SeqBim 2022

`fimpera`: drastic improvement of Approximate Membership Query data-structures with counts

Lucas Robidou, Pierre Peterlongo

2022-11-17

Inria Rennes

context

Some context

My dream:

- to index (large) genomic datasets
  - SRA is now about 42 PB
  - (6 months ago: about 36 PB)
- to query those indexed datasets
  - using k-mers
  - instantly
  - no hash table scales, so we use an AMQ
  - but it introduces some false positives

Some context

My dream:

- to index (large) genomic datasets
  - SRA is now about 42 PB
  - (6 months ago: about 36 PB)
- to query those indexed datasets
  - using k-mers
  - instantly
  - no hash table scales, so we use an AMQ
  - but it introduces some false positives

Camille Marchet, Christina Boucher, Simon J Puglisi, Paul Medvedev, Mikaël Salson, and Rayan Chikhi. **Data structures based on k-mers for querying large collections of sequencing data sets.** Genome Research, 31(1):1–12, 2021.

Some context

My dream:

- to index (large) genomic datasets
  - SRA is now about 42 PB
  - (6 months ago: about 36 PB)
- to query those indexed datasets
  - using k-mers
  - instantly
  - no hash table scales, so we use an AMQ
  - but it introduces some false positives
  - my mission, should I decide to accept it: **locate and terminate those false positives**

Camille Marchet, Christina Boucher, Simon J Puglisi, Paul Medvedev, Mikaël Salson, and Rayan Chikhi. **Data structures based on k-mers for querying large collections of sequencing data sets.** Genome Research, 31(1):1–12, 2021.

Challenges

- indexation time
- **abundance storage**
- **index size**
- **query time**
- **false positive rate**

fimpera

Main idea of fimpera

How to reduce the false positive rate ?

Let's consider the 13-mer 'datastructure'. Its 11-mers are:

- 'datastructu' (abundance: 5)
- 'atastructur' (abundance: 2)
- 'tastructure' (abundance: 2)

$\Longrightarrow$ abundance of 'datastructure' can't be more than 2.

Note that increasing the abundance of one 11-mer will not change the above sentence.

Some notations

Rather than indexing k-mers, **let's index s-mers**, $s < k$.
Let's introduce $z = k - s$, so that a k-mer is made of $z + 1$ smaller s-mers.

A k-mer is said 'found' iif the $z + 1$ s-mers composing it are found in the filter.

This leads to an exponential decrease of the false positive rate wrt z.

Indexation with fimpera

- count k-mers

Indexation with fimpera

- count k-mers
- compute s-mers $s_{abundance}$ (max of count of k-mers containing this s-mer)

Indexation with fimpera

- count k-mers
- compute s-mers $s_{abundance}$ (max of count of k-mers containing this s-mer)
- index s-mers along their $s_{abundance}$ in an AMQ $(*)$

Indexation with fimpera

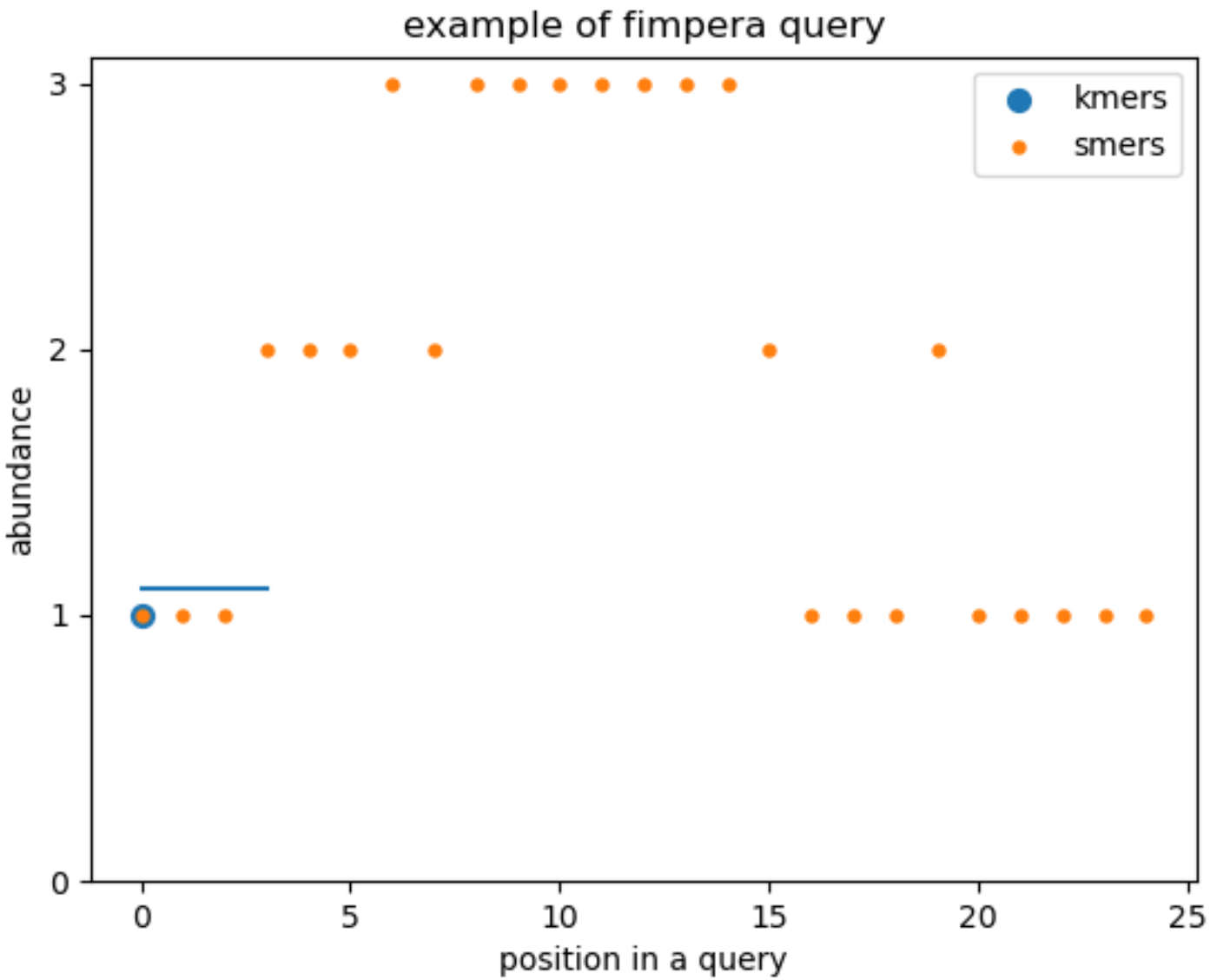
- count k-mers
- compute s-mers $s_{abundance}$ (max of count of k-mers containing this s-mer)
- index s-mers along their $s_{abundance}$ in an AMQ $(*)$

$(*)$ e.g. counting Bloom filter

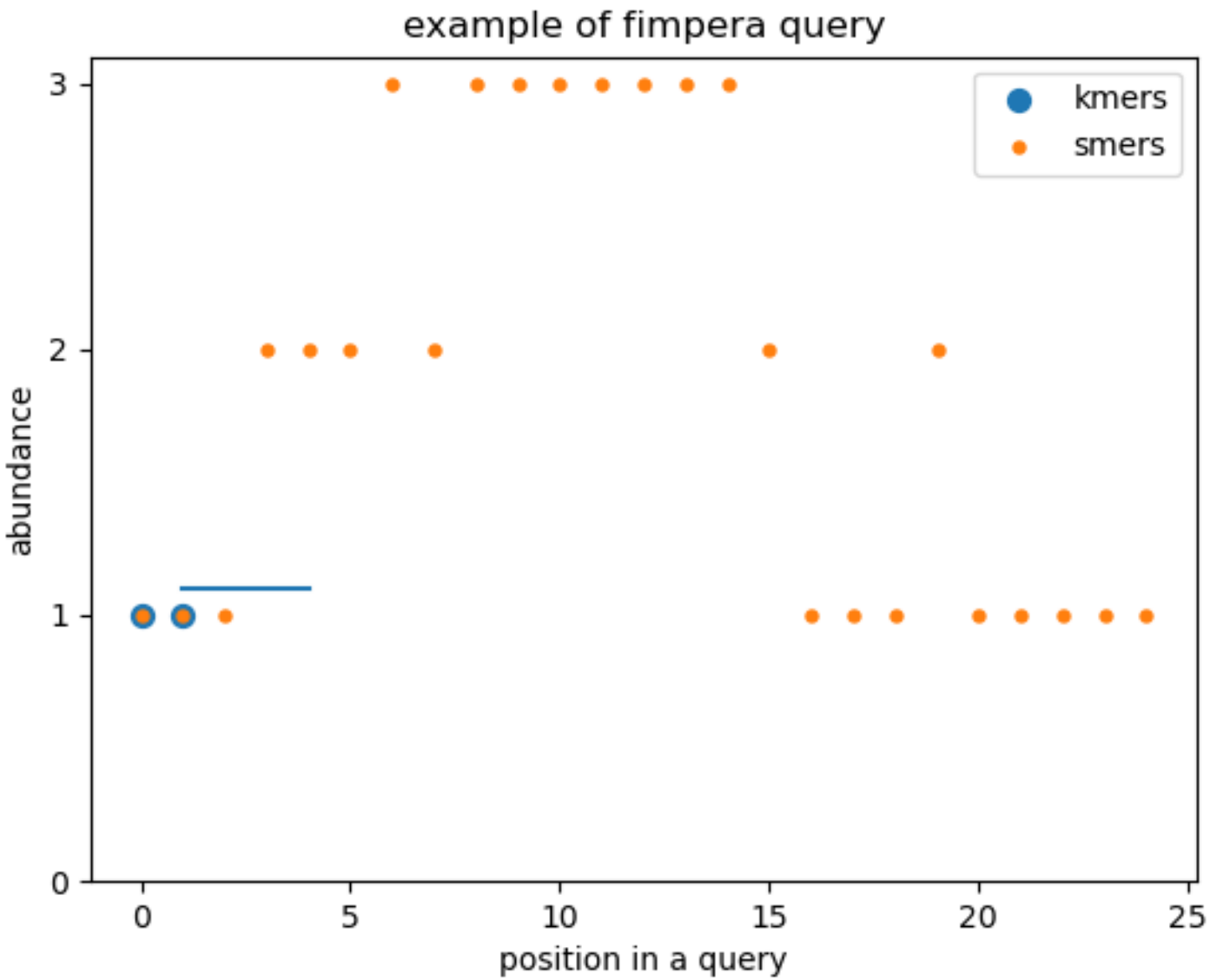
Query on fimpera

- query abundance of every s-mers
- compute k-mers abundance (minimum of its s-mers $s_{abundance}$)

