




SOFTWARE TOOL ARTICLE

Expanding the Orthologous Matrix (OMA) programmatic interfaces: REST API and the *OmaDB* packages for R and Python [version 1; referees: awaiting peer review]

Klara Kaleb¹, Alex Warwick Vesztröcy^{1,2}, Adrian Altenhoff^{2,3},
Christophe Dessimoz  ^{1,2,4-6}

¹Centre for Life's Origins and Evolution, Department of Genetics, Evolution and Environment, University College London, London, WC1E 6BT, UK

²Swiss Institute of Bioinformatics, Lausanne, Switzerland

³Department of Computer Science, ETH Zurich, Zurich, Switzerland

⁴Department of Computer Science, University College London, London, WC1E 6BT, Switzerland

⁵Department of Computational Biology, University of Lausanne, Lausanne, 1015, Switzerland

⁶Center for Integrative Genomics, University of Lausanne, Lausanne, 1015, Switzerland

V1 First published: 10 Jan 2019, 8:42 (
<https://doi.org/10.12688/f1000research.17548.1>)

Latest published: 10 Jan 2019, 8:42 (
<https://doi.org/10.12688/f1000research.17548.1>)

Abstract

The Orthologous Matrix (OMA) is a well-established resource to identify orthologs among many genomes. Here, we present two recent additions to its programmatic interface, namely a REST API, and user-friendly R and Python packages called *OmaDB*. These should further facilitate the incorporation of OMA data into computational scripts and pipelines. The REST API can be freely accessed at <https://omabrowser.org/api>. The R *OmaDB* package is available as part of Bioconductor at <http://bioconductor.org/packages/OmaDB/>, and the *omadb* Python package is available from the Python Package Index (PyPI) at <https://pypi.org/project/omadb/>.

Keywords

orthologs, paralogs, hierarchical orthologous groups, comparative genomics, orthologous matrix, oma, API, R, python, REST, bioconductor

Open Peer Review

Referee Status: *AWAITING PEER*

REVIEW

Discuss this article

Comments (0)



This article is included in the **Bioconductor** gateway.



This article is included in the **RPackage** gateway.



This article is included in the **Python Collection** collection.

Corresponding author: Christophe Dessimoz (Christophe.Dessimoz@unil.ch)

Author roles: **Kaleb K:** Conceptualization, Formal Analysis, Investigation, Methodology, Software, Writing – Original Draft Preparation; **Warwick Vesztrocy A:** Methodology, Software; **Altenhoff A:** Conceptualization, Investigation, Resources, Software, Supervision, Validation; **Dessimoz C:** Conceptualization, Funding Acquisition, Methodology, Project Administration, Supervision, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: We acknowledge support by Swiss National Science Foundation grant 150654, UK BBSRC grant BB/M015009/1, the Swiss State Secretariat for Education, Research and Innovation (SERI), as well as a UCL Genetics, Evolution and Environment Departmental Summer Bursary (to KK).

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2019 Kaleb K *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution Licence](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Kaleb K, Warwick Vesztrocy A, Altenhoff A and Dessimoz C. **Expanding the Orthologous Matrix (OMA) programmatic interfaces: REST API and the *OmaDB* packages for R and Python [version 1; referees: awaiting peer review]** *F1000Research* 2019, **8**:42 (<https://doi.org/10.12688/f1000research.17548.1>)

First published: 10 Jan 2019, **8**:42 (<https://doi.org/10.12688/f1000research.17548.1>)

Introduction

Orthologs are pairs of protein coding genes that have common ancestry and have diverged due to speciation events¹. The detection of orthologs is of fundamental importance in many fields in biology, such as comparative genomics, as it allows us to propagate existing biological knowledge to ever growing newly sequenced data^{2,3}.

The Orthologous Matrix (OMA) is a method and resource for the inference of orthologs among complete genomes⁴. The OMA database (<https://omabrowser.org>) features broad scope and size with currently over 2,100 species from all three domains of life.

The OMA browser has supported multiple ways of exporting the underlying data from its beginning. Users can download data either via bulk archives or interactively through the browser—using where possible standard file formats, such as FASTA, OrthoXML⁵, or PhyloXML⁶. For programmatic access, early OMA database releases offered an Application Programming Interface (API) in the form of the Simple Object Access Protocol (SOAP). However, the complexity and limited adoption of SOAP has prompted us to recently switch to the simpler, faster, and more widely used Representational State Transfer (REST) protocol for the OMA API⁴. Here, we provide a description of this new OMA REST API.

