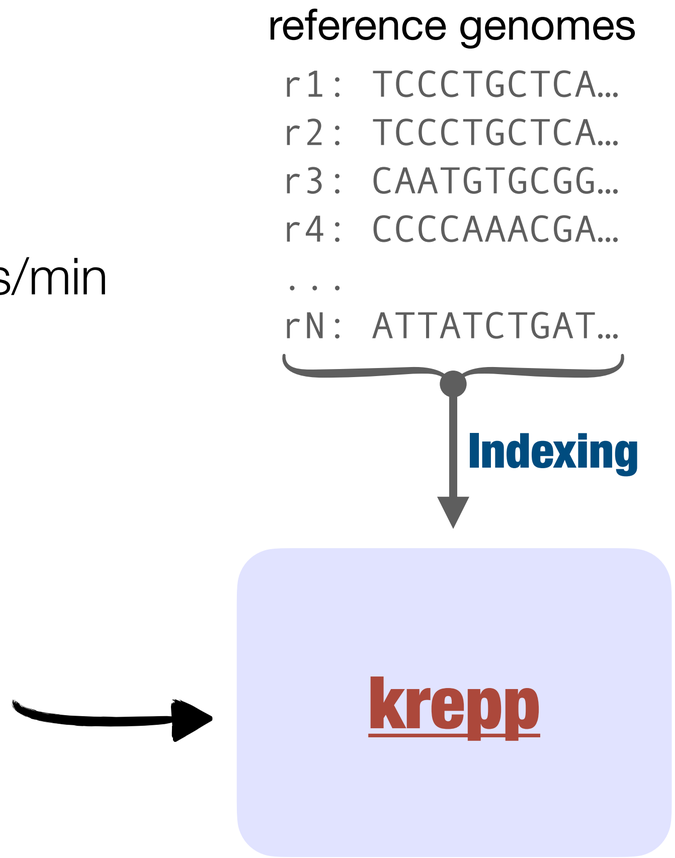
Read-genome distance calculation & phylogenetic placement using krepp

Ali Osman Berk Şapcı
UC San Diego
Aug 14, 2025 — IMSI

Tool overview

- Whole genome sequencing
- N>100K references
- High throughput: >2M short reads/min
- Extensively tested for short reads

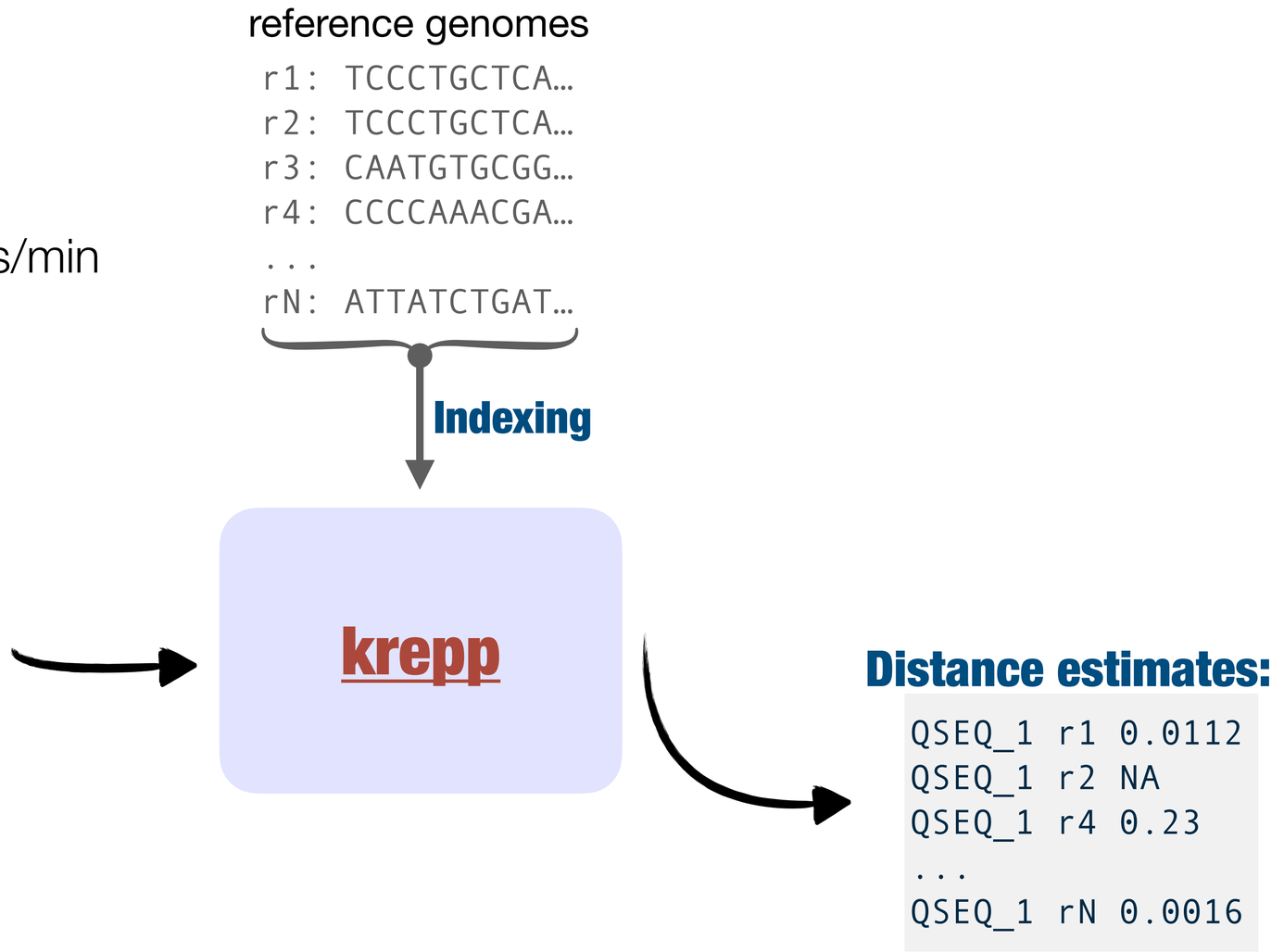


A query of any length from anywhere on the genome:

```
@QSEQ_1
ATACCTAGGAG…
```

Tool overview

- Whole genome sequencing
- N>100K references
- High throughput: >2M short reads/min
- Extensively tested for short reads