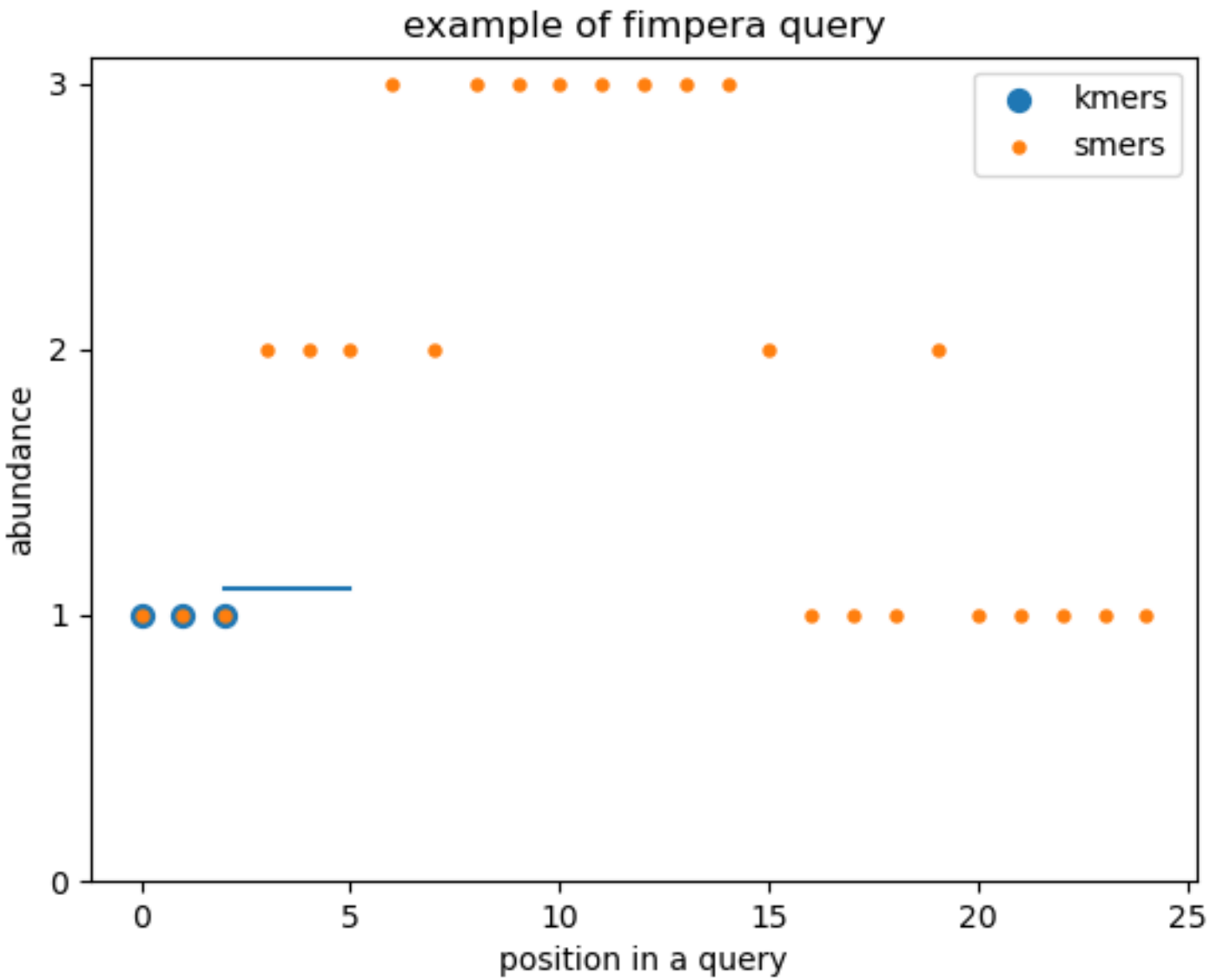
Query on fimpera



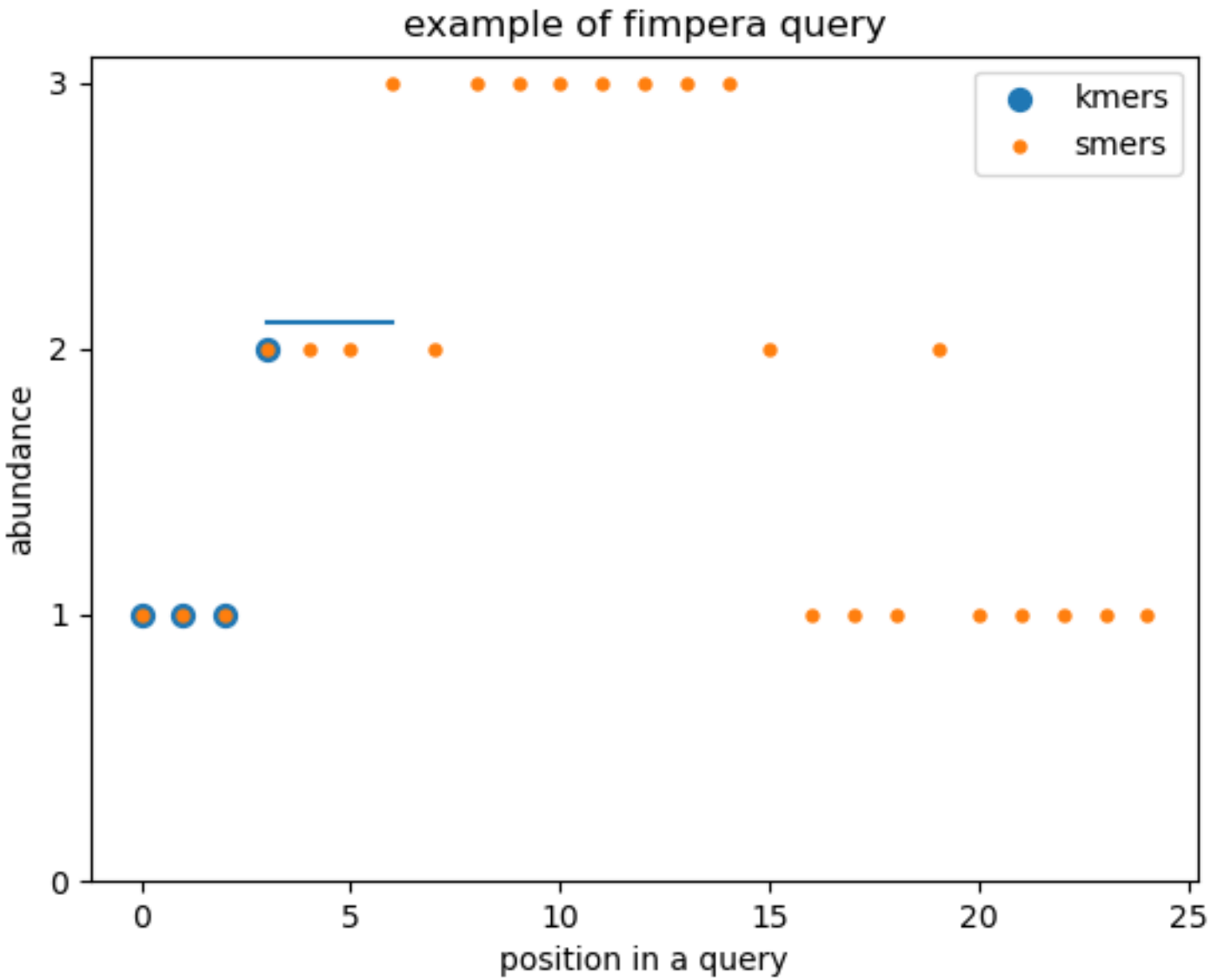
Query on fimpera



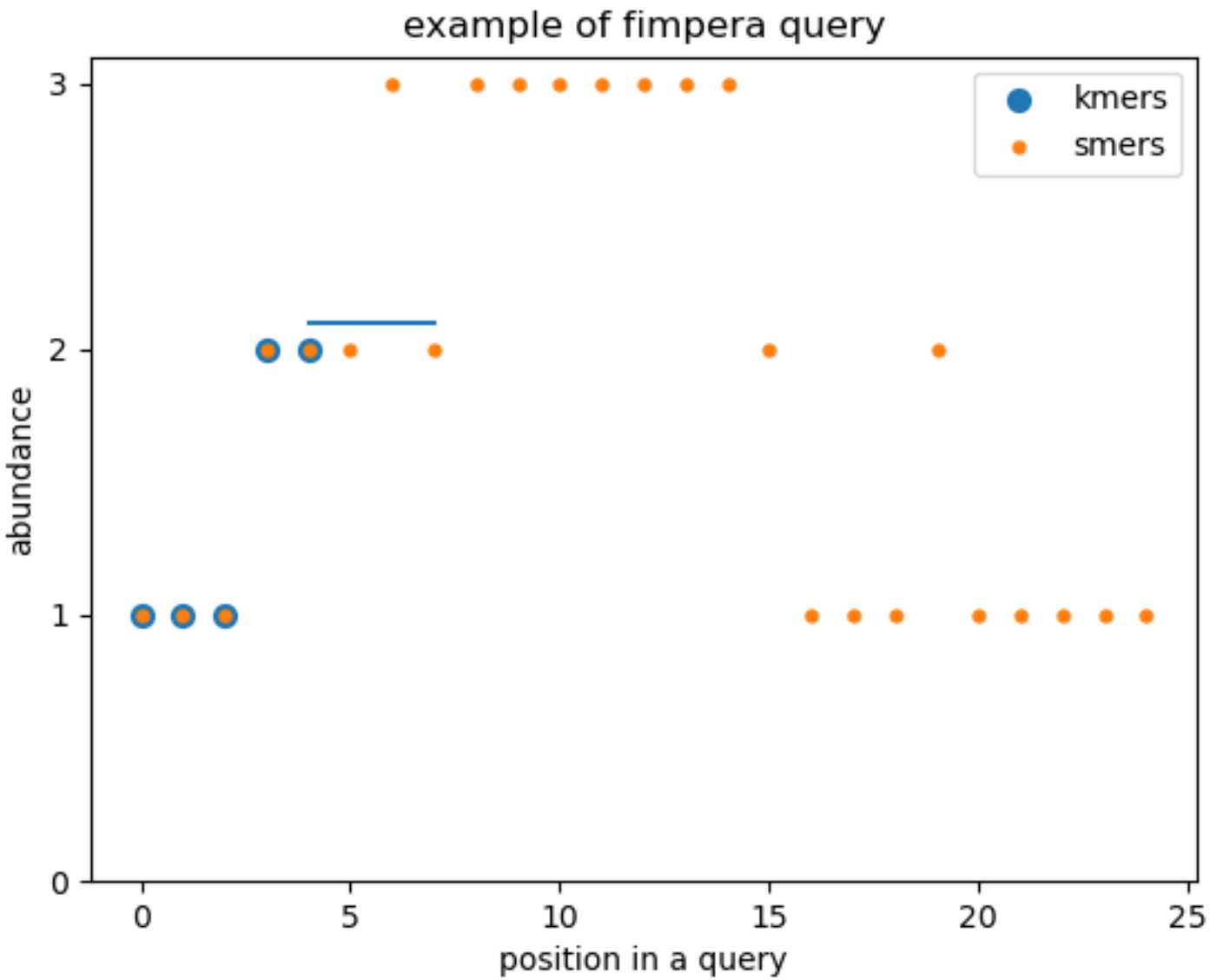
Query on fimpera



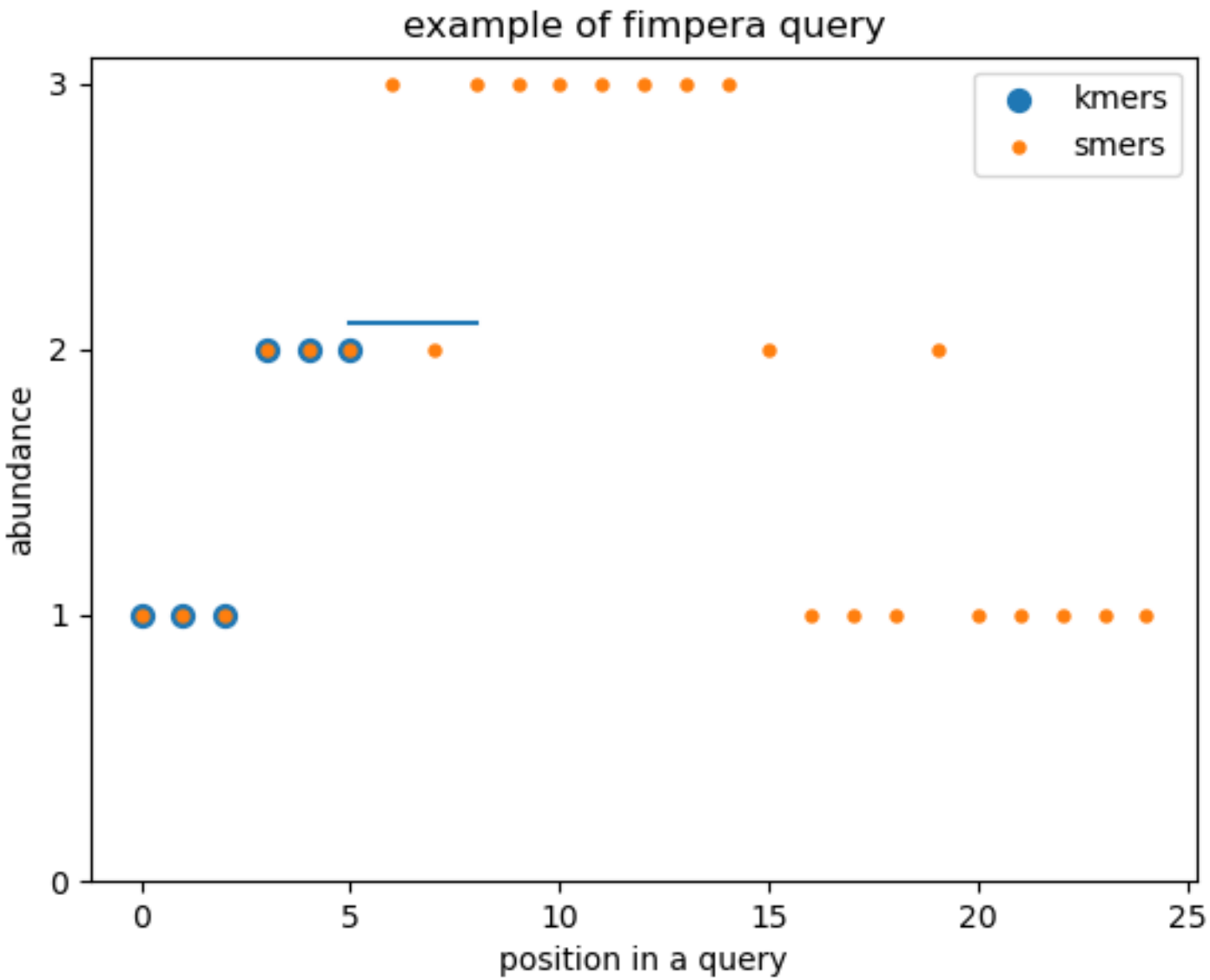
Query on fimpera



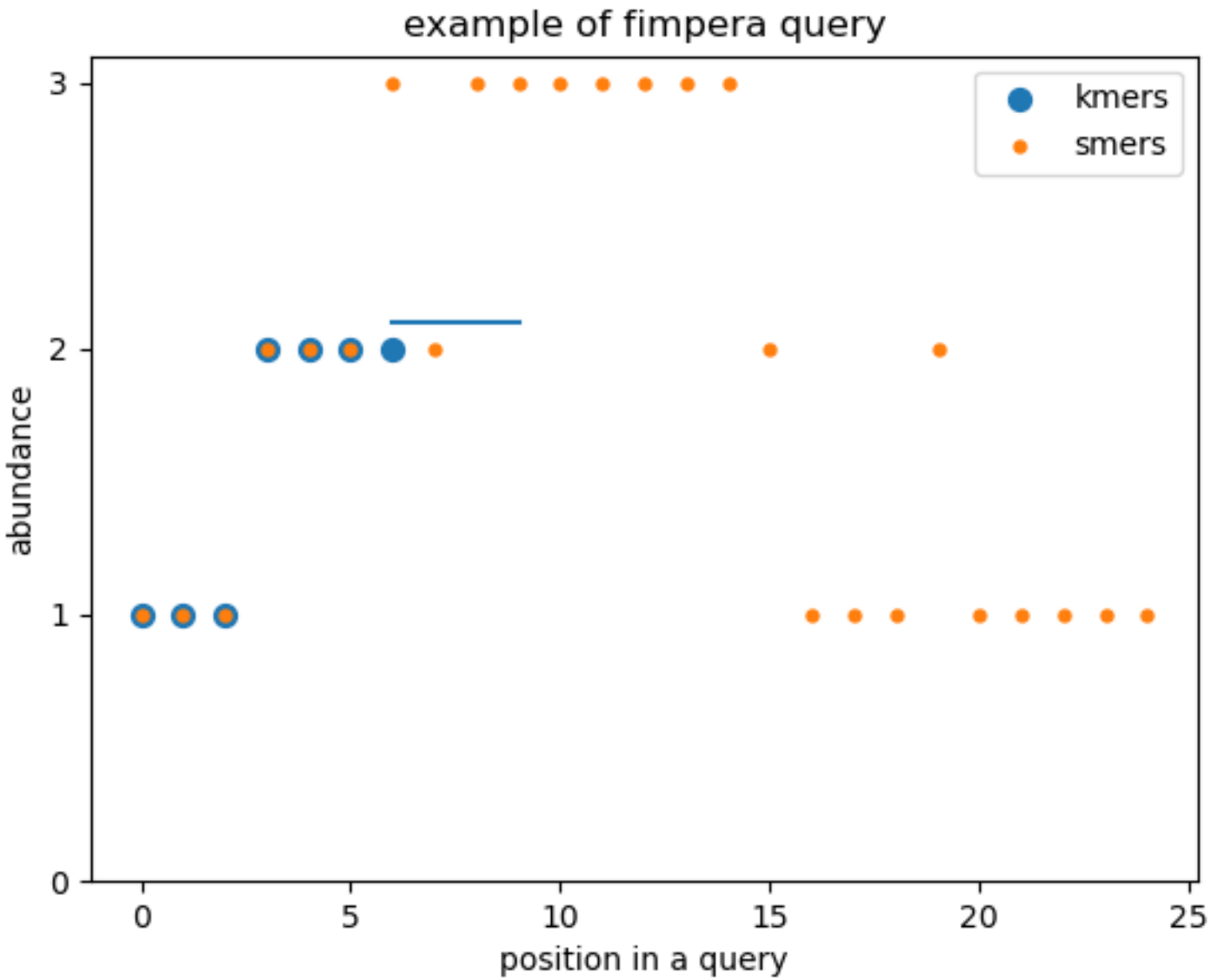
Query on fimpera



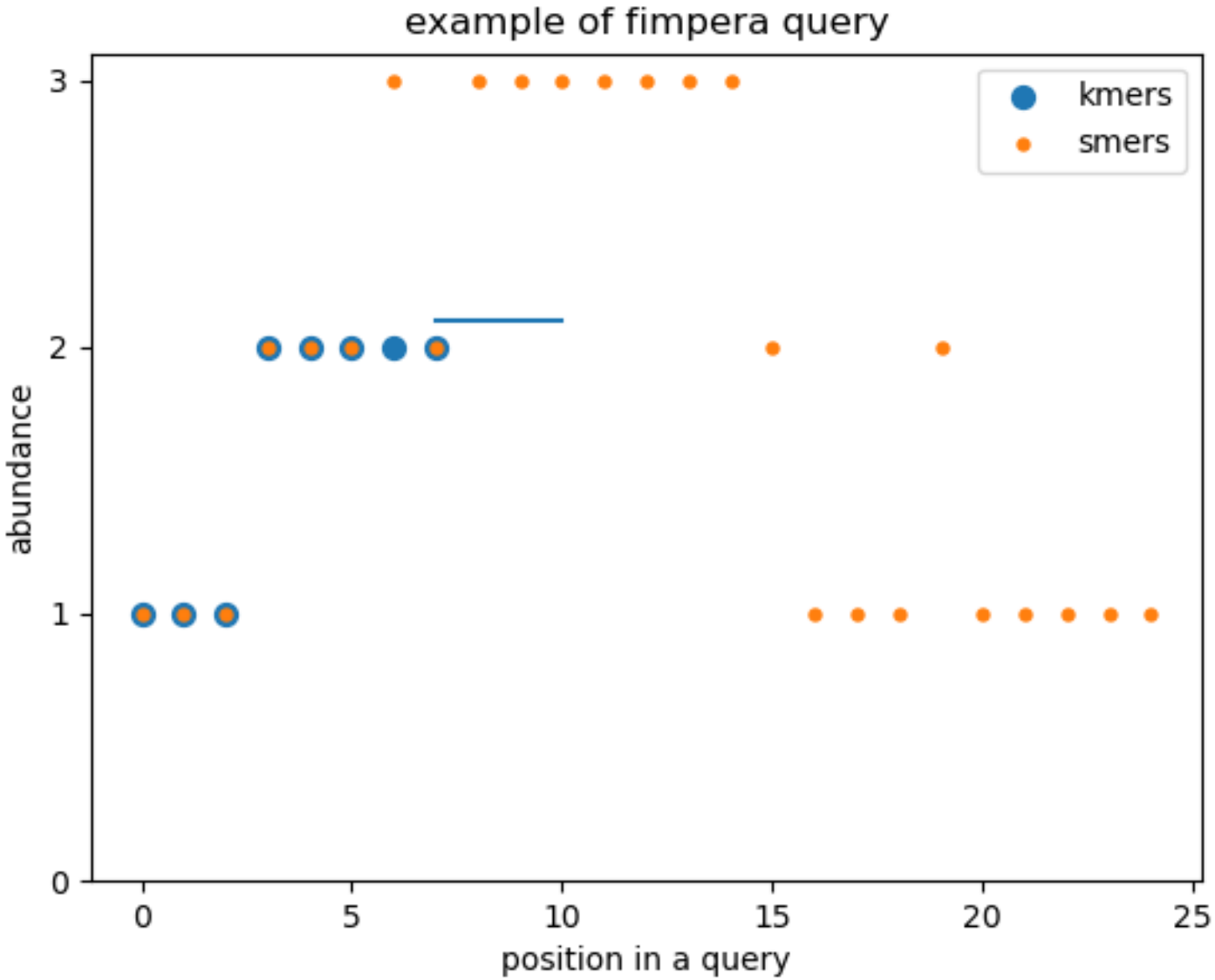
Query on fimpera



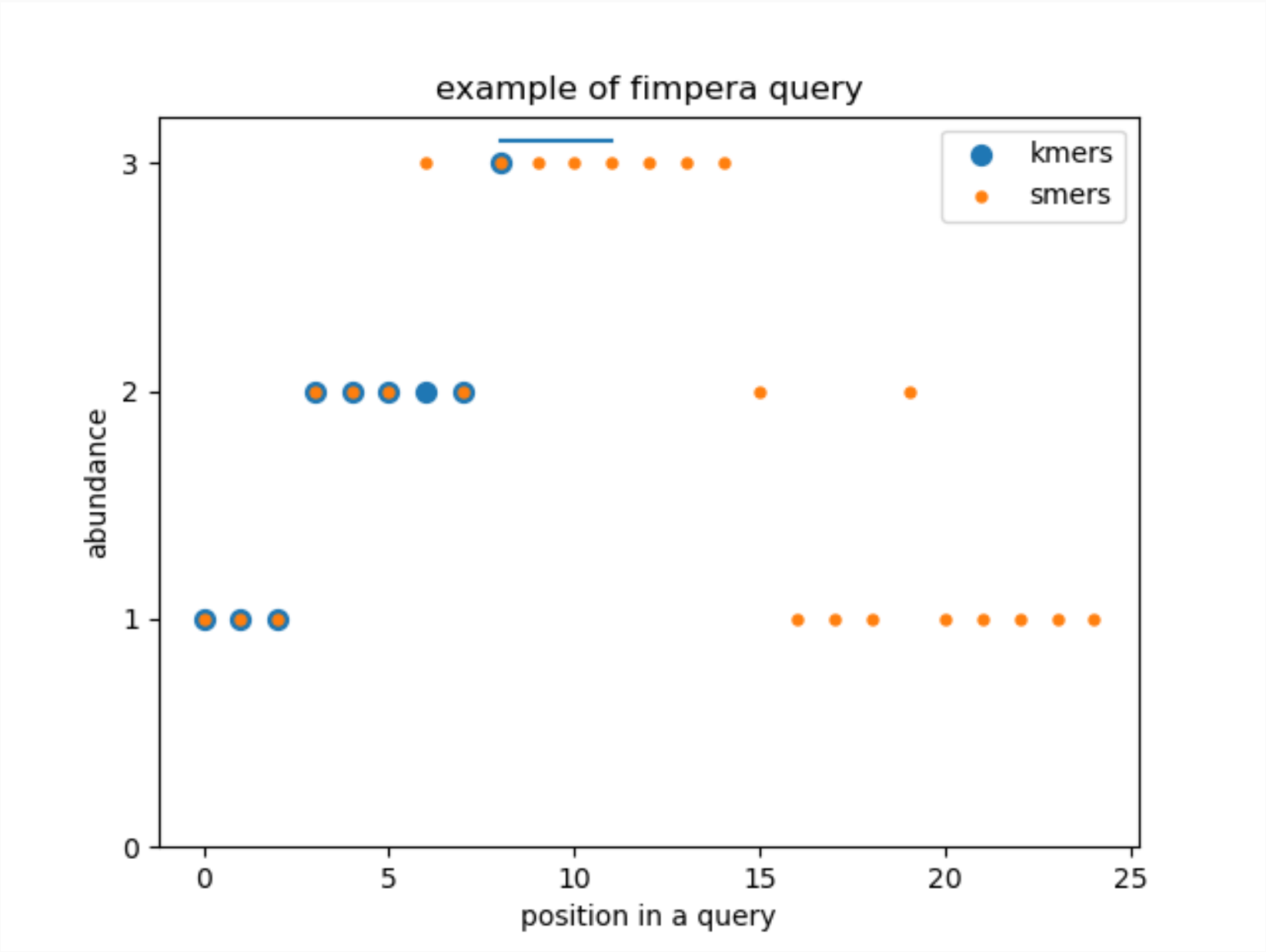
Query on fimpera



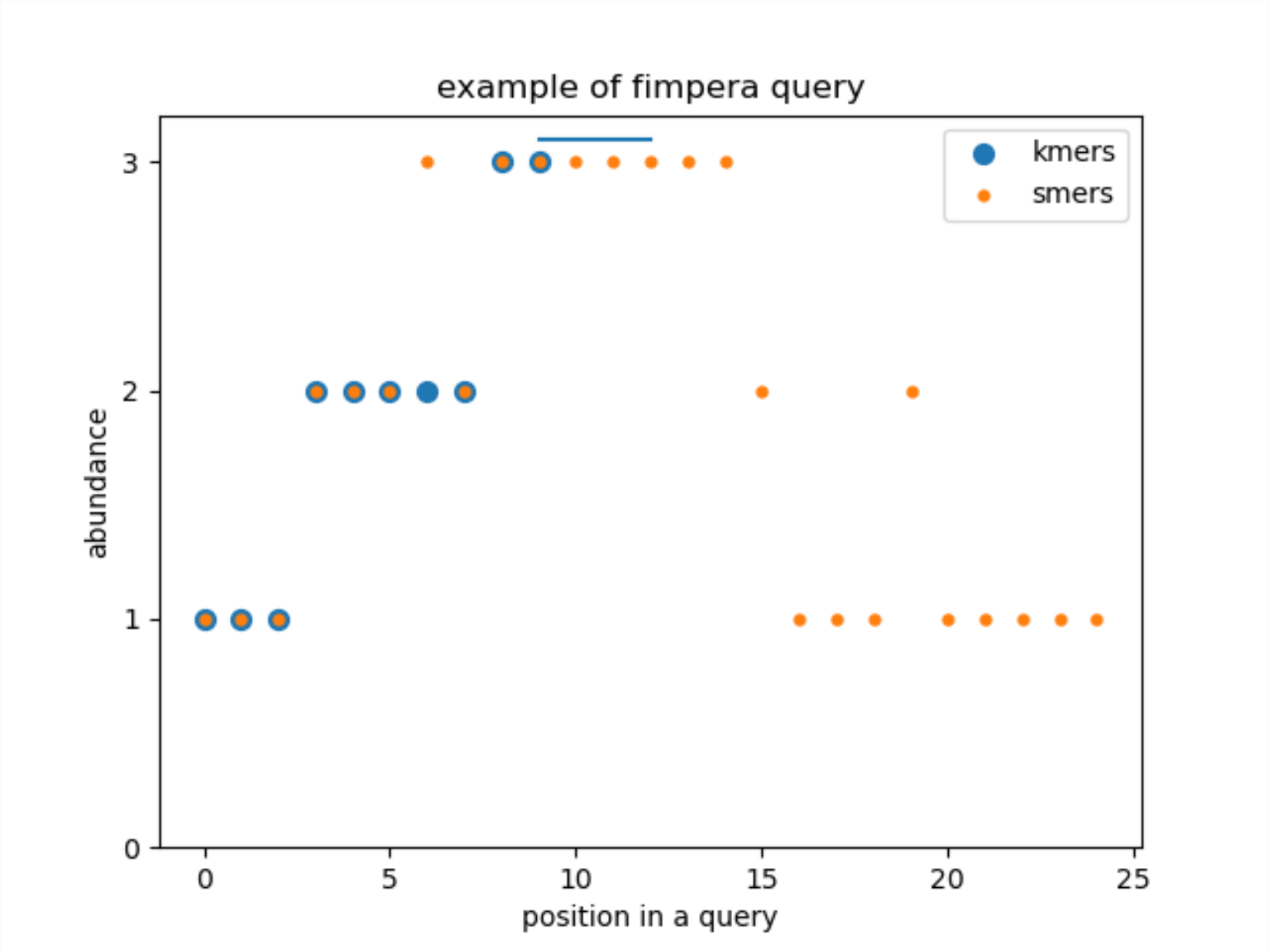
Query on fimpera



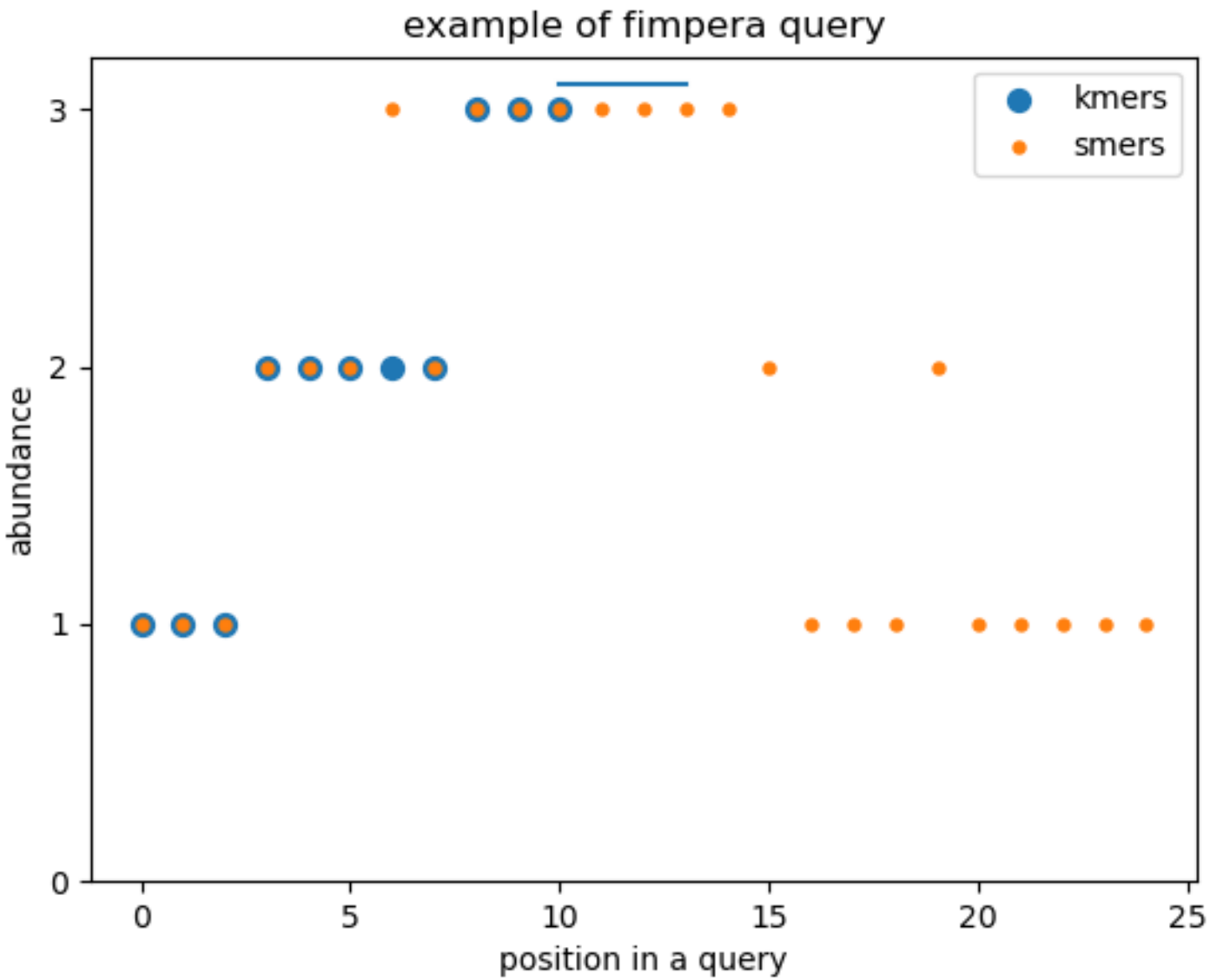
Query on fimpera



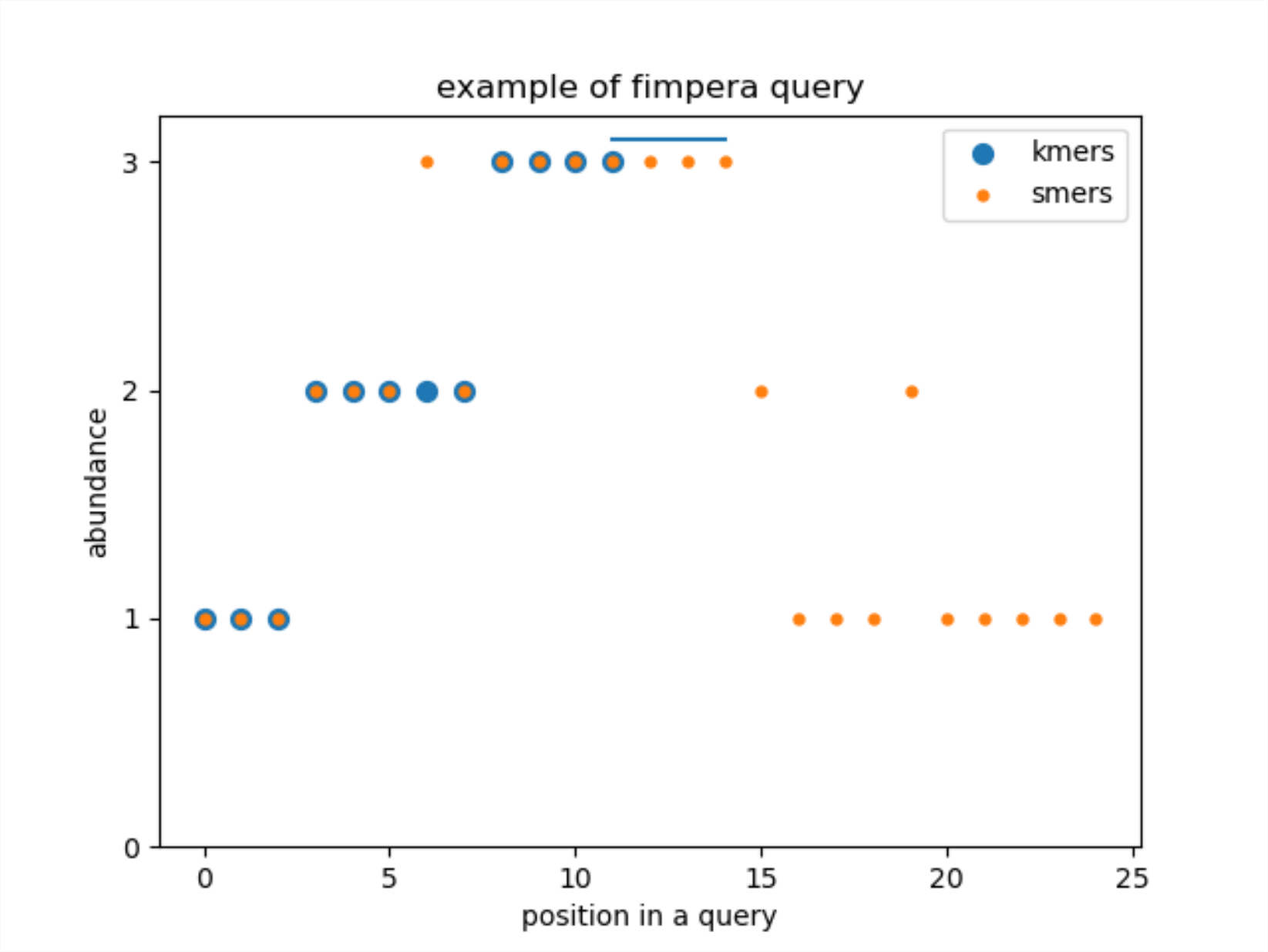
Query on fimpera



