May 26, 2022

Exercise 04
Algorithmic Foundations of Data Science
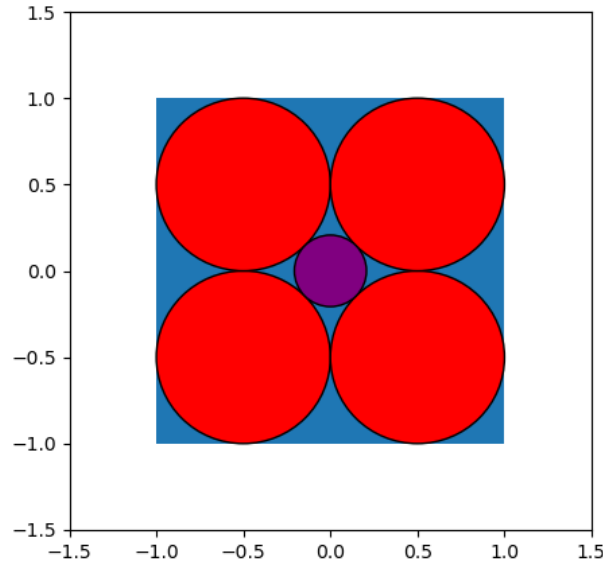
Fabian Grob
Simon Michau
Til Mohr

Exercise 1

(a)

Situation for l=2 and s=2:

$$Q_{2,2} = \{[x_1 x_2]^T \in \mathbb{R}^2 \mid |x_i| \le 1 \text{ for all } i = 1, 2\} = [-1, 1]^2$$

(b)

First, let's calculate the radius of the inner hyperball for any $l \in \mathbb{N}, s \in \mathbb{R}_{>0}$:
The distance from the center of the inner hyperball (equal to the center of the hypercube) to the center of one of the 2^l outter hyperballs (doesn't matter which one) can be calculated the following:

$$d := \sqrt{l \cdot \left(\frac{s}{4}\right)^2}$$

Thus, the radius of the inner hyperball is equal to:

$$r := d - \frac{s}{4} = \sqrt{l \cdot \left(\frac{s}{4}\right)^2} - \frac{s}{4}$$

Now, we simply must solve the following inequality to find an $l \in \mathbb{N}$ for an arbitrary but fixed $s \in \mathbb{R}_{>0}$ such that $B(Q_{l,s}) \not\subseteq Q_{l,s}$:

$$\frac{s}{2} < r$$
$$\frac{s}{2} < d - \frac{s}{4}$$
$$\frac{s}{2} < \sqrt{l \cdot \left(\frac{s}{4}\right)^2} - \frac{s}{4}$$
$$\frac{3 \cdot s}{4} < \sqrt{l} \cdot \frac{s}{4}$$
$$3 < \sqrt{l}$$
$$l > 9$$

May 26, 2022
Exercise 04
Algorithmic Foundations of Data Science

Fabian Grob
Simon Michau
Til Mohr

Exercise 2

(a)

Let's first find all eigenvalues of A_c:

$$\{\lambda \in \mathbb{R} \mid \det\left(A_c - \lambda \cdot I_3\right) = 0\}$$

$$\{\lambda \in \mathbb{R} \mid \det \begin{bmatrix} 2-\lambda & 0 & c \\ 0 & 1-\lambda & 0 \\ c & 0 & 1-\lambda \end{bmatrix} = 0\}$$

$$\{\lambda \in \mathbb{R} \mid -\lambda^3 + 4 \cdot \lambda^2 + c^2 \cdot \lambda - 5 \cdot \lambda + 2 - c^2\}$$

$$\{\lambda \in \mathbb{R} \mid (\lambda - 1) \cdot (-\lambda^2 + 3\dot\lambda + c^2 - 2)\}$$

$$\{1, \frac{3 - \sqrt{4 \cdot c^2 + 1}}{2}, \frac{3 + \sqrt{4 \cdot c^2 + 1}}{2}\} =: \Lambda_{A_c}$$

So, if for all $\lambda \in \Lambda_{A_c}$ it must hold that $\lambda \geq 0$, we must set c such that the following inequality holds:

$$\frac{3 - \sqrt{4 \cdot c^2 + 1}}{2} \geq 0$$
$$3 - \sqrt{4 \cdot c^2 + 1} \geq 0$$
$$3 \geq \sqrt{4 \cdot c^2 + 1}$$
$$9 \geq 4 \cdot c^2 + 1$$
$$8 \geq 4 \cdot c^2$$
$$2 \geq c^2$$
$$c \in (-\sqrt{2}, \sqrt{2}) \subseteq \mathbb{R}$$

(b)

Using Theorem 5.19, we create an orthogonal matrix $U \in \mathbb{R}^{n \times n}$ with the columns being the eigenvectors of A. We can now use Theorem 5.20 to create the diagonal matrix $\Lambda \in \mathbb{R}^{n \times n}$ where the eigenvalue-entry $\lambda_{i,i}$ corresponds to eigenvector in the column i of U.
A simple verification:

$$A = U\Lambda U^\top$$
$$A = U\Lambda U^{-1}$$
$$AU = U\Lambda$$
$$Av_i = v_i\lambda_{i,i} \qquad, \forall i \in [n], v_i := col_i(U)$$
$$Av_i = \lambda_{i,i}v_i \qquad, \forall i \in [n], v_i := col_i(U)$$

May 26, 2022

Exercise 04
Algorithmic Foundations of Data Science

Fabian Grob
Simon Michau
Til Mohr

Since A is positive semi-definite, all entries $\lambda_{i,i} \geq 0$. Thus, we can create a diagonal matrix Λ' consisting of the entries $\lambda'_{i,i} := \sqrt{\lambda_{i,i}}$. Thus, $\Lambda = \Lambda'\Lambda'^{\top}$. Let $B := UU\Lambda'$. Now:

$$A = U\Lambda U^{\top}$$
$$A = U(\Lambda'\Lambda'^{\top})U^{\top}$$
$$A = U(\Lambda'\Lambda'^{\top})U^{\top}$$
$$A = (U\Lambda')(\Lambda'^{\top}U^{\top})$$
$$A = (U\Lambda')(U\Lambda')^{\top}$$
$$A = BB^{\top}$$

(c)

Since A is symmetric, the following holds $\forall x, y \in R^n$:

$$\langle Ax, y \rangle = \langle x, Ay \rangle$$

Let $v_1 \in E_1, v_2 \in E_2$. Then $Av_1 = \lambda_1 v_1$ and $Av_2 = \lambda_2 v_2$.

