Lab 2: Assignment 1 - Finding modes

Data structures and Algorithms for CL III

Anna Dick, Lea Grüner

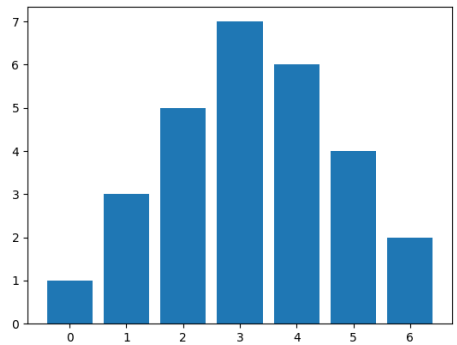
November 16, 2020

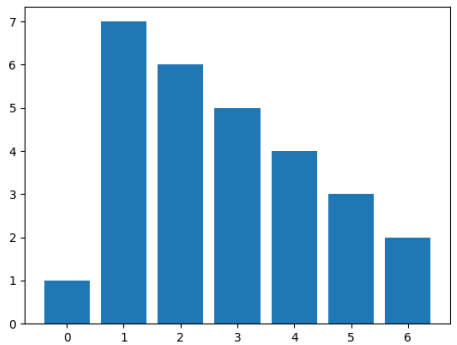
What is a mode?

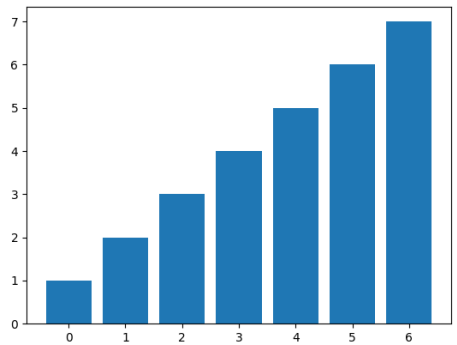
Mode: a peak in a sample
Element is a mode, if the element preceding and the following element are smaller
`if seq[i-1] < seq[i] and seq[i] > seq[i+1]`
1st element: if the element is greater than the following
Last element: if the element is greater than the preceding

Modes

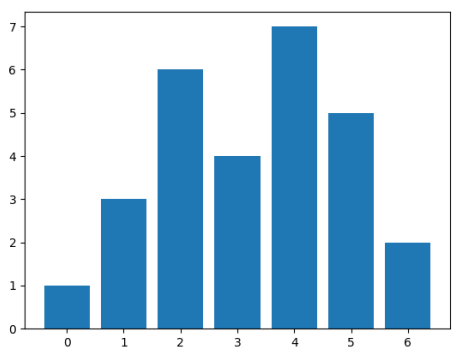
Unimodal:





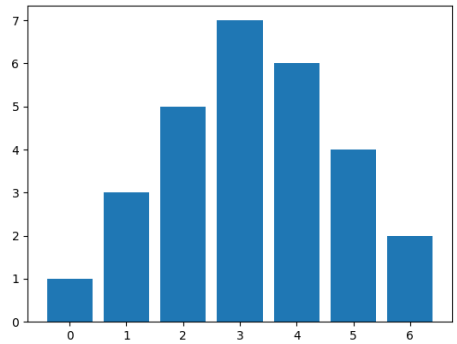


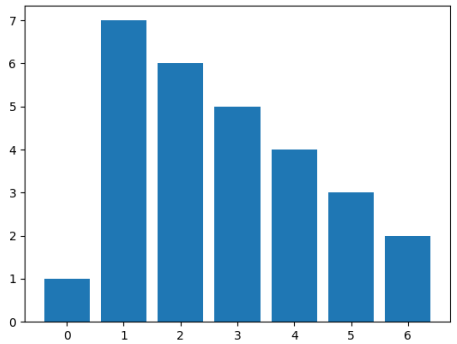
Bimodal:

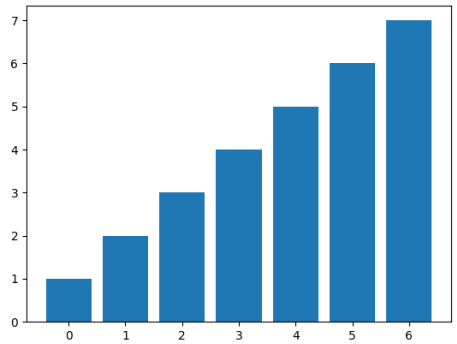


Modes in this assignment

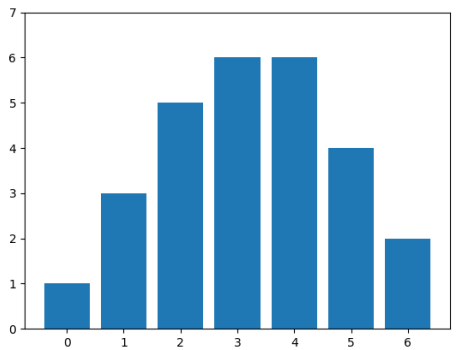
Unimodal without repeating values:







Not allowed:



Examples

Mode of a sequence `[1,3,5,7,6,4,2]` :
Value 7 at index 3

Mode of a sequence `[1,7,6,5,4,3,2]` :
Value 7 at index 1

Mode of a sequence `[1,2,3,4,5,6,7]` :
Value 7 at index 6

Modes in a 2D matrix

An element in a 2D matrix is a mode, if the elements in the left and right columns are smaller and the elements in the top and bottom rows are smaller

	0	1	2	3
0	0.00921	0.01661	0.01102	0.00269
1	0.04673	0.08427	0.05591	0.01364
2	0.08723	0.15730	0.10435	0.02547
3	0.05990	0.10802	0.07166	0.01749

Value 0.15730 at index [2][1] is mode

	0	1	2	3
0	0.01870	0.00743	0.00109	0.00006
1	0.08601	0.03419	0.00500	0.00027
2	0.14551	0.05784	0.00846	0.00045
3	0.09056	0.03600	0.00526	0.00028

Value 0.14551 at index [2][0] is mode

Exercise 1.1

- Do a simple linear search over a sequence of unique items and return the index of any mode
- Does not have to be the maximum in the whole sequence, return the first index you find that fits the requirements of a mode
- Think about special cases (empty list, one item, first or last item is the mode,...)

Exercise 1.2

- Find the mode in a unimodal sequence using a more efficient algorithm than linear search
- Review the slides about linear vs. binary search

Exercise 1.3

- Calculate the average running time of a given search function over x random unimodal samples
- Recommended to use the libraries imported in the template
- Creating samples: uniform from numpy.random and norm from scipy.stats, concatenate two sorted sequences, it's just important that there is one mode and the samples are random each run
- Measuring the run time: use time.time() rather than timeit because we want a different random sample for each run, only start timing after the sample is created
- Test if your implementation of 1.2 runs faster than 1.1

Exercise 1.4

- Find a mode in a 2D matrix
- Best to use numpy arrays (np.array(input))
- Greedy hill climbing algorithm: take the available maximum and check if all neighbours are lesser (no need to worry about local or global maxima because distribution is assumed to be unimodal)
- Test using multivariant_normal from scipy.stats