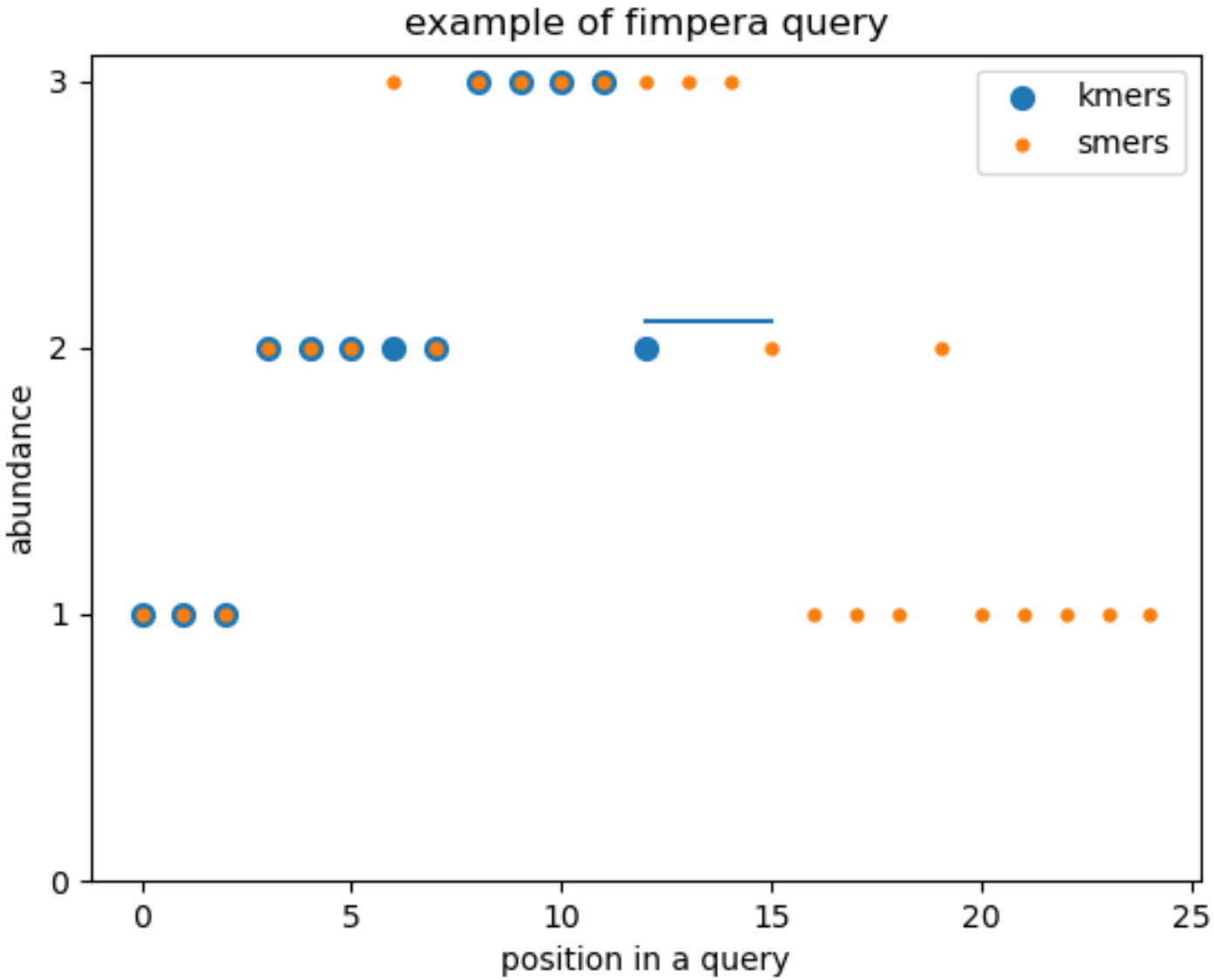
Query on fimpera



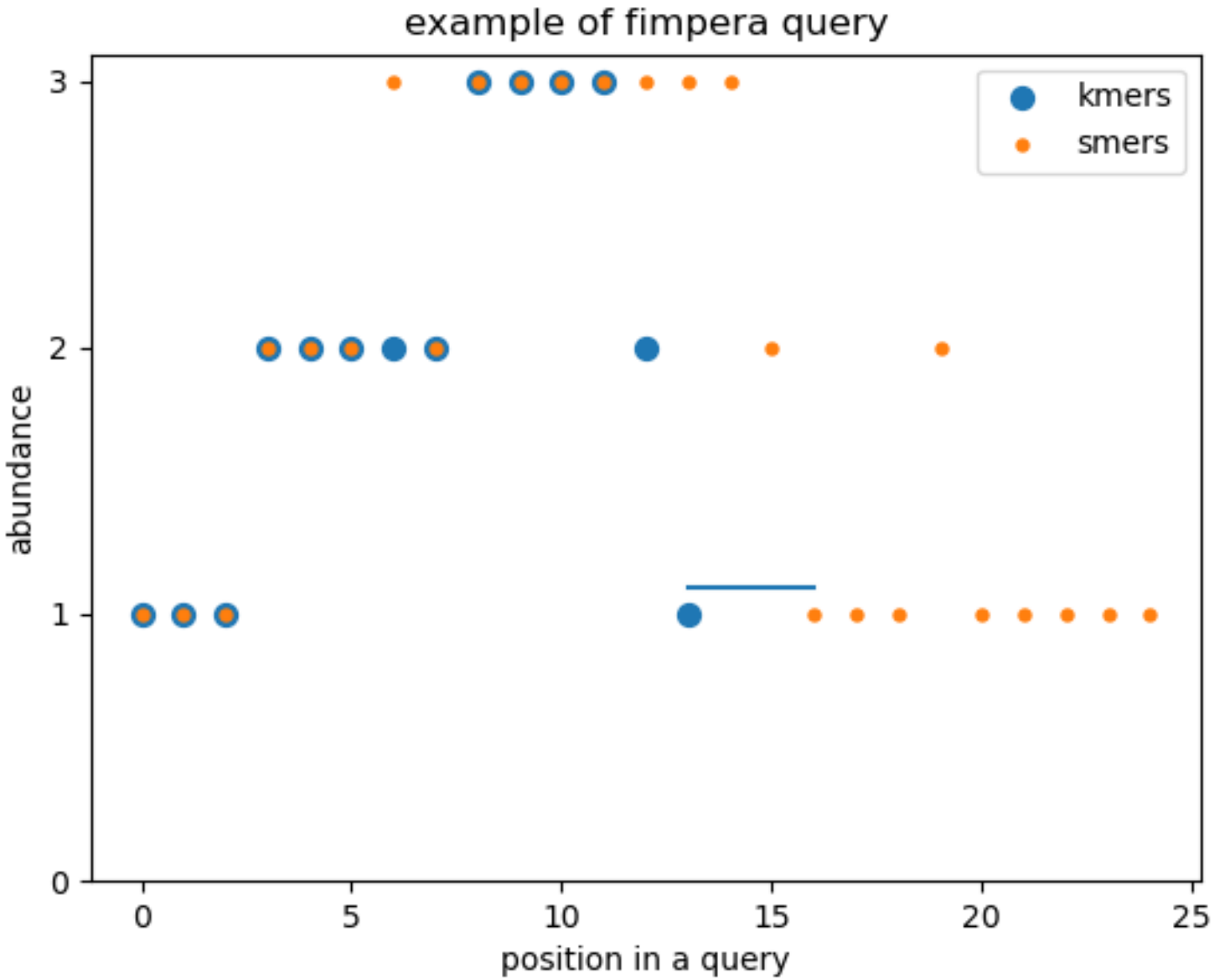
Query on fimpera



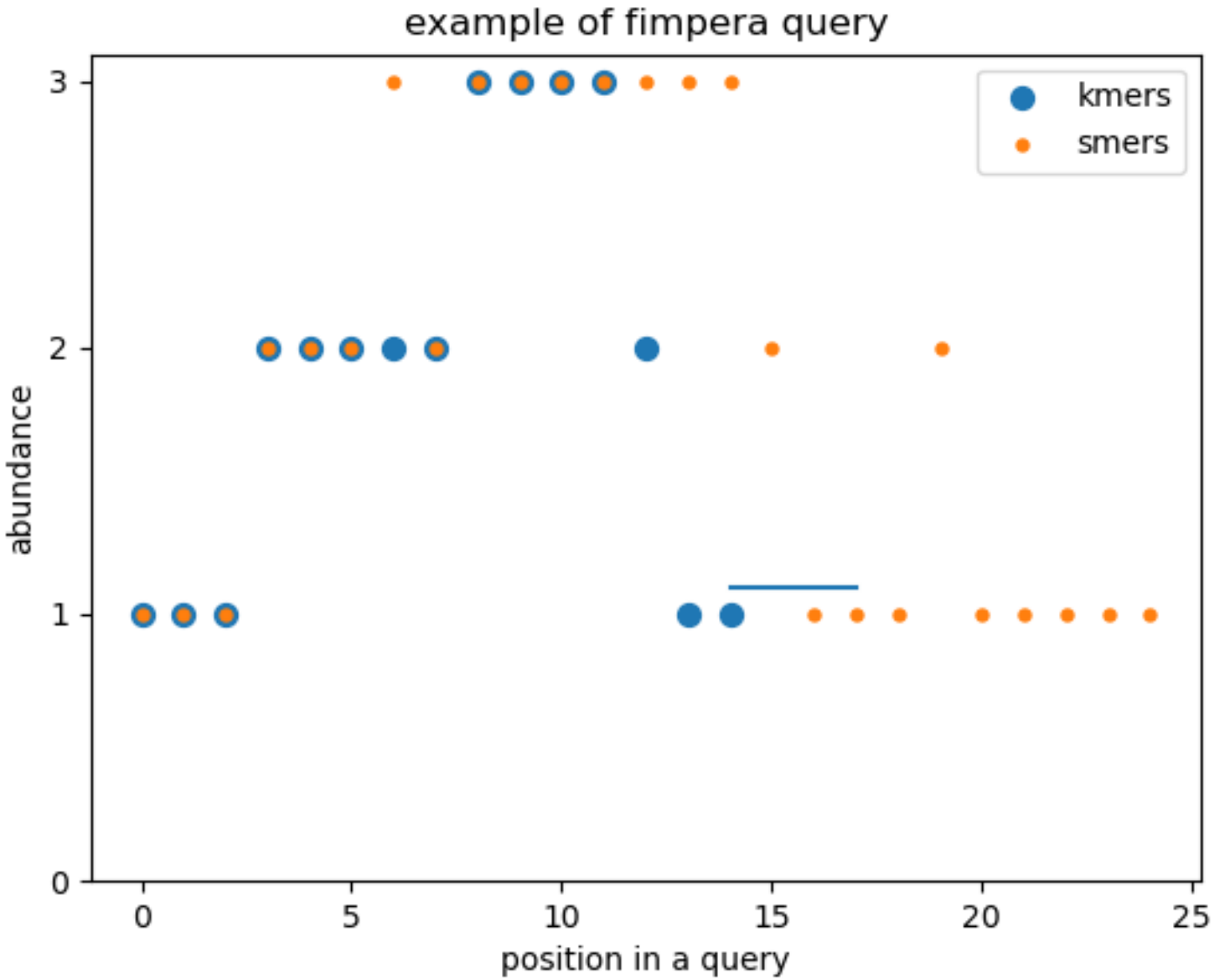
Query on fimpera



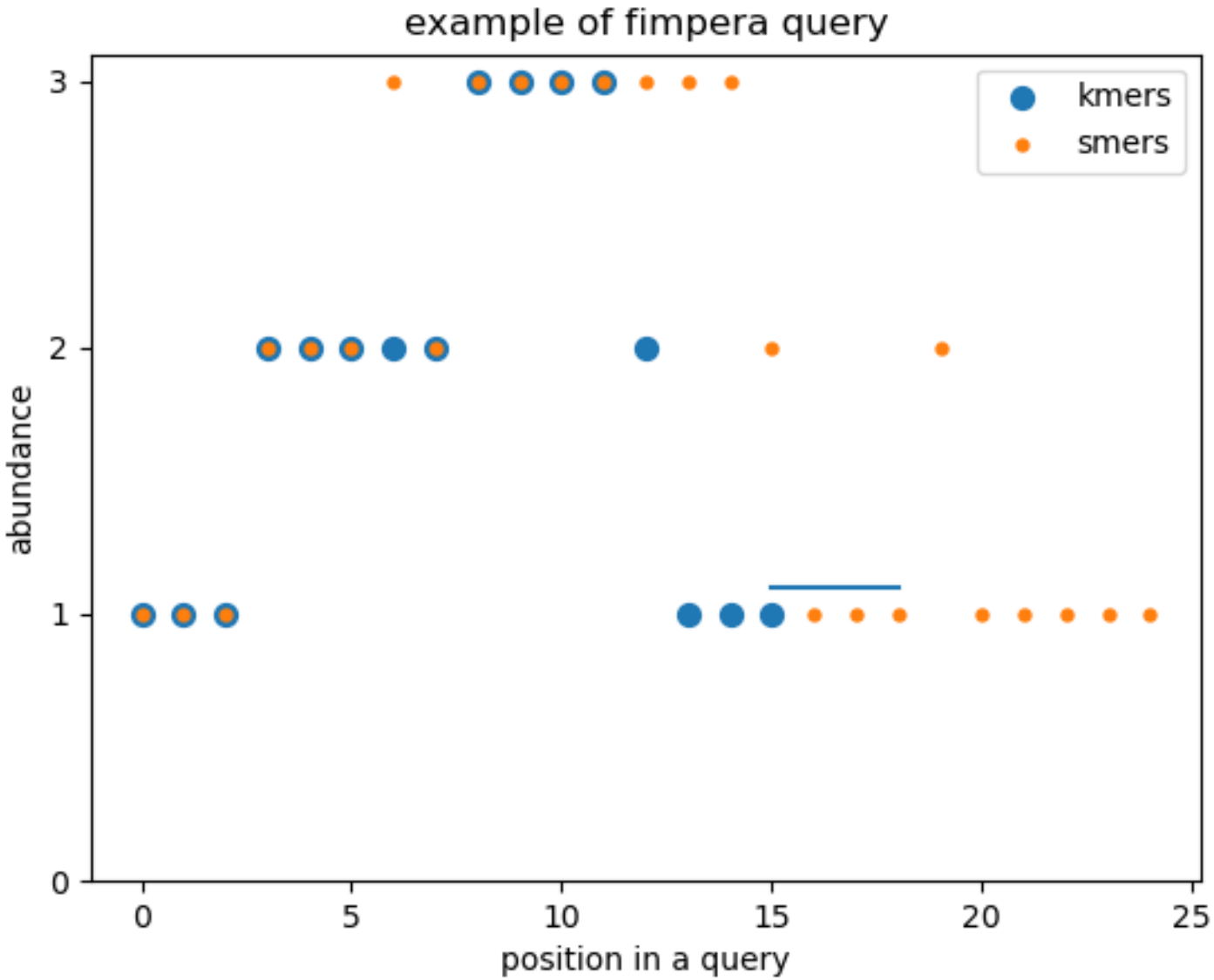
Query on fimpera



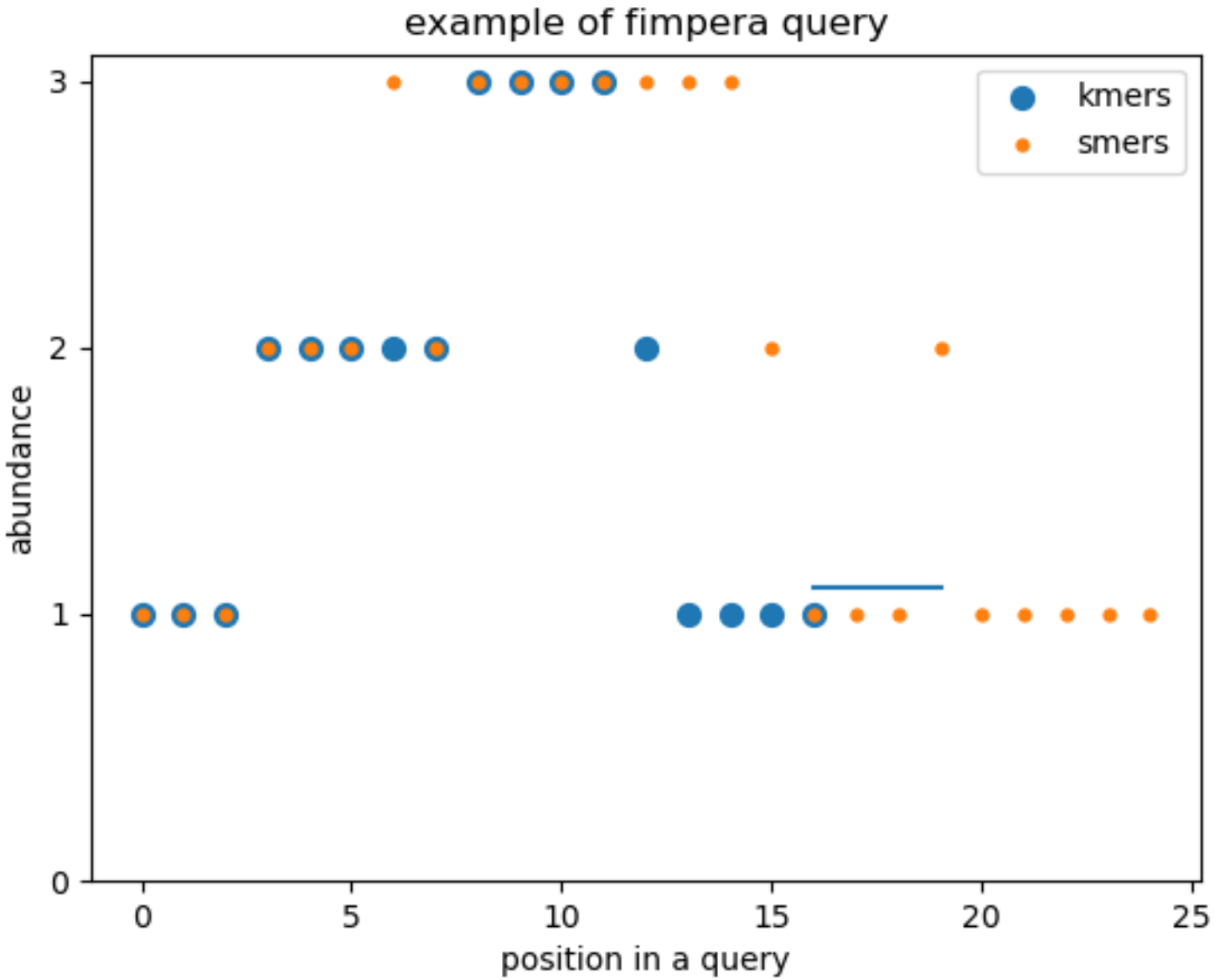
Query on fimpera



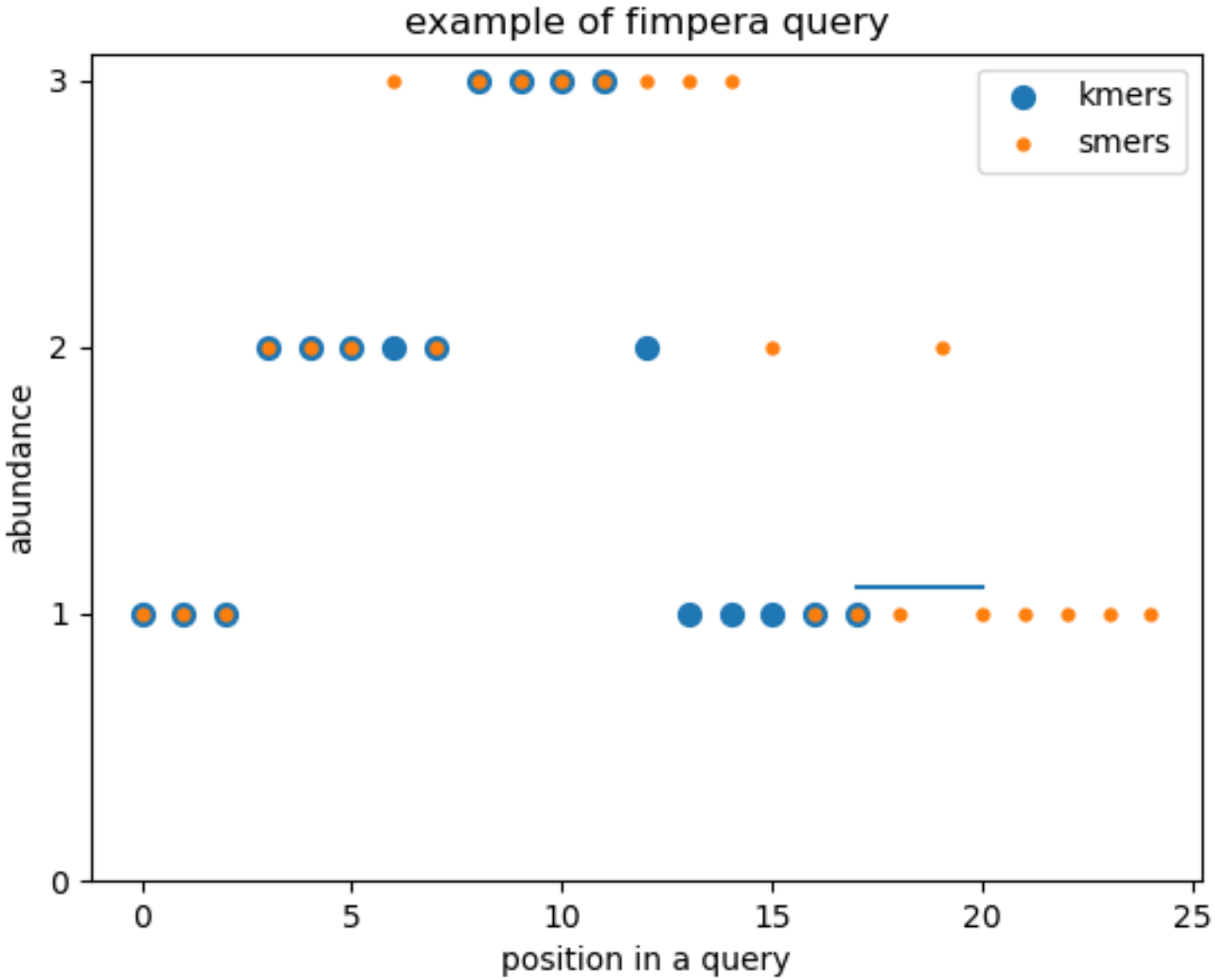
Query on fimpera



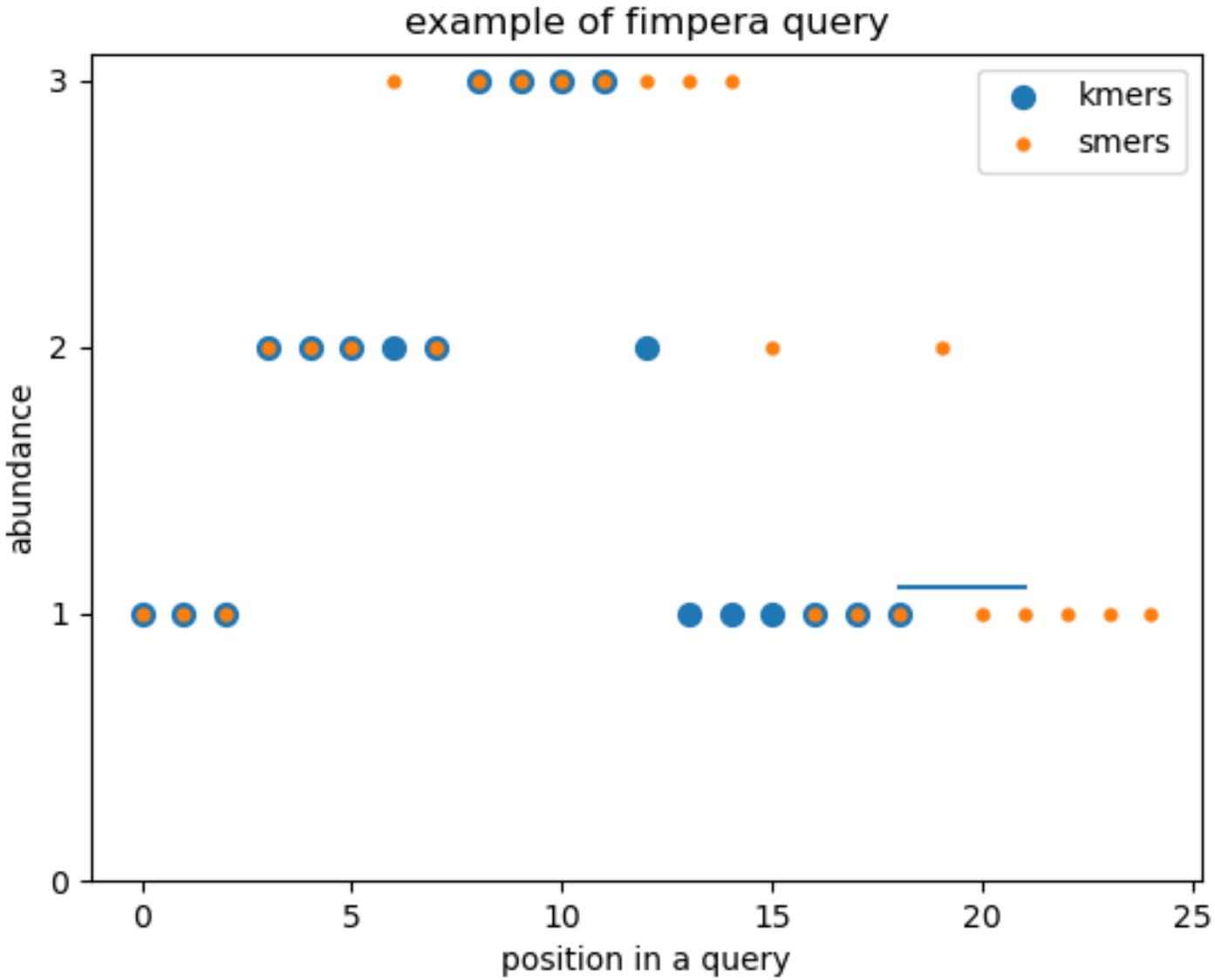
Query on fimpera



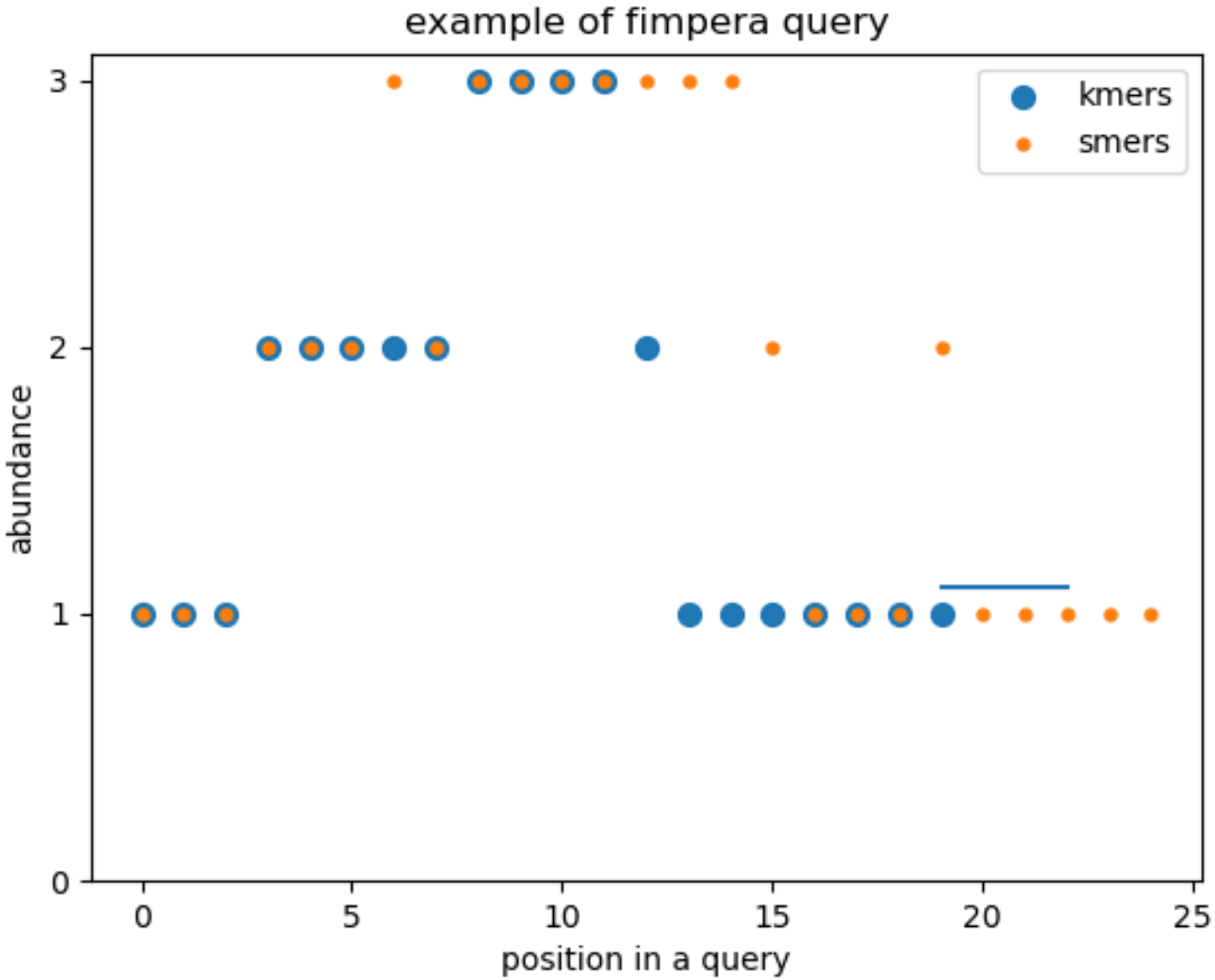
Query on fimpera



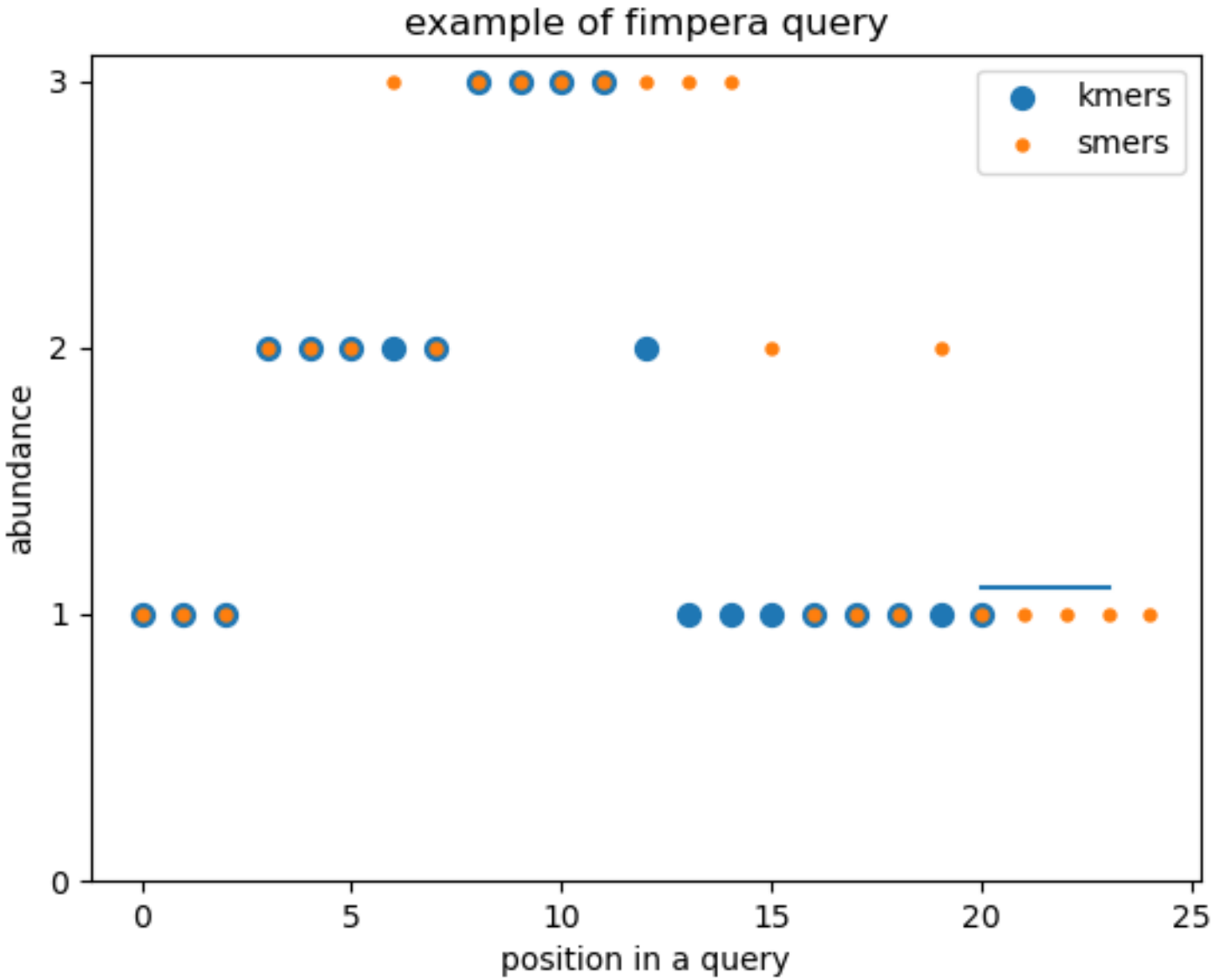
Query on fimpera



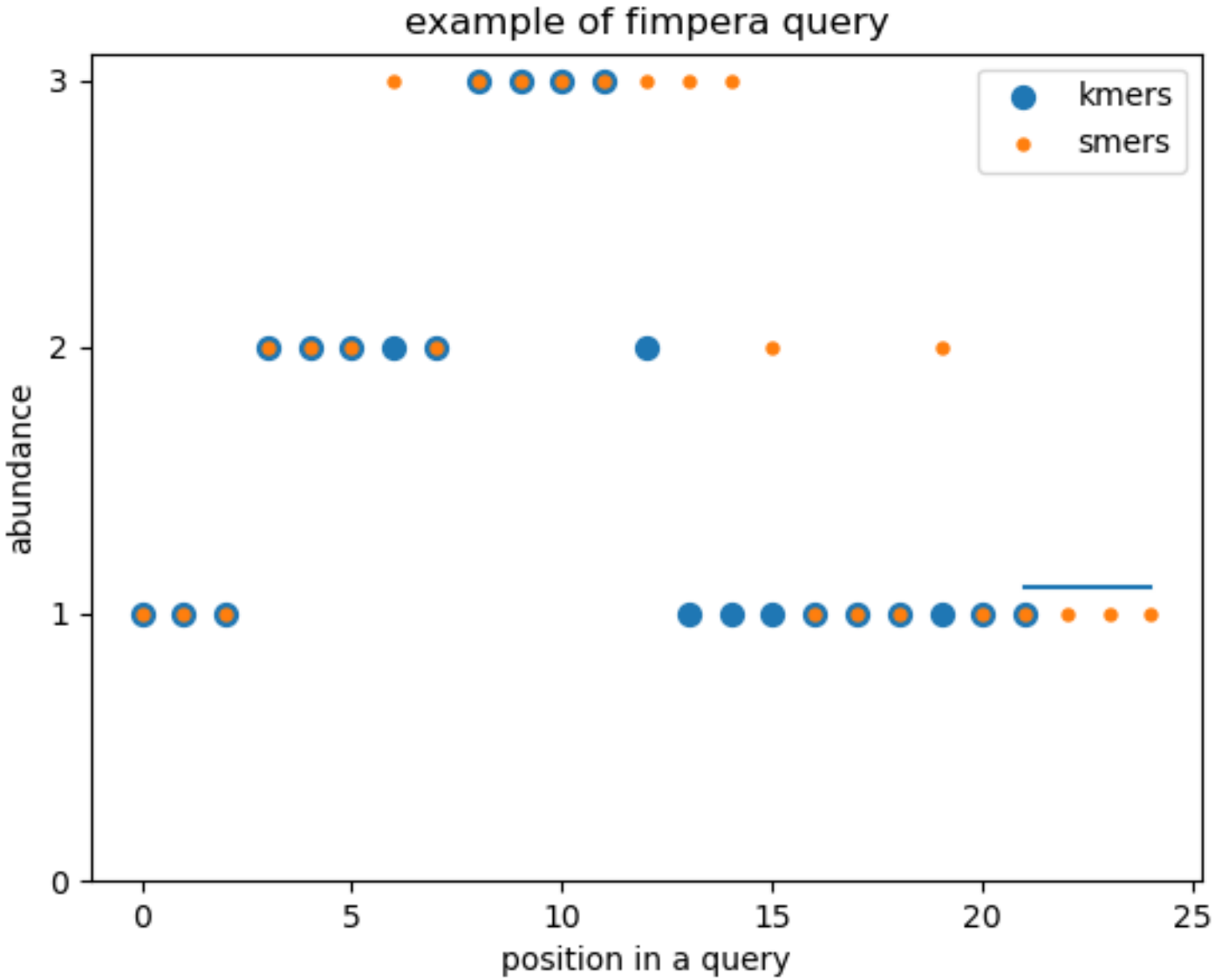
Query on fimpera



Query on fimpera