Tool overview

- Whole genome sequencing
- N>100K references
- High throughput: >2M short reads/min
- Extensively tested for short reads

reference genomes

```
r1: TCCCTGCTCA…
r2: TCCCTGCTCA…
r3: CAATGTGCGG…
r4: CCCCAAACGA…
...
rN: ATTATCTGAT…
```

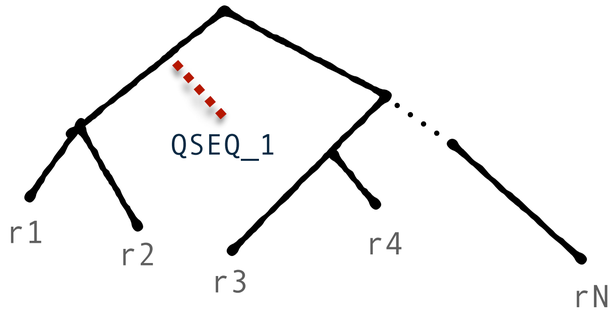
Indexing

A query of any length from anywhere on the genome:

```
@QSEQ_1
ATACCTAGGAG…
```

krepp

Phylogenetic placement:

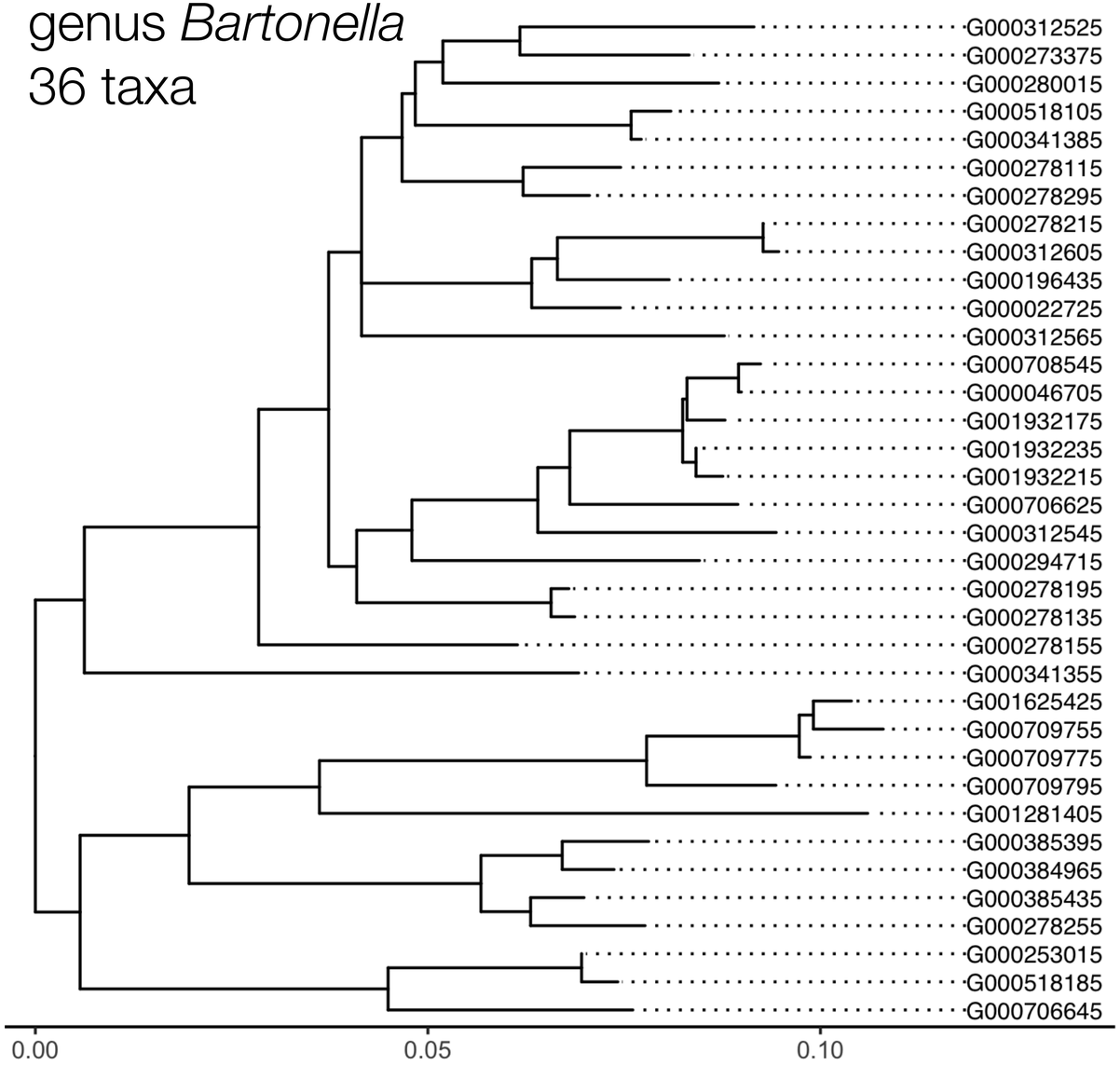


Distance estimates:

