Soluções para a Lista 3
Processamento da Informação – 2020.1

Universidade Federal do ABC

Aritanan Gruber

aritanan.gruber@ufabc.edu.br

http://professor.ufabc.edu.br/∼aritanan.gruber

Entregue todos os exercícios via Tidia-4 até às 22/06/2020 às 19h00.

1. *Área de triângulos.* Sejam $a = (a_x, a_y)$, $b = (b_x, b_y)$ e $c = (c_x, c_y)$ três pontos no plano $\mathbb{R}^2$. O determinante

$$\delta = \begin{Vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{Vmatrix}$$

pode ser utilizado para determinar se a, b e c formam um triângulo ($\delta \neq 0$) e, em caso afirmativo, calcular sua área: $|\delta|/2$. Nota: quando diferente de 0, o sinal de δ especifica se o ponto c está à esquerda ou à direita do segmento $\overline{ab}$.

Escreva uma função que recebe a, b e c e devolve $|\delta|/2$.

Solução:

Avaliando o determinante, temos que $\delta = a_x b_y + b_x c_y + c_x a_y - (a_x c_y + b_x a_y + c_x b_y)$. O código que segue completa a resposta.

```
def area_triangle (ax: float, ay: float,
                   bx: float, by: float,
                   cx: float, cy: float) -> float:
  return abs (ax*by + bx*cy + cx*ay - ax*cy - bx*ay - cx*by) / 2
```

2. *Pontos internos.* Sejam $x[0..n-1]$ e $y[0..n-1]$ duas listas que especificam as coordenadas (no plano $\mathbb{R}^2$) dos vértices de um polígono convexo com $n \geq 3$ lados. Suponha que os vértices estão ordenados em sentido anti-horário e que pontos na "borda" são interiores.

Por exemplo: se $x = [0, 2, 2, 0]$ e $y = [0, 0, 2, 2]$, temos que os pontos $(1, 1)$, $(0, 0)$ e $(1, 0)$ são interiores; já os pontos $(1, 3)$ e $(-1, -1)$ são exteriores.

Escreva uma função que recebe x, y e um ponto $c = (c_x, c_y)$ e determina se c é um ponto interior ao polígono. Dica: utilize o resultado do exercício anterior.

Solução: Sejam a e b vértices do polígono em sentido anti-horário e seja r a reta determinada por eles, orientada de b a a. Caso c seja interior ao polígono, $\delta < 0$. Caso $c \in r$, então $\delta = 0$. Logo, não é difícil perceber que c pertence ao polígono se, e somente se $\delta \leq 0$ para todas as retas induzidas por pares de vértices adjacentes em sentido anti-horário.

```
def is_internal (x: [float], y: [float], cx: float, cy: float) -> bool:
  def det3pt (ax, ay, bx, by, cx, cy):
    return ax*by + bx*cy + cx*ay - ax*cy - bx*ay - cx*by

  n = len (x); assert (n >= 3)
  for i in range (n):
    j = (i+1) % n
    if det3pt (x[j], y[j], x[i], y[i], cx, cy) > 0:
      return False
  return True
```

3. *Determinantes.* Escreva uma função que recebe uma matriz quadrada $A \in \mathbb{R}^{n \times n}$ e devolve $\mathsf{det}(A)$, o determinante de A. Dica: utilize eliminação gaussiana.

   Solução: Durante uma eliminação gaussiana em uma matriz A, três operações costumam ocorrer:

   (i) Troca de posições de duas linhas i e j distintas de A,
   (ii) Multiplicação de uma linha de A por um escalar λ não nulo,
   (iii) Soma de uma linha i com um múltiplo escalar de outra linha j, substituindo a linha i.

   Tais operações têm os seguintes efeitos no cálculo do determinante de A:

   (i) $\mathsf{det}(A)$ é multiplicado por -1,
   (ii) $\mathsf{det}(A)$ é multiplicado por λ,
   (iii) $\mathsf{det}(A)$ permanece inalterado.

   Podemos utilizar uma variável d, inicialmente com valor 1, para manter o controle sobre o ocorrido:

   (i) $d = -d$,
   (ii) $d = \lambda d$,
   (iii) d permanece inalterada.

   Ao final da eliminação gaussiana em A, teremos uma matriz A' tal que $\mathsf{det}(A') = \prod_{i=0}^{n-1} A'[i][i]$, e $\mathsf{det}(A) = d \cdot \mathsf{det}(A')$. O código abaixo conclui a questão.

```python
from math import prod

def det (A: [[float]]) -> float:
  n = len (A); d = 1
  for k in range (n-1):
    l = max (range (k, n), key = lambda i: abs (A[i][k]))
    if A[l][k] == 0: return 0
    A[l], A[k] = A[k], A[l]
    if k != l: d = -d

    for i in range (k+1, n):
      c = -A[i][k] / A[k][k]
      A[i][k] = 0
      for j in range (k+1, n):
        A[i][j] += c * A[k][j]

  return d * prod (A[i][i] for i in range (n))
```

4. *Matrizes de Hilbert.* Uma matriz quadrada $H \in \mathbb{R}^{n \times n}$ é de *Hilbert* se $H[i][j] = 1/(i + j + 1)$. Por exemplo, a matriz 5×5 abaixo é de Hilbert.

$$H = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{pmatrix}.$$

(a) Escreva uma função que recebe n e devolve uma matriz de Hilbert $n \times n$.

Solução:

```python
def hilbert_matrix (n: int) -> [[float]]:
  return [[1/(i+j+1) for j in range (n)] for i in range (n)]
```

(b) Escreva uma função que recebe uma matriz de Hilbert H e devolve H^{-1}, a matriz inversa a H. Dica: use eliminação gaussiana.

Solução:

```python
def invert_matrix (A: [[float]], eps: float = 1e-6) -> (bool, [[float]]):
  n = len (A)
  # inicialmente, B é a matriz identidade
  B = [[(1 if i == j else 0) for j in range (n)] for i in range (n)]

  # eliminação gaussiana direta
  for k in range (n-1):
    l = max (range (k, n), key = lambda i: abs (A[i][k]))
    if A[l][k] == 0: return (False, B)
    if k != l:
      A[l], A[k] = A[k], A[l]
      B[l], B[k] = B[k], B[l]

    pivot = A[k][k]
    for i in range (k+1, n):
      c = -A[i][k] / pivot
      A[i][k] = 0
      for j in range (k+1, n): A[i][j] += c * A[k][j]
      for j in range (n):      B[i][j] += c * B[k][j]

  # eliminação gaussiana reversa
  for k in reversed (range (1, n)):
    pivot = A[k][k] # pivot != 0, sempre
    for i in reversed (range (k)):
      c = -A[i][k] / pivot
      A[i][k] = 0
      for j in reversed (range (k+1, n)): A[i][j] += c * A[k][j]
      for j in range (n):                 B[i][j] += c * B[k][j]

  # divisões nas diagonais
  for i in range (n):
    if abs (1.0 - A[i][i]) > eps:
      d, A[i][i] = A[i][i], 1.0
      for j in range (n): B[i][j] /= d

  return (True, B)
```

(c) Construa as matrizes H e H^{-1} para $n = 50$ usando as funções que você desenvolveu nos itens anteriores. É verdade que $H \cdot H^{-1} = I_n$, a matriz identidade $n \times n$? O que ocorre?

Solução:

A matriz de Hilbert H tem condicionamento baixo. Logo, os erros numéricos se acumulam durante a eliminação gaussiana, fazendo com que a matriz H^{-1} devolvida não seja a inversa exata de H.