Fast, sustainable and secure bioinformatics
with Rust-Bio and Rust-HTSlib

Johannes Köster
University of Duisburg-Essen
https://koesterlab.github.io

The problem

Bioinformatics software should be:
- efficient
- robust
- maintainable

Do (so far) mainstream languages support this goal?

```cpp
#include <iostream.h>

int main()
{
    cout << "Hello World! I am fast \n    but you might spend the \n    rest of your life with debugging. \n    
    return 0;
}
```

A solution:
the Rust programming language

Ownership concept:
1. Each value is **owned** by one variable.
2. Only **one owner** at a time.
3. When owner goes **out of scope**, the value will be **dropped**.

Compile-time guarantees:
- memory safety: no dangling pointers, no segmentation faults
- thread safety: no race conditions, compiler forces you to explicitly consider synchronization

Rust-Bio and Rust-HTSlib

Algorithms:
- local/semiglobal/global alignment
- pairwise distances
- sparse k-mer based alignment
- Myers bit-pairalignment
- Ukkonen
- Knuth-Morris-Pratt
- Shift-And
- Horspool
- BNDM
- BOM
- PSSM-based motif search

Utilities:
- numerically stable log-probabilities
- PHRED-scale conversion
- cumulative distribution functions
- combinatorics
- read/variant ringbuffers

Data structures:
- suffix array
- BWL
- FM(D) index
- q-gram index
- rank/select
- Fenwick tree
- pair-HMM

File format support:
- SAM/BAM/CRAM
- VCF/BCF/Tabix
- FASTA
- FASTQ
- BED
- GFF
- GTF

https://rust-bio.github.io