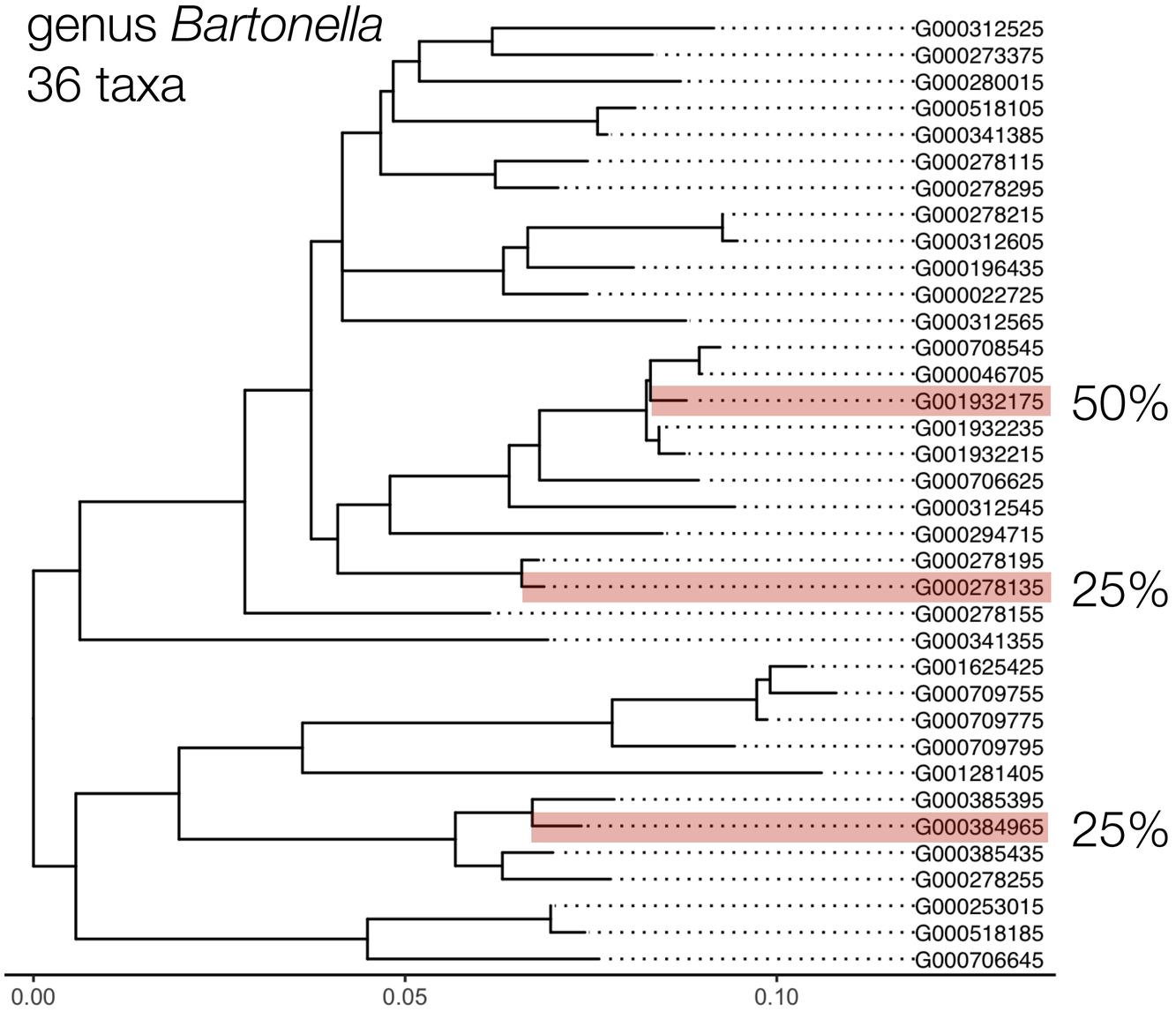
```
QSEQ_1 r1 0.0112
QSEQ_1 r2 NA
QSEQ_1 r4 0.23
...
QSEQ_1 rN 0.0016
```

A toy example

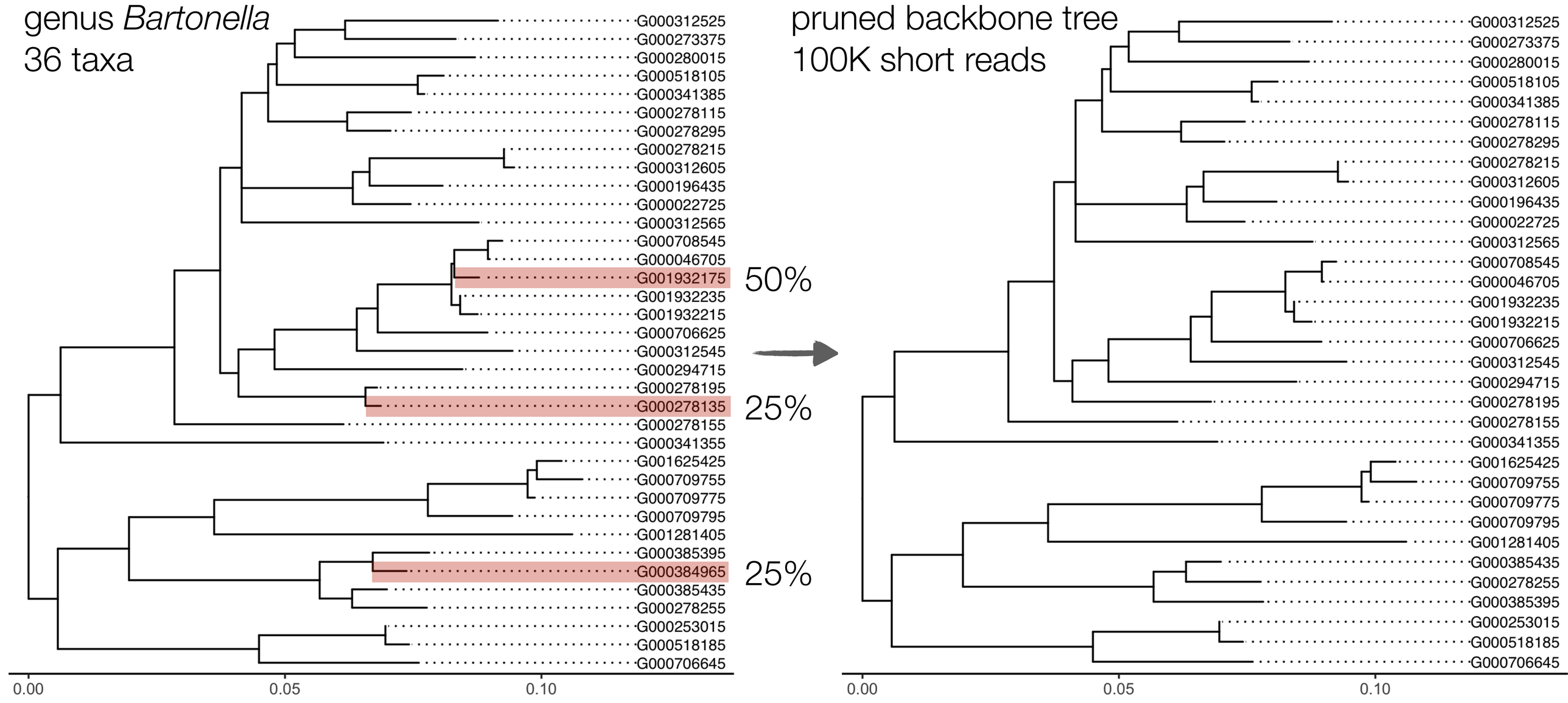
genus *Bartonella*
36 taxa



A toy example



A toy example



Installation and the repository

Pre-compiled binaries are available for the latest release (macOS and Linux)

just download a binary and get started!

