



COMP1730/COMP6730

Programming for Scientists

Sequence types

Lecture outline

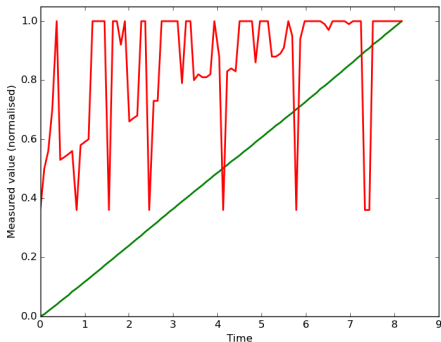
- * Sequence data types
- * Indexing, length & slicing

Problem: Sensor modelling

★ Time series of two measurements:

★ IR sensor
(% of range)

★ Tachometer
($1/360$ th rev.)



Sequences

- * A *sequence* contains zero or more values.
- * Each value in a sequence has a *position*, or *index*, ranging from 0 to $n - 1$.
- * The *indexing operator* can be applied to all sequence types, and returns the value at a specified position in the sequence.
 - Indexing is done by writing the index in square brackets after the sequence value, like so:
sequence[*pos*]

Sequence data types

- * python has three built-in sequence types:
 - strings (`str`) contain only text;
 - lists (`list`) can contain a mix of value types;
 - tuples (`tuple`) are like lists, but immutable.
- * Sequence types provided by other modules (but not covered here):
 - NumPy arrays (`numpy.ndarray`).

Indexing & length

sequence:	3.0	1.5	0.0	-1.5	-3.0
index:	0	1	2	3	4
	-5	-4	-3	-2	-1

- * In python, all sequences are indexed from 0.
- * The index must be an integer.
- * python also allows indexing from the sequence end using negative indices, starting with -1.
- * The length of a sequence is the number of elements, *not* the index of the last element.

- * `len(sequence)` returns sequence length.
- * Sequence elements are accessed by writing the index in square brackets, `[]`.

```
>>> x = [3, 1.5, 0, -1.5, -3]
```

```
>>> x[1]
```

```
1.5
```

```
>> x[-1]
```

```
-3.0
```

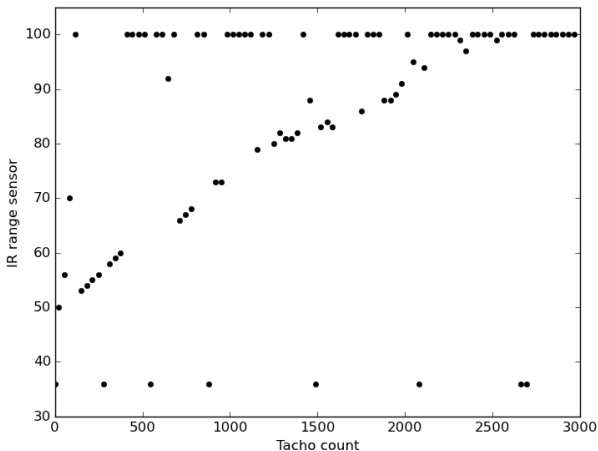
```
>>> len(x)
```

```
5
```

```
>>> x[5]
```

```
IndexError: index 5 is out of bounds  
for axis 0 with size 5
```

- * Sensor modelling: Is there a linear relation between x and y ?





- ★ Fit a straight line ($y = ax + b$) as close to all of the points as possible.
 - This can be done by solving a least-squares optimisation problem.
 - Simpler idea: Calculate the average slope between pairs of (adjacent) points.
- ★ Need to remove or ignore “outliers”.
- ★ Calculate residuals ($r_i = y_i - (ax_i + b)$) and check if they are normally distributed.

Functions on sequences

There are many built-in functions that operate on sequences:

- * `min` and `max` return the smallest and largest elements in the sequence.
- * `sum` returns the sum of the elements in the sequence.
- * `len` returns the number of elements in the sequence.
- * `sorted` returns a list with the elements of the sequence arranged in ascending order.
- * `x in sequence` returns `True` iff `x` is an element of the sequence.

Generalised indexing

- * Most python sequence types support *slicing* – accessing a subsequence by indexing a range of positions:

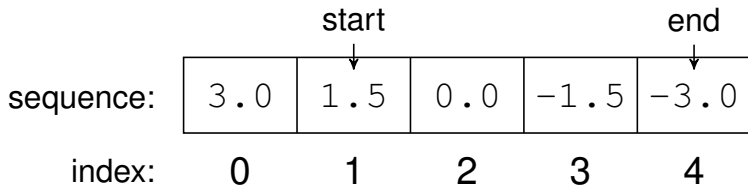
```
sequence[start:end]
```

```
sequence[start:end:step-size]
```

Slicing

- * The slice range is “half-open”: start index is included, end index is one after last included element.

```
>>> x = [3, 1.5, 0, -1.5, -3]
>>> x[1:4]
[ 1.5, 0, -1.5]
```



Slicing is an operator

- * The slicing operator returns a sequence, which can be indexed (or sliced):

```
>>> x[1:4][1]  
0.0
```

- * Slicing associates to the left.

Indexing vs. Slicing

- * Indexing a sequence returns an element: The index must be valid (i.e., between 0 and `length-1` or `-1` and `-length`).
- * Slicing returns a subsequence of the same type: Indexes in a slice do not have to be valid. And a slice may contain 0 or more elements.

Take home message

- * `list` data type to store an (ordered) sequence of values.
- * Sequence index starts from 0, not 1!
- * Indexing operator returns an element, whereas slicing operator returns a sequence.