Furthermore, the R environment is widely used in bioinformatics due to its flexibility as a high-level scripting language, statistical capabilities, and numerous bioinformatics libraries. In particular, the Bioconductor open source framework contains over 2,000 packages to facilitate either access to or manipulation of biological data⁷. This motivated us to develop the OmaDB Bioconductor package which provides a more idiomatic and user-friendly access to OMA data in R implemented on top of the REST API.

Finally, to also enable Python users to easily interact with the database, we have developed a similar package in that language, compliant with the conventions and with support of typical complementary Python packages as outlined below.

Methods

We start by describing the OMA REST API, before moving on to detail the OmaDB Bioconductor package, and finally outline the omadb Python package.

OMA REST API

The REST framework is an API architectural style that is based on URLs and HTTP protocol methods. It was designed to be stateless and thus is context independent. That is, it does not save data internally between the HTTP requests which minimises server-side application state, thus easing parallelism. The combination of the HTTP and JSON data formats makes it particularly suitable for web applications and easily supported by most programming languages.

Since the backend of the OMA browser is almost fully based on Python and its frontend is supported by the Django web framework⁸, we have opted to use the Django Rest Framework (DRF) to implement a REST API in our latest release⁴. Most API calls require querying the OMA database, stored in HDF5⁹, using a custom Python library (“pyoma”). The query results are serialised in the format requested by the user — typically JSON.

Most data available through the OMA browser is now also accessible via the API. This includes individual genes and their attributes such as protein or cDNA sequences, cross-references, pairwise orthologs, hierarchical orthologous groups¹⁰, as well as species trees and the corresponding taxonomy. The API documentation as well as the interactive interface can be found at <https://omabrowser.org/api/docs> (Figure 1).

OmaDB Bioconductor package

To facilitate simplified access to the API and downstream analyses in the R environment, we have also developed an API wrapper package in R, now available in Bioconductor⁷ (<http://bioconductor.org/packages/OmaDB/>). This allowed for abstraction of the server interface, eliminating the need to know structure of the database or the URL endpoints to access the required data.

The package consists of a collection of functions that import OMA data into R friendly objects, namely S3 objects and data frames—depending on the query supplied. Due to the volume of the data available, some selected object attributes are at first given as URL endpoints. However, these are automatically loaded upon accession. OmaDB also facilitates further downstream analyses with other Bioconductor packages, such as GO enrichment analysis with topGO¹¹, sequence analysis with BioStrings¹², phylogenetic analyses using ggtree¹³ or gene locus analyses with the help of GenomicRanges¹⁴.

close_groups ⇌ INTERACT

GET `/api/group/{group_id}/close_groups/`

Retrieve the sorted list of closely related groups for a given OMA group.

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
group_id <small>required</small>	an unique identifier for an OMA group - either its group number, its fingerprint or an entry id of one of its members

```
# Load the schema document
$ coreapi get https://omabrowser.org/api/docs

# Interact with the API endpoint
$ coreapi action group close_groups -p group_id=...
```

close_groups DATA RAW

group_id *

68

an unique identifier for an OMA group - either its group number, its fingerprint or an entry id of one of its members

GET `/api/group/68/close_groups/` 200

```
[
  {
    "oma_group": 764412,
    "group_url": "https://omabrowser.org/api/group/764412",
    "hits": 6
  },
  {...} // 3 items
]
```

Figure 1. Showcase of the OMA REST API documentation page, with an example of the interactive query and response.

The open source code is hosted at <https://github.com/DessimozLab/OmaDB/>. The package requires R version ≥ 3.6 and Bioconductor version ≥ 3.9 , as well as a stable internet connection.

Package Installation

```
if (!requireNamespace("BiocManager"))
  install.packages("BiocManager")
BiocManager::install("OmaDB")
#load the package
library(OmaDB)
```

omadb Python package

For Python users, we provide an analogous package also named *omadb*. Results are supplied to users as a hybrid attribute-dictionary object. As such, both attribute and key-based access is possible. Where the URL of a further API call is listed in a response, this has been designed to be automatically requested for the user.

For data that can be represented as a table, the *pandas* package¹⁵ is supported. HOGs can be analysed or displayed using the *pyham* library¹⁶. Trees are retrievable as *DendroPy*¹⁷ or *ETE3*¹⁸ Tree objects. Gene Ontology enrichment analyses are possible through the use of the *goatools* package¹⁹.