▾ **Assets** 4

krepp-v050-linux64.tar.gz	2.27 MB	3 weeks ago
krepp-v050-macOS.tar.gz	816 KB	3 weeks ago
Source code (zip)		3 weeks ago
Source code (tar.gz)		3 weeks ago

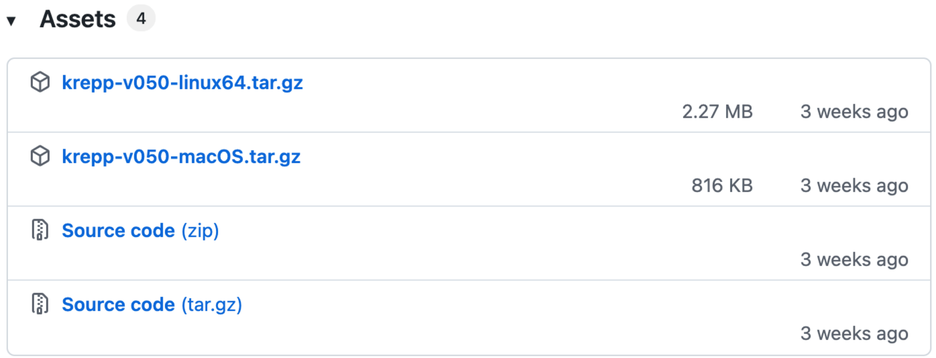
`github.com/bo1929/krepp`

An extensive documentation and a tutorial are available on GitHub

Installation and the repository

Pre-compiled binaries are available for the latest release (macOS and Linux)

just download a binary and get started!



`github.com/bo1929/krepp`

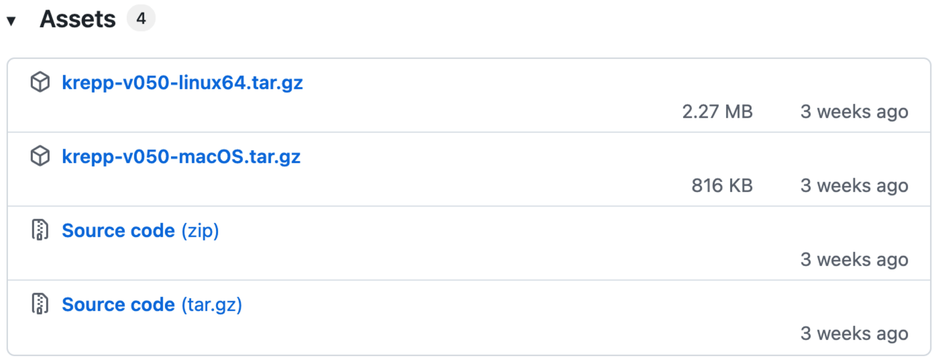
An extensive documentation and a tutorial are available on GitHub

```
curl -L -O 'https://github.com/bo1929/krepp/releases/download/v0.5.0/krepp-v050-macOS.tar.gz'
tar -xzvf krepp-v050-macOS.tar.gz
cp ./krepp-v050-macOS/krepp ~/.local/bin
krepp --help
```

Installation and the repository

Pre-compiled binaries are available for the latest release (macOS and Linux)

just download a binary and get started!



`github.com/bo1929/krepp`

An extensive documentation and a tutorial are available on GitHub

```
curl -L -O 'https://github.com/bo1929/krepp/releases/download/v0.5.0/krepp-v050-macOS.tar.gz'
tar -xzvf krepp-v050-macOS.tar.gz
cp ./krepp-v050-macOS/krepp ~/.local/bin
krepp --help
```

Soon on Bioconda!

Step 1: Installation

Indexing reference genomes

reference genomes

```
r1: TCCCTGCTCA…
r2: TCCCTGCTCA…
r3: CAATGTGCGG…
r4: CCCCAAACGA…
...
rN: ATTATCTGAT…
```

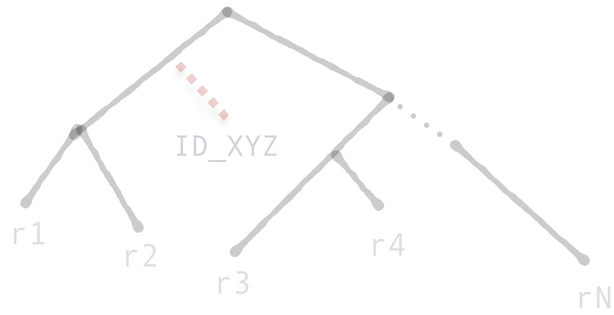
Indexing

A query sequence of any length:
(short reads, long reads, contigs)

```
>ID_XYZ
ATACCTAGGAG…
```

krepp

Phylogenetic placement:



Distance estimates:

```
ID_XYZ r1 0.0112
ID_XYZ r2 0.1445
ID_XYZ r4 0.23
...
ID_XYZ rN 0.0016
```

Indexing reference genomes

```
krepp index -i $INPUT_FILE -t $NWK_FILE -o $INDEX_DIR
```

Indexing reference genomes

```
krepp index -i $INPUT_FILE -t $NWK_FILE -o $INDEX_DIR
```

- `-i $INPUT_FILE:` a TSV file <path> mapping reference IDs to (gzip compatible) paths or URLs

```
UNIQUE_ID <TAB> PATH
REF_XYA    /path/to/XYB.fa.gz
REF_XYB    https://url/XYB.fa.gz
…
REF_XYZ    /path/to/XYZ.fasta
```

PATH:
- ✅ FASTA/FASTQ
- ✅ URL/FILEPATH
- ✅ COMPRESSED

Indexing reference genomes

```
krepp index -i $INPUT_FILE -t $NWK_FILE -o $INDEX_DIR
```

- `-i $INPUT_FILE`: a TSV file <path> mapping reference IDs to (gzip compatible) paths or URLs
- `-t $NWK_FILE`: <path> to the backbone tree (Newick format)—input IDs should match tip labels

```
UNIQUE_ID <TAB> PATH
REF_XYA    /path/to/XYB.fa.gz
REF_XYB    https://url/XYB.fa.gz
…
REF_XYZ    /path/to/XYZ.fasta
```

