The Journal of Open Source Software

# CRBHits: From Conditional Reciprocal Best Hits to Codon Alignments and Ka/Ks in R

**Kristian K Ullrich**[1]

**1** Max Planck Institute for Evolutionary Biology, Scientific IT group, August Thienemann Str. 2, 24306 Plön

**DOI:** 10.21105/joss.02424

**Software**

- Review
- Repository
- Archive

**Editor:** Kristina Riemer

**Reviewers:**

- @clauswilke
- @a-r-j

**Submitted:** 26 May 2020
**Published:** 12 November 2020



## Summary

CRBHits is a coding sequence (CDS) analysis pipeline in R (R Core Team, 2019). It reimplements the Conditional Reciprocal Best Hit (CRBH) algorithm crb-blast and covers all necessary steps from sequence similarity searches, codon alignments to Ka/Ks calculations and synteny. The new R package targets ecology, population and evolutionary biologists working in the field of comparative genomics.

The Reciprocal Best Hit (RBH) approach is commonly used in bioinformatics to show that two sequences evolved from a common ancestral gene. In other words, RBH tries to find orthologous protein sequences within and between species. These orthologous sequences can be further analysed to evaluate protein family evolution, infer phylogenetic trees and to annotate protein function (Altenhoff et al., 2019). The initial sequence search step is classically performed with the Basic Local Alignment Search Tool (blast) (Altschul et al., 1990) and due to evolutionary constraints, in most cases protein coding sequences are compared between two species. Downstream analysis use the resulting RBH to cluster sequence pairs and build so-called orthologous groups like e.g. OrthoFinder (Emms & Kelly, 2015) and other tools.

The CRBH algorithm was introduced by Aubry et al. (2014) and builds upon the traditional RBH approach to find additional orthologous sequences between two sets of sequences. As described earlier (Aubry et al., 2014; Scott, 2017), CRBH uses the sequence search results to fit an expect value (E-value) cutoff given each RBH to subsequently add sequence pairs to the list of bona-fide orthologs given their alignment length.

Unfortunately, as mentioned by Scott (2017), the original implementation of CRBH (crb-blast) lag improved blast-like search algorithm to speed up the analysis. As a consequence, Scott (2017) ported CRBH to python shmlast, while shmlast cannot deal with IUPAC nucleotide code so far.

CRBHits constitutes a new R package, which build upon previous implementations and ports CRBH into the R environment, which is popular among biologists. CRBHits improve CRBH by additional implemented filter steps (Rost, 1999) and the possibility to apply custom filters prior E-value fitting. Further, the resulting CRBH pairs can be evaluated for the presence of tandem duplicated genes, gene order based syntenic groups and evolutionary rates.