Query on fimpera



limitation of fimpera

For a chosen k, if z is too high, then fimpera will index and query very small s-mers. In such case, the probability of having indexed all those s-mers is *high*.

limitation of fimpera

For a chosen k, if z is too high, then fimpera will index and query very small s-mers. In such case, the probability of having indexed all those s-mers is *high*.

Typical value of k: 31
Typical value of s: 28

Example of construction overestimation

- indexing ‘ACTGAC’ with $s = 3$
- indexed s-mers include ‘GAC’, ‘ACT’ and ‘CTG’
- k-mer ‘GACTG’ would be found with the abundance of ‘ACTGAC’

(Note it is easy to do with 3-mers, but much more difficult to do with 28-mers!)

sliding minimum window

sliding minimum window

i	0	1	2	3	4	5	6	7	8	9	10	11
v	5	3	1	7	4	5	3	4	5	6	9	5
j	0					1				2		
min_left												
min_right												
min_sliding	1	1	1	3	3	3	3	4				

Table 1: Computation example of the *min_sliding* vector, with a window of size 5. Tables *min_left* and *min_right* are represented for helping the comprehension, but are not implicitly created in practice. The j row indicates the starting positions of the fixed windows.

sliding minimum window

i	0	1	2	3	4	5	6	7	8	9	10	11
v	5	3	1	7	4	5	3	4	5	6	9	5