PATH:
- ✅ FASTA/FASTQ
- ✅ URL/FILEPATH
- ✅ COMPRESSED

optional: indexing wo/ a backbone tree
- can estimate distances
- must specify later for placement

benefit: slightly faster & smaller indices

Indexing reference genomes

```
krepp index -i $INPUT_FILE -t $NWK_FILE -o $INDEX_DIR
```

- `-i $INPUT_FILE`: a TSV file <path> mapping reference IDs to (gzip compatible) paths or URLs
- `-t $NWK_FILE`: <path> to the backbone tree (Newick format)—input IDs should match tip labels
- `-o $INDEX_DIR`: directory <path> to store the index

```
UNIQUE_ID <TAB> PATH
REF_XYA    /path/to/XYB.fa.gz
REF_XYB    https://url/XYB.fa.gz
…
REF_XYZ    /path/to/XYZ.fasta
```

PATH:
- ✅ FASTA/FASTQ
- ✅ URL/FILEPATH
- ✅ COMPRESSED

optional: indexing wo/ a backbone tree
- can estimate distances
- must specify later for placement

benefit: slightly faster & smaller indices

output: a directory with a bunch of binary files required for distance estimation & placement

Configuration of the index

Algorithm parameters:

- `-k,--kmer-len [19 - 31]`: Length of k-mers [29]
- `-w,--win-len`: Length of minimizer window (w>k) [k+6]
- `-h,--num-positions`: Number of positions for the LSH [k-16]

Symmetric DUST to filter low-complexity regions:

- `--sdust-t`: SDUST threshold [20]
- `--sdust-w`: SDUST window [64]

Configuration of the index

Algorithm parameters:

- `-k,--kmer-len [19 - 31]`: Length of k-mers [29]

Defaults just work, don't worry about these!

(unless you are doing something peculiar like aDNA analysis)

- `--sdust-t`: SDUST threshold [20]
- `--sdust-w`: SDUST window [64]

Configuration of the index

Algorithm parameters:

- `-k,--kmer-len [19 - 31]`: Length of k-mers [29]

Defaults just work, don't worry about these!

(unless you are doing something peculiar like aDNA analysis)

- `--sdust-t`: SDUST threshold [20]
- `--sdust-w`: SDUST window [64]

What if we don't have enough memory?

Partial indexing & partial loading existing full indices:

- `-m,—modulo-lsh`: Modulo value to partition LSH space
- `-r,--residue-lsh`: Included a k-mer only if r = LSH(x) mod m

subsampling rate:
$(r+1)/m$

Configuration of the index

Algorithm parameters:

- `-k,--kmer-len [19 - 31]`: Length of k-mers [29]

Defaults just work, don't worry about these!

(unless you are doing something peculiar like aDNA analysis)

- `--sdust-t`: SDUST threshold [20]
- `--sdust-w`: SDUST window [64]

What if we don't have enough memory?

Sacrifice a little accuracy, save a lot of memory!

(more details in the documentation)

`-I, --residue-ish`: Include a k-mer only if $h < 2^{64}/(x)$ mod m

subsampling rate: $(r+1)/m$

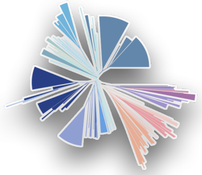
Our index catalogue

- Indexing is relatively cumbersome and expensive
- We provide a catalogue of indices **available to download** from our FTP server!

Our index catalogue

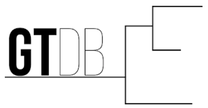
- Indexing is relatively cumbersome and expensive
- We provide a catalogue of indices **available to download** from our FTP server!

Microbial genomes (archaeal and bacterial):



Web of Life (WoL): lightweight and the most reliable phylogeny
65 GB - 16,000 refs. + an accurate phylogeny

RefSeq-microbial: denser sampling and a high quality phylogeny
180 GB - 120,000 refs. + expanded WoL tree using uDance

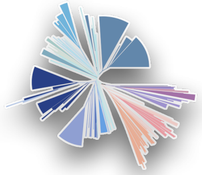


GTDB: the latest release, an up-to-date reference
230 GB - 145,000 refs. + tree w/ taxonomically labeled internal nodes

Our index catalogue

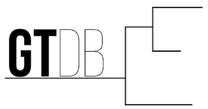
- Indexing is relatively cumbersome and expensive
- We provide a catalogue of indices **available to download** from our FTP server!

Microbial genomes (archaeal and bacterial):



Web of Life (WoL): lightweight and the most reliable phylogeny
65 GB - 16,000 refs. + an accurate phylogeny

RefSeq-microbial: denser sampling and a high quality phylogeny
180 GB - 120,000 refs. + expanded WoL tree using uDance



GTDB: the latest release, an up-to-date reference
230 GB - 145,000 refs. + tree w/ taxonomically labeled internal nodes

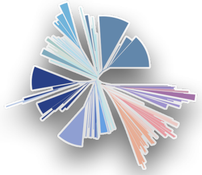
Eukaryotic genomes: all RefSeq eukaryotic references — available soon!

Viral genomes: a representative viral reference — available soon!

Our index catalogue

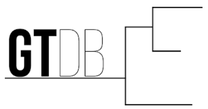
- Indexing is relatively cumbersome and expensive
- We provide a catalogue of indices **available to download** from our FTP server!

Microbial genomes (archaeal and bacterial):



Web of Life (WoL): lightweight and the most reliable phylogeny
65 GB - 16,000 refs. + an accurate phylogeny

RefSeq-microbial: denser sampling and a high quality phylogeny
180 GB - 120,000 refs. + expanded WoL tree using uDance



GTDB: the latest release, an up-to-date reference
230 GB - 145,000 refs. + tree w/ taxonomically labeled internal nodes

Eukaryotic genomes: all RefSeq eukaryotic references — available soon!

Viral genomes: a representative viral reference — available soon!

Links are on GitHub!

