exemplo

Para você testar seu algoritmo, use-o nesta matriz:

```
 59 57 44 42 65 23 47 66 78 71
 94 36 83 17 12 86  4 92 43 95
 12 61 18 10 94 45 94 81 30 76
  9 49 92 22 90 22 27  6 21 26
 31 45 96 26 26 75 59 39 47 98
 33 27  9 94 29  5 49 68  9 33
 61 81 17 18 61 74 16 37 34 45
 64 85 46 74 49 81 69 20 18 59
 17 54 66 65 94 54 76 54 30 76
 68 91 61 89  6 57 35 92 52 80
```

Aqui a resposta (sexta linha, sétima coluna, vezes 1000) deverá ser 151.989, deverá ser 151.989, já devidamente multiplicada por 1000.

☞ Para você fazer

A seguir uma matriz A de dimensão 10×10 (o que corresponderia a um sistema linear de 10 equações e 10 incógnitas). Você deve calcular a matriz inversa, seguindo o algoritmo acima, e depois responder qual o valor que está localizado na sexta linha e sétima coluna de A^{-1}, (em Python: a[5][6]) multiplicado por 1000. Eis a matriz:

```
 58 48 51 11  9 85 40  4 37 72
 33 75  8 34 17  1 97 24 98 57
 96 38 15 75 99 14 84  2 22 30
 64 82 83 85 94 66 45 68  4 41
 45 90 41  2 21 88 81 15 93 94
  1 17 60 25 99 62 58 48 58 24
 99 73 40 76 97 18 14 69 67 42
 20 75 76 84 50  1 38 58 67 30
 67 72 44 81 20 83 33 67 74 15
 95 41 27 70 21 20 46 71 62 21
```

Responda aqui:

$A^{-1}[5][6] \times 1000 =$