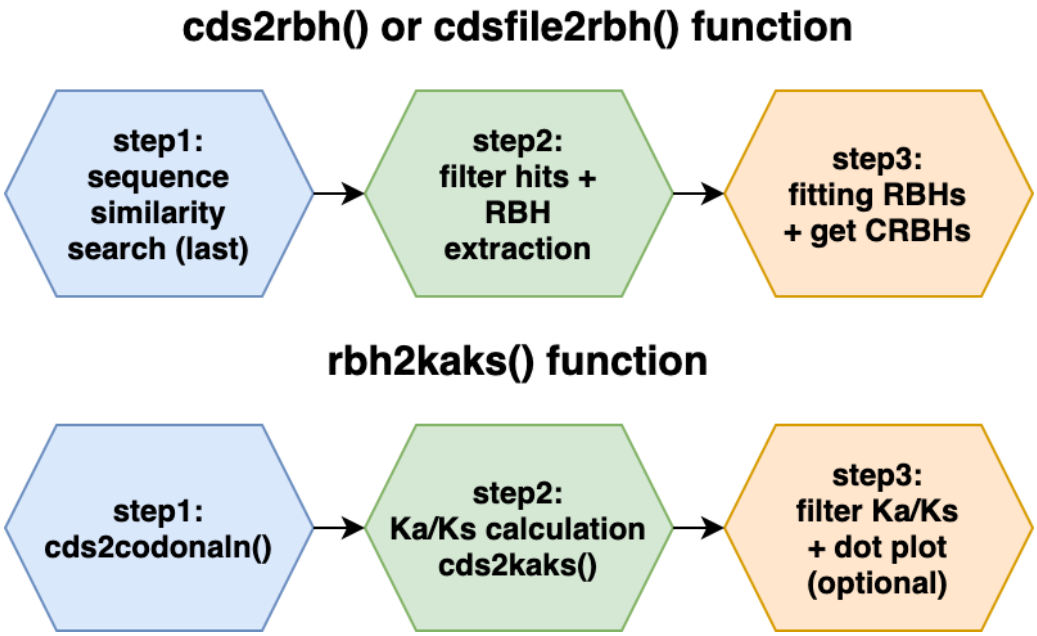


**Figure 1:** Overview of the two main pipeline function and its subtasks. cds2rbh(): from CDS to CRBHit pairs; rbh2kaks(): from CRBHit pairs to Ka/Ks values.

## Coding sequence analysis and synteny

Calculating synonymous (Ks) and nonsynonymous substitutions (Ka) per orthologous sequence pair is a common task for evolutionary biologists, since its ratio Ka/Ks can be used as an indicator of selective pressure acting on a protein (Kryazhimskiy & Plotkin, 2008). However, this task is computational more demanding and consist of at least two steps, namely codon sequence alignment creation and Ka/Ks calculation. Further, the codon sequence alignment step consist of three subtasks, namely coding nucleotide to protein sequence translation, pairwise protein sequence alignment calculation and converting the protein sequence alignment back into a codon based alignment.

Downstream of CRBH creation, CRBHits features all above mentioned steps and subtasks. CRBHits has the ability to directly create codon alignments within R with the help of the widely used R package Biostrings (Pagès et al., 2017) (more than 200k downloads per year since 2014). These codon alignments can be subsequently used to calculate synonymous and nonsynonymous substitutions per sequence pair and is implemented in a multithreaded fashion either via the R package seqinr (Charif & Lobry, 2007) or the use of an R external tool KaKs_Calculator2.0 (Wang et al., 2010).

As gene duplication is one driving force in evolution (Ohno, 1970), the classification of genes as duplicates is one important step to provide us with insights into the molecular events responsible for the current genome architecture of species (Haas et al., 2004). New long-read sequencing technology make more and more chromosome scale assemblies for model and non-model species available. The resulting chromosomal gene order information can be used with sequence similarity scores to classify genes into different types of duplication events, like tandem duplicates or chromosomal segments (syntenic regions) derived from e.g. whole-genome duplication. CRBHits features this classification step via the integration of the R external tool DAGchainer (Haas et al., 2004) and offers the possibility to directly link it with evolutionary rate estimations (see Figure 3).

## Implementation

Like shmlast, CRBHits benefits from the blast-like sequence search software LAST(Kiełbasa et al., 2011) and plots the fitted model of the CRBH E-value based algorithm. In addition, users can filter the hit pairs prior to CRBH fitting for other criteria like query coverage, protein identity and/or the twilight zone of protein sequence alignments according to Rost (1999). The implemented filter uses equation 2 (see Rost, 1999):

$$f(x_{\text{hit pair}}) = \begin{cases} 100 \text{ , for } L_{\text{hit pair}} < 11 \\ 480 * L^{-0.32*(1+e^{\frac{-L}{1000}})} \text{ , for } L_{\text{hit pair}} \leq 450 \\ 19.5 \text{ , for } L_{\text{hit pair}} > 450 \end{cases}$$

where $x_{\text{hit pair}}$ is the expected protein identity given the alignment length $L_{\text{hit pair}}$. If the actual protein identity of a hit pair exceeds the expected protein identity ($pident_{\text{hit pair}} \geq f(x_{\text{hit pair}})$), it is retained for subsequent CRBH calculation.

In contrast to previous implementations, CRBHits only take coding nucleotide sequences (CDS) as the query and target inputs. This is due to the downstream functionality of CRBHits to directly calculate codon alignments within R, which rely on CDS. The inputs are translated into protein sequences, aligned globally (Smith & Waterman, 1981) and converted into codon alignments.

Functions are completely coded in R and only the external prerequisites (LAST, KaKs_Calculator2.0 and DAGchainer) need to be compiled. However, all of them are forked within CRBHits and can be easily build with the dedicated R functions `make.last()`, `make.KaKs_Calculator2()` and `make.dagchainer()`. Further, users can create their own RBH filters before CRBH calculation.