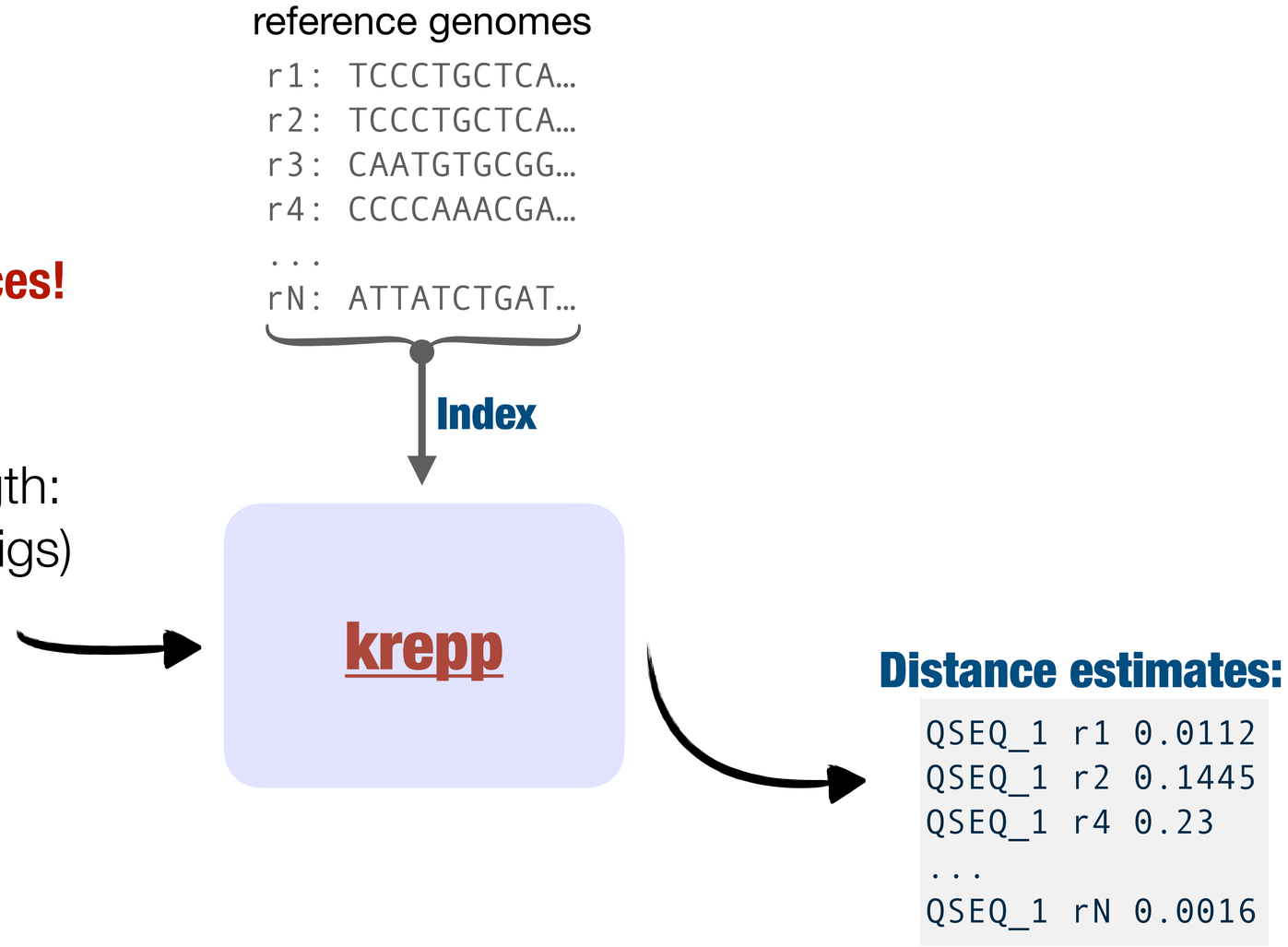
Step 2: Indexing

Estimating read-to-reference distances

Distances of each read to all sufficiently close (~0.25) references!

A query sequence of any length:
(short reads, long reads, contigs)

```
@QSEQ_1
ATACCTAGGAG…
```



Estimating read-to-reference distances

Distances of each read to all sufficiently close (~0.25) references!

A query sequence of any length:
(short reads, long reads, contigs)

```
@QSEQ_1
ATACCTAGGAG…
```

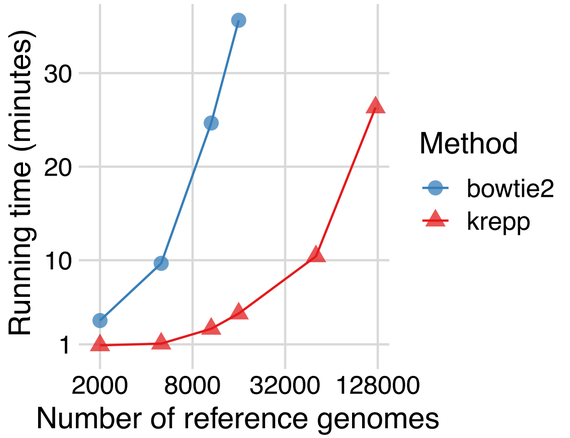
reference genomes

```
r1: TCCCTGCTCA…
r2: TCCCTGCTCA…
r3: CAATGTGCGG…
r4: CCCCAAACGA…
...
rN: ATTATCTGAT…
```

Index

krepp

Estimates are akin to alignment HD, yet more sensitive and faster!



Distance estimates:

```
QSEQ_1 r1 0.0112
QSEQ_1 r2 0.1445
QSEQ_1 r4 0.23
...
QSEQ_1 rN 0.0016
```

Estimating read-to-reference distances

```
krepp dist -i $INDEX_DIR -q $QUERY_FILE -o $OUTPUT_FILE
```

Estimating read-to-reference distances

```
krepp dist -i $INDEX_DIR -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index

Estimating read-to-reference distances

```
krepp dist -i $INDEX_DIR -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index
- `-q $QUERY_PATH`: query FASTA/FASTQ file <path> (or URL) (gzip compatible)

Query FASTQ

```
@QSEQ_1
GCGGATATCTGGGAAACGCCGGTTGCAAGCAGAG
+
=CCGGGGGGGGGGGCJJJJGJJGGGJJJJCC8JJ
@QSEQ_2
TCACCCTCTATTTCCACCACACATACCCGGCAAC
+
=C=GGGCG=CGGGG=JJJJJJ=JJCJJJJCJJJJ
...
```

Estimating read-to-reference distances

```
krepp dist -i $INDEX_DIR -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index
- `-q $QUERY_PATH`: query FASTA/FASTQ file <path> (or URL) (gzip compatible)
- `-o $OUTPUT_FILE`: write output (TSV) to a file at <path> (default: stdout)

Query FASTQ

