Foundations of data science, summer 2020

JONATHAN LENNARTZ, MICHAEL NÜSKEN, ANNIKA TARNOWSKI

8. Exercise sheet
Hand in solutions until Thursday, 11 June 2020, 12:00

Exercise 8.1 (The greedy algorithm works). (14 points)

We would like to check that the greedy algorithm works in a further sense: compare the definition to the singular value decomposition, ie. numerically the result of the numpy command `numpy.linalg.svd()`.

You may use for your implementation that if we have a singular value decomposition $A = \sum_{i<k} \sigma_i \left|u_i\right\rangle \left\langle v_i\right|$ we have that $\left|v_0\right\rangle$ is exactly the first singular vector.

(i) Prove that the second singular vector $\left|v_1\right\rangle$ is exactly the first singular vector of the matrix $\boxed{3}$

$$A \cdot (1 - \left|v_0\right\rangle \left\langle v_0\right|).$$

Solution. We have seen in class that we can write

$$A = \sum_{i<r} \sigma_i \left|u_i\right\rangle \left\langle v_i\right|.$$

Now

$$\begin{aligned}
A \cdot (1 - \left|v_0\right\rangle \left\langle v_0\right|) &= \sum_{i<r} \left(\sigma_i \left|u_i\right\rangle \left\langle v_i\right| \cdot (1 - \left|v_0\right\rangle \left\langle v_0\right|)\right) \\
&= \sum_{i<r} \left(\sigma_i \left|u_i\right\rangle \left\langle v_i\right| - \sigma_i \left|u_i\right\rangle \left\langle v_i \,\middle|\, v_0\right\rangle \left\langle v_0\right|\right) \\
&= \left(\sigma_0 \left|u_0\right\rangle \left\langle + \right| v_0 - \sigma_0 \left|u_0\right\rangle 1 \left\langle v_0\right|\right) + \\
&\quad \sum_{1 \le i < r} \left(\sigma_i \left|u_i\right\rangle \left\langle v_i\right| - \sigma_i \left|u_i\right\rangle 0 \left\langle v_0\right|\right),
\end{aligned}$$

by commutativity of matrix multiplication, as $\left\langle v_i \,\middle|\, v_0\right\rangle$ is 0 for $i \neq 0$ and for $i = 0$ it is 1.

So we obtain

$$A \cdot (1 - |v_0\rangle \langle v_0|) = \sum_{1 \le i < r} \sigma_i |u_i\rangle \langle v_i|$$

and hence by the remark before the exercise, $|v_1\rangle$ is the first singular vector of the new matrix. ○

$\boxed{4}$ (ii) Write a Python routine `first` that for any matrix A returns the first singular vector. You may use `numpy.linalg.svd()` for this.

Solution. Have a look at the extra file `08-programming_task.ipynb`. ○

$\boxed{3}$ (iii) Randomly generate a matrix for at least dimension $d = 4$ and sample size $n = 8$ (to get an $n \times d$-matrix). Apply this routine to the matrix to obtain the first singular vector and apply (i) to also obtain the second.

Solution. Have a look at the extra file `08-programming_task.ipynb`. ○

$\boxed{1}$ (iv) Compare numerically the second singular vector to the vector $|v_1\rangle$ from the SVD given by `numpy.linalg.svd()`.

Solution. The Python routine you wrote is probably correct if it consistently outputs the same vector as in the numpy method, at least up to sign. (Singular vectors are not unique, as you can choose to multiply them with -1, but apart from that they are usually uniquely determined for a randomly generated matrix.)

Beware that 1 in Python does not denote the identity matrix. ○

$\boxed{3}$ (v) With a method similar to (i), compute also the third singular vector and compare it to the SVD.

Solution. The method you can use here is computing

$$A \cdot (1 - |v_0\rangle \langle v_0|) \cdot (1 - |v_1\rangle \langle v_1|)$$

or equivalently

$$A \cdot (1 - |v_0\rangle \langle v_0| - |v_1\rangle \langle v_1|).$$ ○