## Functions and Examples

The following example shows how to obtain CRBHit pairs between the coding sequences of *Schizosaccharomyces pombe* (fission yeast) (Wood et al., 2012) and *Nematostella vectensis* (starlet sea anemone) (Apweiler et al., 2004) by using two URLs as input strings and multiple threads for calculation.

```
library(CRBHits)
#set URLs for Schizosaccharomyces pombe (fission yeast)
#and Nematostella vectensis (starlet sea anemone) from NCBI Genomes
cds1.url <- paste0("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/945/",
                   "GCF_000002945.1_ASM294v2/",
                   "GCF_000002945.1_ASM294v2_cds_from_genomic.fna.gz")
cds2.url <- paste0("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/209/225/",
                   "GCF_000209225.1_ASM20922v1/",
                   "GCF_000209225.1_ASM20922v1_cds_from_genomic.fna.gz")
#calculate CBRBhit pairs
cds1.cds2.crbh <- cdsfile2rbh(cds1.url, cds2.url, longest.isoform = TRUE,
                              isoform.source = "NCBI", plotCurve = TRUE,
                              threads = 8)
#get help ?cdsfile2rbh
```

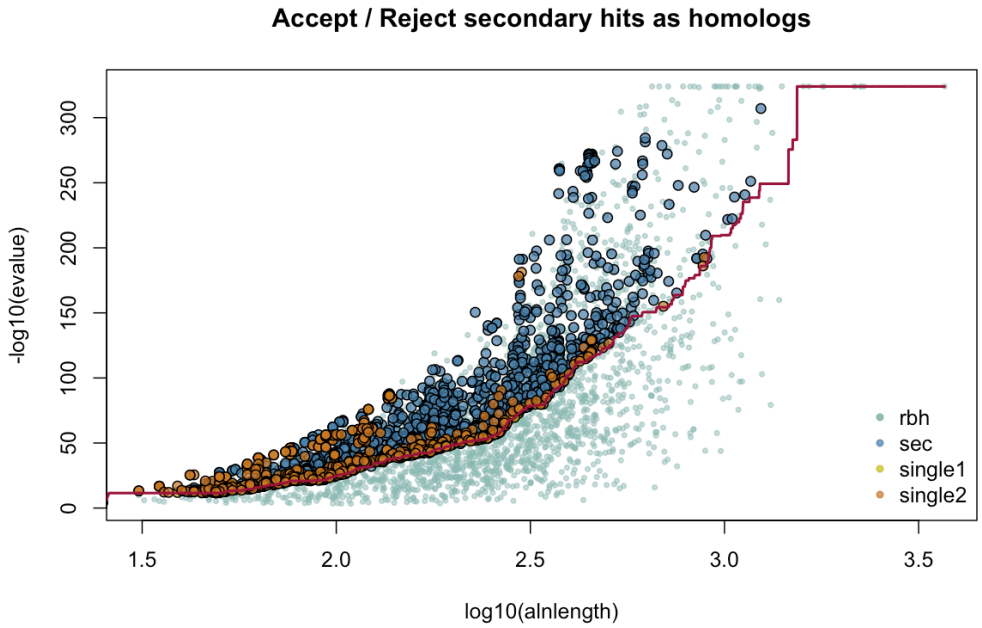


**Figure 2:** Accepted condition reciprocal best hits based on RBH fitting.

The obtained CRBHit pairs can also be used to calculate synonymous (Ks) and nonsynonymous (Ka) substitutions per hit pair using either the model from Li (1993) or from Yang & Nielsen (2000).

```
#download and simultaneously get longest isoform for
#Schizosaccharomyces pombe (fission yeast) and
#Nematostella vectensis (starlet sea anemone)
cds1 <- isoform2longest(Biostrings::readDNAStringSet(cds1.url))
cds2 <- isoform2longest(Biostrings::readDNAStringSet(cds2.url))
#calculate Ka/Ks values for each CRBHit pair
cds1.cds2.kaks.Li <- rbh2kaks(cds1.cds2.crbh, cds1, cds2,
                              model = "Li", threads = 8)
cds1.cds2.kaks.YN <- rbh2kaks(cds1.cds2.crbh, cds1, cds2,
                              model = "YN", threads = 8)

#get help ?rbh2kaks
```