j	0					1				2		
min_left	5	3	1	1	1	5	3	3	3	6	6	5
min_right	1	1	1	4	4	3	3	4	5	5	5	5
min_sliding	1	1	1	3	3	3	3	4				

Table 2: Computation example of the *min_sliding* vector, with a window of size 5. Tables *min_left* and *min_right* are represented for helping the comprehension, but are not implicitly created in practice. The j row indicates the starting positions of the fixed windows.

sliding minimum window

i	0	1	2	3	4	5	6	7	8	9	10	11
v	5	3	1	7	4	5	3	4	5	6	9	5
j	0					1				2		
min_left	5	3	1	1	1	5	3	$\underline{3}$	3	6	6	5
min_right	1	1	1	$\underline{4}$	4	3	3	4	5	5	5	5
min_sliding	1	1	1	$\underline{\mathbf{3}}$	3	3	3	4				

Table 3: Computation example of the *min_sliding* vector, with a window of size 5. Tables *min_left* and *min_right* are represented for helping the comprehension, but are not implicitly created in practice. The j row indicates the starting positions of the fixed windows.

results

Data used

- two fastq files from the TARA ocean dataset (metagenomic)
- AHX_ACXIOSF_6_1_C2FGHACXX.IND4_clean.fastq is indexed
- BHN_AIAIOSF_1_1_C7CA4ACXX.IND13_clean.fastq's 1,000,000 firsts reads are queried
- k-mers seen less than twice are discarded as well as low complexity k-mers
- $k = 31$, $s = 26$