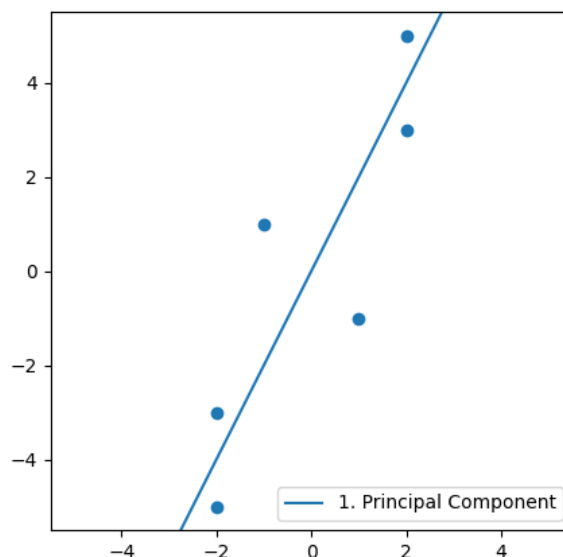$$0 = \langle Av_1, v_2 \rangle - \langle v_1, Av_2 \rangle$$
$$= \langle \lambda_1 v_1, v_2 \rangle - \langle v_1, \lambda_2 v_2 \rangle$$
$$= (\lambda_1 - \lambda_2)\langle v_1, v_2 \rangle$$

Since $\lambda_1 \neq \lambda_2$ it follows, that $\langle v_1, v_2 \rangle = 0$.

Exercise 3

(a)

Plot and estimate first principal component $(\frac{1}{\sqrt{5}}, \frac{2}{\sqrt{5}})$

May 26, 2022

Exercise 04
Algorithmic Foundations of Data Science

Fabian Grob
Simon Michau
Til Mohr

(b)

$$C := A^\top A$$

$$= \begin{bmatrix} -2 & -2 & -1 & 1 & 2 & 2 \\ -5 & -3 & 1 & 1 & 3 & 5 \end{bmatrix} \begin{bmatrix} -2 & -5 \\ -2 & 3 \\ -1 & 1 \\ 1 & -1 \\ 2 & 3 \\ 2 & 5 \end{bmatrix}$$

$$= \begin{bmatrix} 18 & 18 \\ 18 & 70 \end{bmatrix}$$

(c)

View code in the appendix for the calculations.

The eigenvector we got is: $\begin{pmatrix} 0.298 \\ 0.954 \end{pmatrix}$

(d)

Well, they both point in the same general direction with a slope greater than 1. Our slope came out to be at 2, since eye-balling stuff tends to round to integer values. The more optimal slope however seems to be even greater than $3 < \frac{0.954}{0.298}$.

Exercise 4

(a)

$$S := \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$D := \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

May 26, 2022

Exercise 04
Algorithmic Foundations of Data Science

Fabian Grob
Simon Michau
Til Mohr

$$L := D - S = \begin{bmatrix} 3 & 0 & -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 2 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 4 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 4 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 & 0 & 3 \end{bmatrix}$$

(b)

(c)

(d)

(e)

Exercise 5

Appendix

Code for Exercise 3

```python
import functools
import math


def length(v):
    return math.sqrt(functools.reduce(lambda a, b: a + b, map(lambda v_i:
    v_i ** 2, v)))


def truncate(number, decimals=0):
    """
    Returns a value truncated to a specific number of decimal places.
    """
    if not isinstance(decimals, int):
        raise TypeError("decimal places must be an integer.")
    elif decimals < 0:
        raise ValueError("decimal places has to be 0 or more.")
    elif decimals == 0:
        return math.trunc(number)

    factor = 10.0 ** decimals
    return math.trunc(number * factor) / factor


def matrix_times_vector(A, v):
    Av = list()
    for i in range(len(A)):
        assert(len(A[i]) == len(v))
        Av.append(functools.reduce(
            lambda a, b: a + b, map(lambda item: A[i][item[0]] * item[1],
    enumerate(v))))
    return Av
```

```python
31
32
33 def power_iteration(A, x):
34     x_length = length(x)
35     v = tuple(map(lambda x_i: x_i / x_length, x))
36     old_trunc_v = tuple(map(lambda v_i: truncate(v_i, 3), v))
37     while True:
38
39         Av = matrix_times_vector(A, v)
40         Av_length = length(Av)
41         v = tuple(map(lambda v_i: v_i / Av_length, Av))
42
43         # check if does not converge no more
44         new_trunc_v = tuple(map(lambda v_i: truncate(v_i, 3), v))
45         if(old_trunc_v == new_trunc_v):
46             break
47         old_trunc_v = new_trunc_v
48     return v
49
50
51 if __name__ == '__main__':
52     C = [
53         [18, 18],
54         [18, 70]
55     ]
56     v = (1,1)
57     v_power = power_iteration(C,v)
58     print(tuple(map(lambda v_i: round(v_i, 3), v_power)))
```