Given the annotated chromosomal gene positions it is also possible to assign tandem duplicated genes per chromosome and directly compute chains of syntenic genes via the use of the R external tool DAGchainer(Haas et al., 2004). Here, *Arabidopsis thaliana* is compared to itself (so called selfblast) and syntenic groups vsiualized by their Ks values.

```
#download and simultaneously get longest isoform for
#Arabidopsis thaliana
cds3.url <- paste0("ftp://ftp.ensemblgenomes.org/pub/plants/release-48/fasta/",
                   "arabidopsis_thaliana/cds/",
                   "Arabidopsis_thaliana.TAIR10.cds.all.fa.gz")
cds3 <- isoform2longest(Biostrings::readDNAStringSet(cds3.url), "ENSEMBL")
#extract gene position and chromosomal gene order
```

```
cds3.genepos <- cds2genepos(cds3, source = "ENSEMBL")
#calculate CBRBhit pairs
cds3.selfblast.crbh <- cds2rbh(cds3, cds3, longest.isoform = TRUE,
                               qcov = 0.5, rost1999 = TRUE,
                               isoform.source = "ENSEMBL", plotCurve = TRUE,
                               threads = 8)
#compute chains of syntenic genes and plor chr1, chr2, chr3, chr4, chr5
cds3.selfblast.synteny <- rbh2dagchainer(cds3.selfblast.crbh,
                                         cds3.genepos, cds3.genepos,
                                         plotDotPlot = TRUE,
                                         select.chr = c("1", "2", "3", "4", "5"))
#calculate Ka/Ks values for each CRBHit pair
cds3.selfblast.kaks.Li <- rbh2kaks(cds3.selfblast.crbh, cds3, cds3,
                                   model = "Li", threads = 8)

#get help ?rbh2dagchainer
#get help ?plot.dagchainer
#get help ?plot.kaks
```

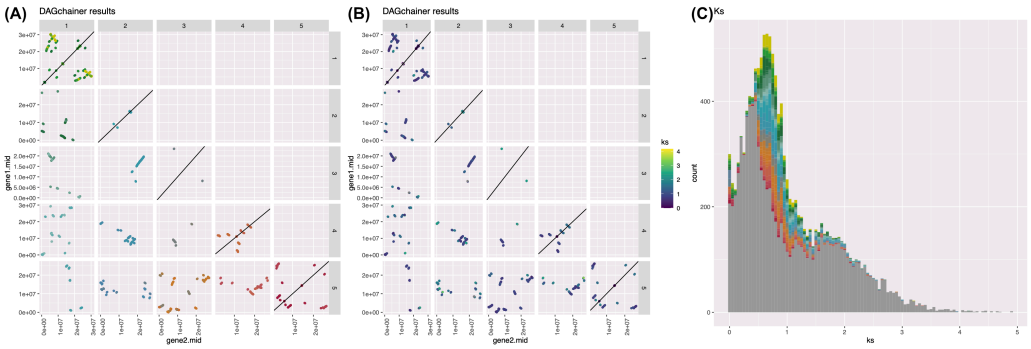


**Figure 3:** Selfblast CRBHit pair results for *Arabidopsis thaliana*. (A) DAGchainer dotplot per chromosome colored by syntenic group and (B) colored by Ks. (C) Histogram of Ks values colored by syntenic group.

**Table 1:** Performance comparison for CRBHit pair (*Schizosaccharomyces pombe* vs. *Nematostella vectensis*) and Ka/Ks calculations (Intel Xeon CPU E5-2620 v3 @ 2.40GHz; 3575 hit pairs).

| Number of Threads | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| Runtime of CRBH(shmlast v1.6) in sec | 38 (s) | 25 (s) | 20 (s) | 16 (s) |
| Runtime of CRBH(CRBHits) in sec | 18 (s) | 10 (s) | 7 (s) | 6 (s) |
| Runtime of kaks.Li in sec | 357 (s) | 167 (s) | 87 (s) | 49 (s) |
| Runtime of kaks.YN in sec | 474 (s) | 230 (s) | 121 (s) | 63 (s) |

## Conclusions

CRBHits implements CRBH in R (see Figure 2), can be used to calculate codon alignment based nucleotide diversities (Ka/Ks) and synteny, in a multithreaded fashion (see Table 1).

## Availability

CRBHits is an open source software made available under the MIT license. It can be installed from its gitlab repository using the devtools package.

```
devtools::install_gitlab("mpievolbio-it/crbhits",
 host = "https://gitlab.gwdg.de")", build_vignettes = TRUE)
```

The R package website, which contain a detailed HOWTO to install the prerequisites (mentioned above) and package vignettes are availbale at https://mpievolbio-it.pages.gwdg.de/crbhits.