fimpera's effect on false positive rate

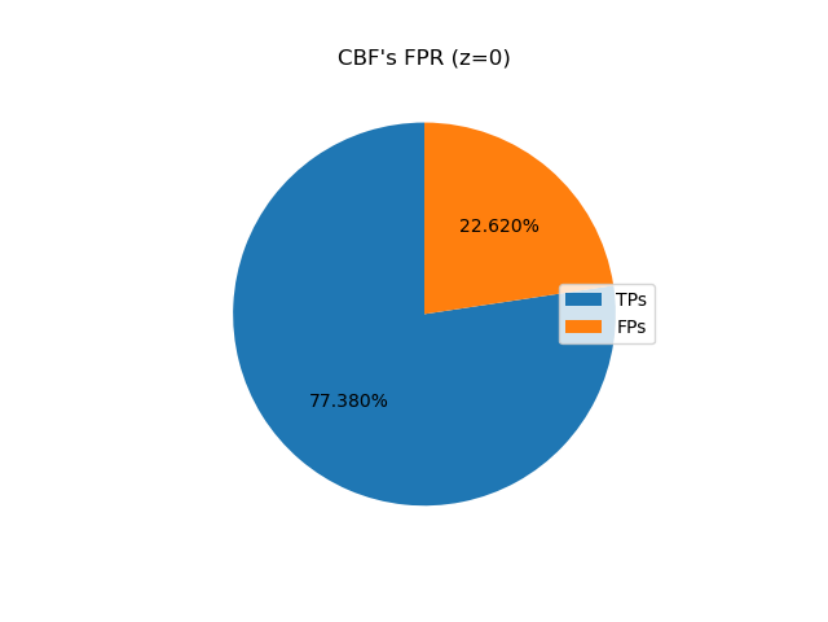


Figure 1: proportion of false positive calls **without** fimpera

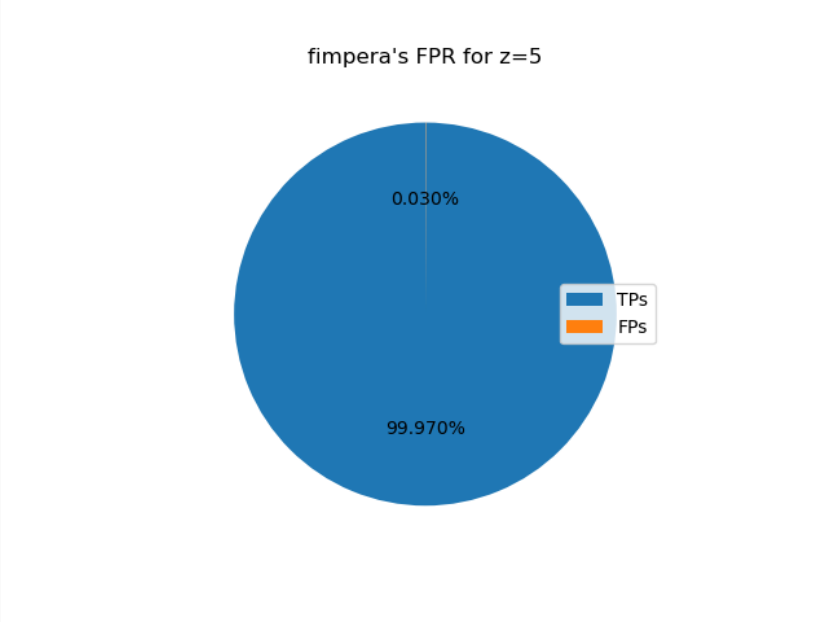


Figure 2: proportion of false positive calls **with** fimpera

fimpera's effect on abundance correctness

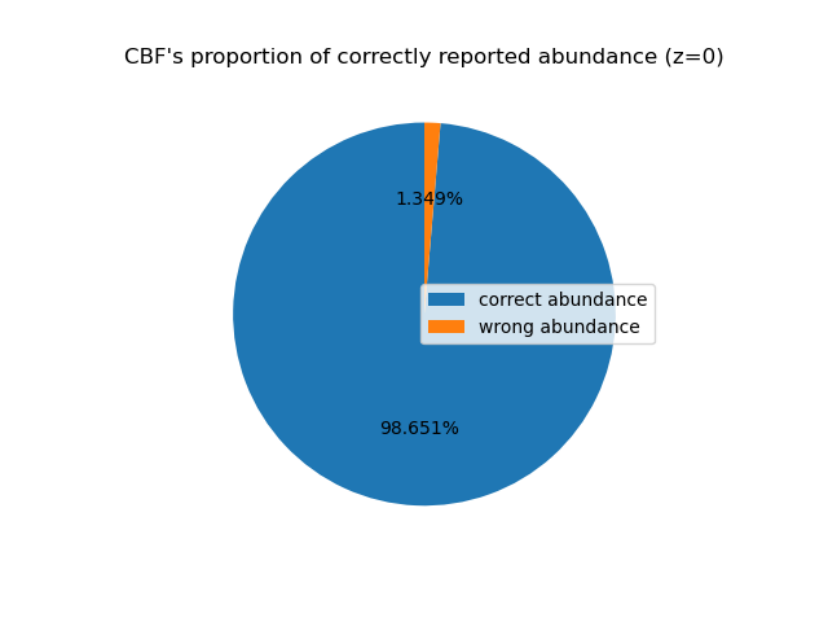


Figure 3: proportion correct abundance calls **without** fimpera

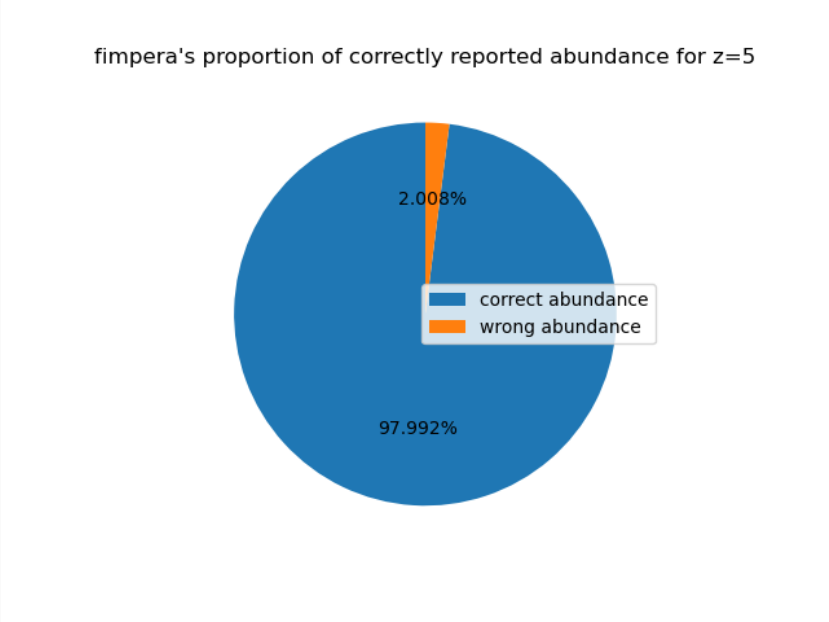


Figure 4: proportion correct abundance calls **with** fimpera

fimpera's effect on abundance error

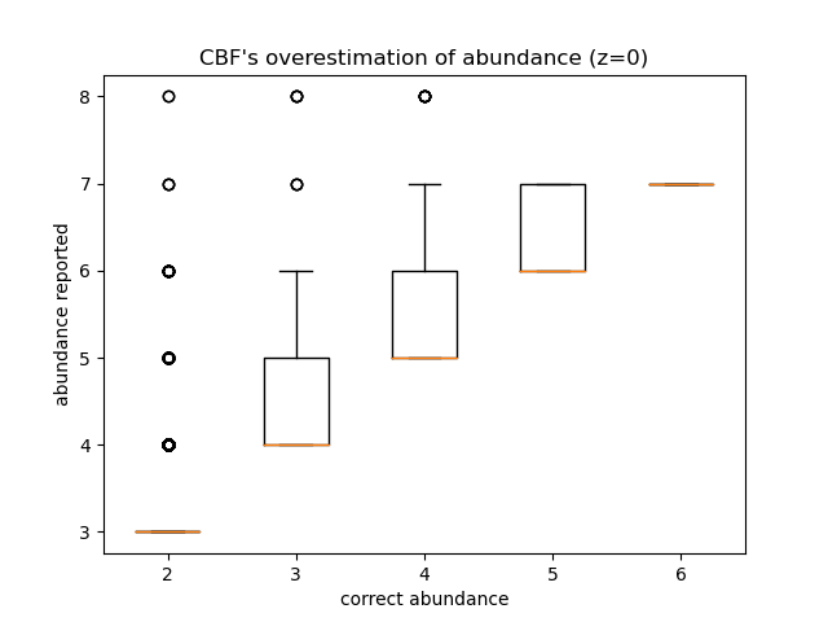


Figure 5: overestimations **without** fimpera

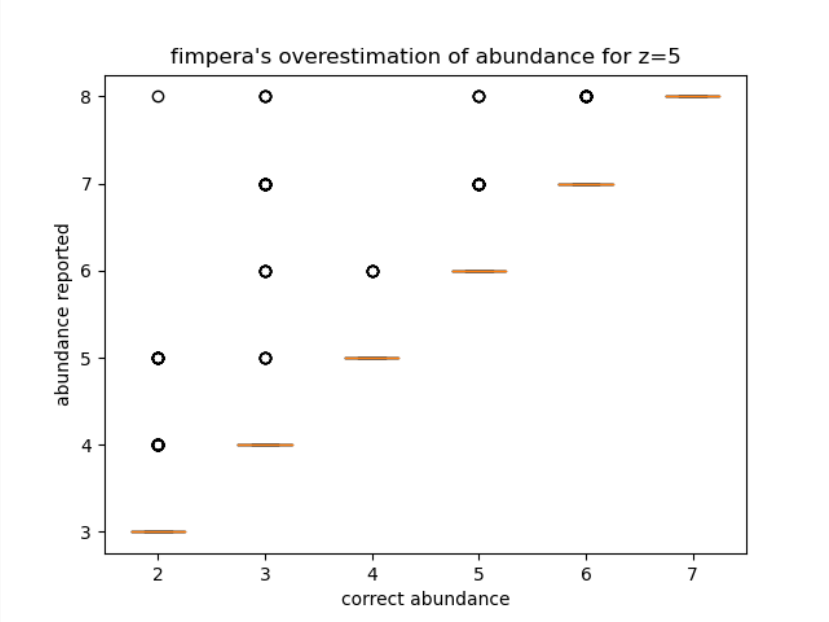


Figure 6: overestimations **with** fimpera

fimpera's effect on presence-absence when used with kmtricks

Last year, we started by implementing the particular case of detecting presence/absence.
The next slides show the effect of using $z = 3$ with a tool made by Téo Lemane, called kmtricks (available at `https://github.com/tlemane/kmtricks`).
`kmtricks` was used to index presence-absence of all TARA datasets (thousands of datasets; 50T of data, 5T after indexation).

fimpera's effect on presence-absence

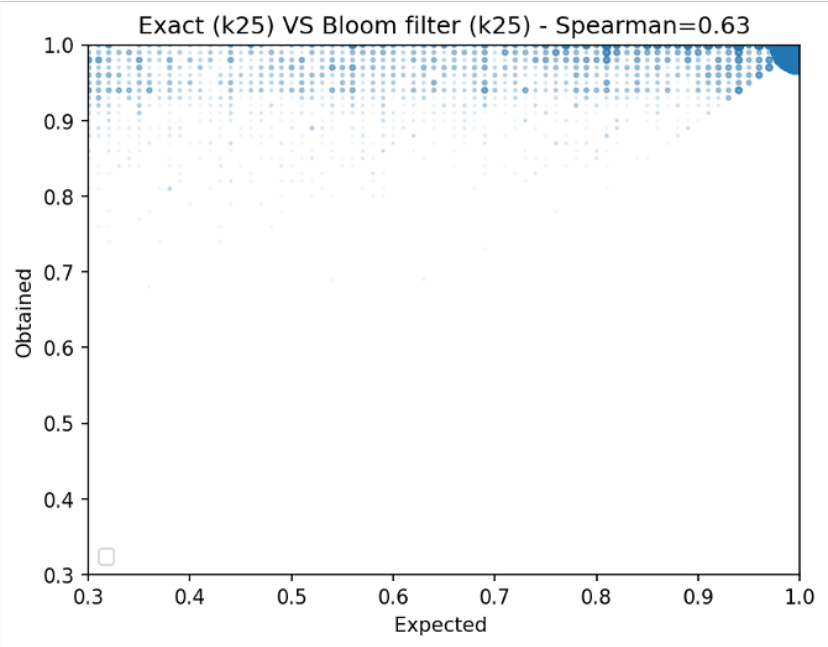


Figure 7: queries made with $z = 0$

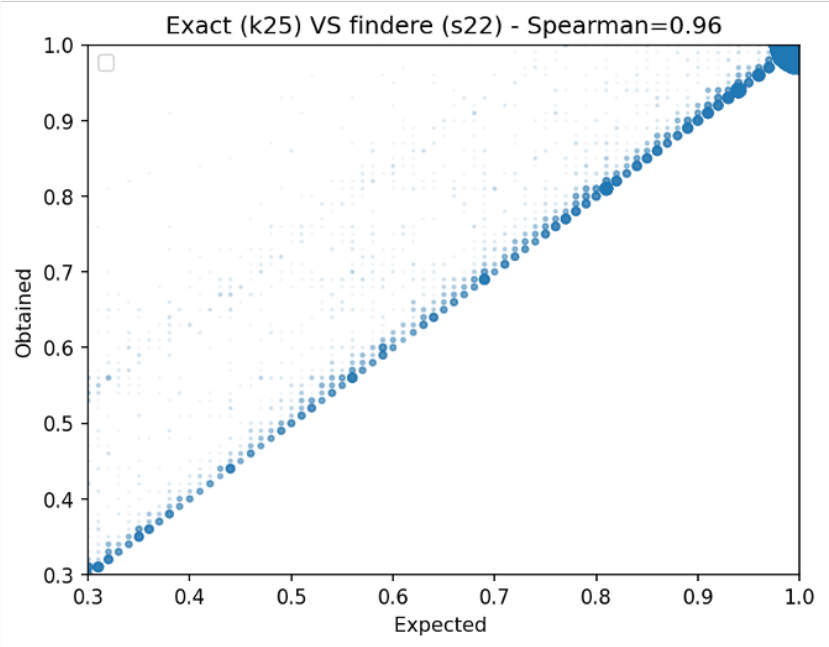


Figure 8: queries made with $z = 3$

conclusion

Conclusion

`fimpera` enables to reduce the false positive rate (or the size) of an *AMQ with abundance*

- `fimpera` is available at `https://github.com/lrobidou/fimpera`
- preprint available at `https://www.biorxiv.org/content/10.1101/2022.06.27.497694v2`