```
@QSEQ_1
GCGGATATCTGGGAAACGCCGGTTGCAAGCAGAG
+
=CCGGGGGGGGGGGCJJJJGJJGGGJJJJCC8JJ
@QSEQ_2
TCACCCTCTATTTCCACCACACATACCCGGCAAC
+
=C=GGGCG=CGGGG=JJJJJJ=JJCJJJJCJJJJ
…
```

`krepp dist` →

Distance report

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
QSEQ_1	REF_XYD	0.12
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
QSEQ_2	REF_XYC	0.22
QSEQ_3	NA	NA

…

Some relevant options for basic filtering

Some relevant options for basic filtering

Default: `--no-filter --multi`

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
QSEQ_1	REF_XYD	0.12
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
QSEQ_2	REF_XYC	0.22
QSEQ_3	NA	NA
…		

can get quite big…

Some relevant options for basic filtering

Default: `--no-filter --multi`

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
QSEQ_1	REF_XYD	0.12
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
QSEQ_2	REF_XYC	0.22
QSEQ_3	NA	NA

…

can get quite big…

`--filter`: only if indistinguishable from the min

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
~~*QSEQ_1*~~	~~*REF_XYD*~~	~~*0.12*~~
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
~~QSEQ_2~~	~~REF_XYC~~	~~0.22~~
QSEQ_3	NA	NA

…

Some relevant options for basic filtering

Default: `--no-filter --multi`

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
QSEQ_1	REF_XYD	0.12
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
QSEQ_2	REF_XYC	0.22
QSEQ_3	NA	NA
…		

can get quite big…

`--filter`: only if indistinguishable from the min

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
~~*QSEQ_1*~~	~~*REF_XYD*~~	~~*0.12*~~
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
~~QSEQ_2~~	~~REF_XYC~~	~~0.22~~
QSEQ_3	NA	NA
…		

`--no-multi`: only the best matching ref.

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
~~*QSEQ_1*~~	~~*REF_XYD*~~	~~*0.12*~~
~~QSEQ_1~~	~~REF_XYZ~~	~~0.03~~
QSEQ_2	REF_XYA	0.001
~~QSEQ_2~~	~~REF_XYC~~	~~0.22~~
QSEQ_3	NA	NA
…		

Some relevant options for basic filtering

Default: `--no-filter --multi`

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
QSEQ_1	REF_XYD	0.12
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
QSEQ_2	REF_XYC	0.22
QSEQ_3	NA	NA
…		

can get quite big…

Other options are discussed in the docs.

`--filter`: only if indistinguishable from the min

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
~~*QSEQ_1*~~	~~*REF_XYD*~~	~~*0.12*~~
QSEQ_1	REF_XYZ	0.03
QSEQ_2	REF_XYA	0.001
~~QSEQ_2~~	~~REF_XYC~~	~~0.22~~
QSEQ_3	NA	NA
…		

`--no-multi`: only the best matching ref.

SEQ_ID	REFERENCE_NAME	DIST
QSEQ_1	REF_XYA	0.02
~~*QSEQ_1*~~	~~*REF_XYD*~~	~~*0.12*~~
~~QSEQ_1~~	~~REF_XYZ~~	~~0.03~~
QSEQ_2	REF_XYA	0.001
~~QSEQ_2~~	~~REF_XYC~~	~~0.22~~
QSEQ_3	NA	NA
…		

Step 3a: Distance estimation

Placing reads on a backbone tree

Placing each query read on a backbone tree to explain its distance estimates!

reference genomes

```
r1: TCCCTGCTCA…
r2: TCCCTGCTCA…
r3: CAATGTGCGG…
r4: CCCCAAACGA…
...
rN: ATTATCTGAT…
```

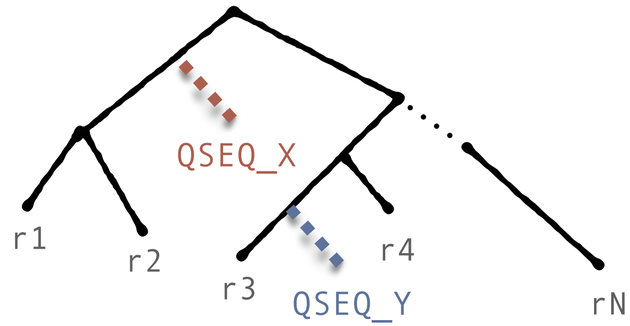
Index

Query sequences of any length:
(short reads, long reads, contigs)

```
>QSEQ_X
ATACCTAGGAG…
>QSEQY
GTTCCTCTGAG…
```

krepp

Phylogenetic placement:



Distance estimates:

```
QSEQ_X r1 0.0112
QSEQ_X r9 0.1445
...
QSEQ_Y r3 0.0016
QSEQ_Y rN 0.1253
```

Placing reads on a backbone tree

```
krepp place -i $INDEX_DIR -t $NWK_FILE -q $QUERY_FILE -o $OUTPUT_FILE
```

Placing reads on a backbone tree

```
krepp place -i $INDEX_DIR -t $NWK_FILE -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index

Placing reads on a backbone tree

```
krepp place -i $INDEX_DIR -t $NWK_FILE -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index
- `-t $NWK_FILE`: backbone tree—tip labels must match ref. IDs (required if the index has no tree)

Placing reads on a backbone tree

```
krepp place -i $INDEX_DIR -t $NWK_FILE -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index
- `-t $NWK_FILE`: backbone tree—tip labels must match ref. IDs (required if the index has no tree)
- `-q $QUERY_PATH`: query FASTA/FASTQ file <path> (or URL) (gzip compatible)

Query FASTQ

```
@QSEQ_1
GCGGATATCTGGGAAACGCCGGTTGCAA
+
=CCGGGGGGGGGGCJJJJGJJGGGJJJ
@QSEQ_2
TCACCCTCTATTTCCACCACACATACCC
+
=C=GGGCG=CGGGG=JJJJJJ=JJCJJJ
...
```

Placing reads on a backbone tree

```
krepp place -i $INDEX_DIR -t $NWK_FILE -q $QUERY_FILE -o $OUTPUT_FILE
```