## References


Altenhoff, A. M., Glover, N. M., & Dessimoz, C. (2019). Inferring orthology and paralogy. In *Evolutionary genomics* (pp. 149–175). Springer. https://doi.org/10.1007/978-1-4939-9074-0_5

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*(3), 403–410. https://doi.org/10.1016/S0022-2836(05)80360-2

Apweiler, R., Bairoch, A., & Wu, C. H. (2004). Protein sequence databases. *Current Opinion in Chemical Biology*, *8*(1), 76–80. https://doi.org/10.1016/j.cbpa.2003.12.004

Aubry, S., Kelly, S., Kümpers, B. M., Smith-Unna, R. D., & Hibberd, J. M. (2014). Deep evolutionary comparison of gene expression identifies parallel recruitment of trans-factors in two independent origins of C4 photosynthesis. *PLoS Genetics*, *10*(6). https://doi.org/10.1371/journal.pgen.1004365

Charif, D., & Lobry, J. R. (2007). SeqinR 1.0-2: A contributed package to the r project for statistical computing devoted to biological sequences retrieval and analysis. In *Structural approaches to sequence evolution* (pp. 207–232). Springer. https://doi.org/10.1007/978-3-540-35306-5_10

Emms, D. M., & Kelly, S. (2015). OrthoFinder: Solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. *Genome Biology*, *16*(1), 157. https://doi.org/10.1186/s13059-015-0721-2

Haas, B. J., Delcher, A. L., Wortman, J. R., & Salzberg, S. L. (2004). DAGchainer: a tool for mining segmental genome duplications and synteny. *Bioinformatics*, *20*(18), 3643–3646. https://doi.org/10.1093/bioinformatics/bth397

Kiełbasa, S. M., Wan, R., Sato, K., Horton, P., & Frith, M. C. (2011). Adaptive seeds tame genomic sequence comparison. *Genome Research*, *21*(3), 487–493. https://doi.org/10.1101/gr.113985.110

Kryazhimskiy, S., & Plotkin, J. B. (2008). The population genetics of dN/dS. *PLoS Genetics*, *4*(12). https://doi.org/10.1371/journal.pgen.1000304

Li, W.-H. (1993). Unbiased estimation of the rates of synonymous and nonsynonymous substitution. *Journal of Molecular Evolution*, *36*(1), 96–99. https://doi.org/10.1007/BF02407308

Ohno, S. (1970). *Evolution by gene duplication*. Springer Science & Business Media. https://doi.org/10.1007/978-3-642-86659-3

Pagès, H., Aboyoun, P., Gentleman, R., & DebRoy, S. (2017). Biostrings: Efficient manipulation of biological strings. *R Package Version*, *2*(0). https://doi.org/10.18129/B9.bioc.Biostrings

R Core Team. (2019). *R: A language and environment for statistical computing*. https://www.r-project.org/

Rost, B. (1999). Twilight zone of protein sequence alignments. *Protein Engineering*, *12*(2), 85–94. https://doi.org/10.1093/protein/12.2.85

Scott, C. (2017). Shmlast: An improved implementation of conditional reciprocal best hits with LAST and python. *Journal of Open Source Software*, *2*(9), 142. https://doi.org/doi:10.21105/joss.00142

Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, *147*(1), 195–197. https://doi.org/10.1016/0022-2836(81)90087-5

Wang, D., Zhang, Y., Zhang, Z., Zhu, J., & Yu, J. (2010). KaKs_calculator 2.0: A toolkit incorporating gamma-series methods and sliding window strategies. *Genomics, Proteomics & Bioinformatics*, *8*(1), 77–80. https://doi.org/10.1016/S1672-0229(10)60008-3

Wood, V., Harris, M. A., McDowall, M. D., Rutherford, K., Vaughan, B. W., Staines, D. M., Aslett, M., Lock, A., Bähler, J., Kersey, P. J., & others. (2012). PomBase: A comprehensive online resource for fission yeast. *Nucleic Acids Research*, *40*(D1), D695–D699. https://doi.org/10.1093/nar/gkr853

Yang, Z., & Nielsen, R. (2000). Estimating synonymous and nonsynonymous substitution rates under realistic evolutionary models. *Molecular Biology and Evolution*, *17*(1), 32–43. https://doi.org/10.1093/oxfordjournals.molbev.a026236