- `-i $INDEX_DIR`: directory <path> containing the reference index
- `-t $NWK_FILE`: backbone tree—tip labels must match ref. IDs (required if the index has no tree)
- `-q $QUERY_PATH`: query FASTA/FASTQ file <path> (or URL) (gzip compatible)
- `-o $OUTPUT_FILE`: write output (jplace—JSON-based) to a file at <path> (default: stdout)

Query FASTQ

```
@QSEQ_1
GCGGATATCTGGGAAACGCCGGTTGCAA
+
=CCGGGGGGGGGGGCJJJJGJJGGGJJJ
@QSEQ_2
TCACCCTCTATTTCCACCACACATACCC
+
=C=GGGCG=CGGGG=JJJJJJ=JJCJJJ
…
```

krepp place →

jplace report

```
{
"version" : 3,
"fields" : ["edge_num", …, "like_weight_ratio"],
"placements" : [
                {"n" : ["QSEQ_1"], "p" : [
                 [39, 0.0010, …, 1]]},
                {"n" : ["QSEQ_2"], "p" : [
                 [35, 0.005, …, 0.437993]]}
               ]
"metadata" : {<METADATA>},
“tree” : <LABELED_NEWICK_STRING>
}
```

More on jplace format

Just a JSON; not meant to be directly inspected; process using suitable tools: gappa

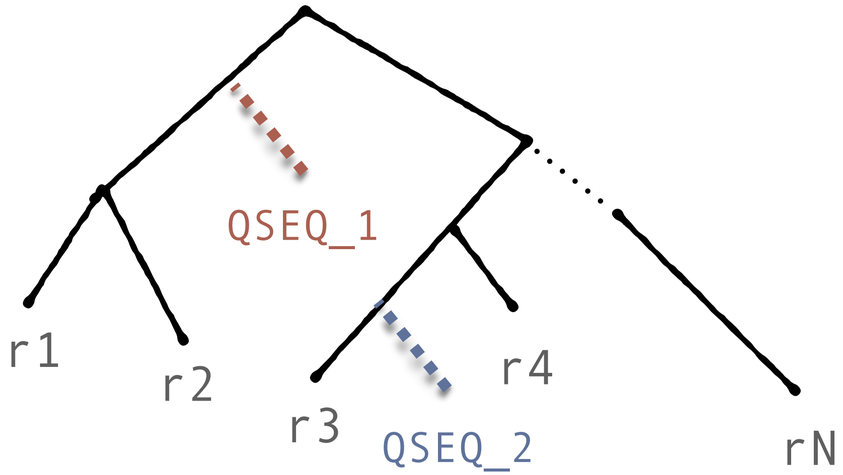
github.com/lczech/gappa

branch numbers from a post-order traversal

```
{
"version" : 3,
"fields" : ["edge_num", "pendant_length", "distal_length", "likelihood", "like_weight_ratio"],
"placements" : [
                {"n" : ["QSEQ_1"], "p" : [
                 [3, 0, 0.00108799, -11.5816, 1]]},
                {"n" : ["QSEQ_2"], "p" : [
                 [18, 0, 0.00585016, -16.6413, 0.437993]]}
               ]
"metadata" :{
      "software" : "krepp",
      "version" : "v0.4.8",
      "repository" : "https://github.com/bo1929/krepp",
      "num_queries" : "2",
      "invocation" : "krepp place -i index -q query.fq"
  },
"tree" : "(REF_XYA:0.42{0},((REF_XYB:0.03{1},REF_XYC:0.12{2})N_XY:0.22{3},…)"
}
```

Multiplacement option

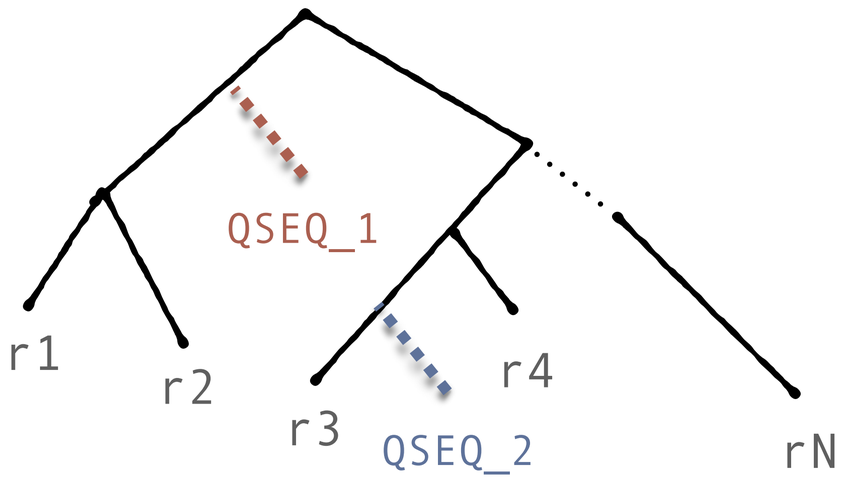
Default: `--no-multi`



Reports only the best placement

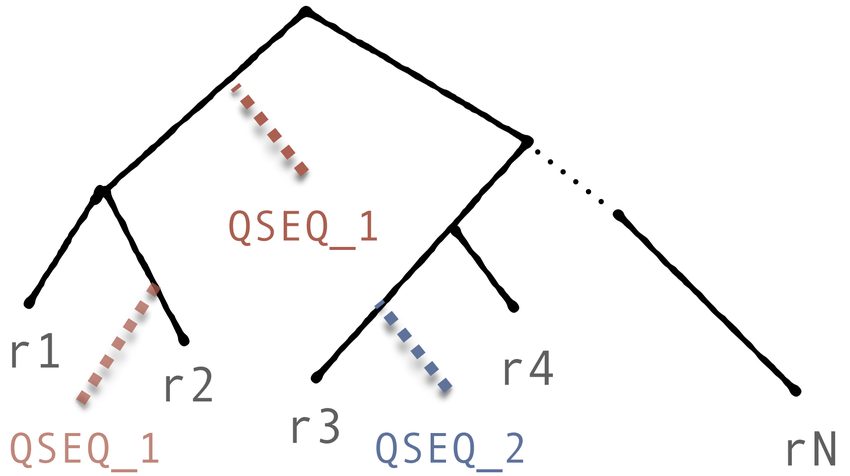
Multiplacement option

Default: `--no-multi`



Reports only the best placement

`--multi`



Multiple likely placements w/ weights

Step 3b: Phylogenetic placement

Visualizing krepp placements with gappa