The open source code is hosted at <https://github.com/DessimozLab/pyomadb/>. The package requires Python ≥ 3.6 , as well as a stable internet connection. It is also available to download from PyPI, installable using pip.

Package Installation

```
# Install in shell, using pip
$ pip install omadb

# In Python, load the package
>>> from omadb import Client
# Initialise the client
>>> c = Client()
```

Results

We provide six illustrative examples in R. The first shows a direct call to the REST API, while the other five showcase the OmaDB R library. These examples are also available as a Jupyter notebook²⁰ as part of the OmaDB R code repository. We have also provided analogous examples in Python, also in the form of a Jupyter notebook, included in its code repository—with the exception of Example 6, which uses a package only available in R.

Example 1 - Simply accessing the API, in R, via URLs

One way to access the API is to directly send a request using `httr`²¹ in R. This approach requires the user to know the URL of the API endpoint, as well as the URL of the API function of interest. Some additional processing steps of the resultant response is usually needed. A simple example to retrieve information on the P53_RAT protein is provided below.

```
library(httr)

url <- "https://omabrowser.org/api/protein/P53_RAT/"
response <- GET(url)

response_content_list <- httr::content(response, as = "parsed")
```

Example 2 - Using a sequence to find its gene family (Hierarchical Orthologous Group) and function via gene ontologies

Below is a simple workflow using the OmaDB package to annotate a given protein sequence, using the `mapSequence()` function.

```
library(OmaDB)

sequence <- 'MKLVFLVLLFLGALGLCLAGRRRSVQCAVSQPEATKCFQWRNMRKVRGPPVSCIKRDSPIQCIQA
IAENRADAVTLDDGGFIYEAGLAPYKLRPVAAEVYGTERRQPRTHYAVAVVKKGGSFQLNELQGLKSCHTGLRRRTA
GWNVPIGTLRPFNLWTGPPPEIEAAVARFFSASCVPADKGFQPNLCRLCAGTGKCAFSSQEPYFYSYSGA
FKCLRDGAGDVAFIRESTVFEDLSDEAERDEYELLCPDNTRKPKVDFKDKDCHLARVPSHAVVARSVNGKEDAI
WNLRLQAQEKFGKDKSPKQFLFGSPSQKDLLFKDSAIGFSRVPFRIDSGLYLGSYFTAIQNLKSEEEVA
ARRARVWCAVGEQELRKNQWSGLSEGSVTCSSASTTEDICIALVLKGEADAMSLDGGYVYTAGKCGLVPVL
AENYKSQQSSDPDNCVDRPVEGYLAVAVRRSDTSLTWNVSKGKKSCHTAVDRTAGWNI PMGLLFNQTGSC
KFDEYFSQSCAPGSDPRSNLALCALCIGDEQGENKCVNSNERYYGYTGAFRCLAENAGDVAFVKDVTVLQNTD
GNNNEAWAKDLKLDLDFALLCLDGKRKPVTEARSCHLAMAPNHAVVSRMDKVERLKQVLLHQQAKFGRNGSDC
PDKFCLFQSETKNLLFNDNTECLARLHGKTTYEKYLGPPQYVAGITNLKCKSTSPLEACEFLRK '
```

```
seq_annotation <- mapSequence(sequence)
```

In this example, the sequence mapping identified one target sequence. From the `seq_annotation` object further information can be obtained as follows:

```
seq_annotation$targets[[1]]$canonicalid # 'TRFL_HUMAN'
```

Thus, our sequence is human lactotransferrin (also known as lactoferrin). Lactotransferrin is one of four subfamilies of transferrins in mammals²².

To investigate the evolutionary history of genes more precisely, we turn to Hierarchical Orthologous Groups (HOGs)—sets of genes which have descended from a single common ancestral gene within a taxonomic range of interest¹⁰. For an introduction to HOGs, we refer the interested reader to the following short video: <https://youtu.be/5p5x5gzZa>.

By knowing the ID of the HOG to which our sequence belongs, we can obtain a list of all the HOG members (i.e. all genes in the HOG), as follows:

```
hog_id <- seq_annotation$targets[[1]]$oma_hog_id # 'HOG:0413862.1a.1b'
hog <- getHOG(id = hog_id, members = TRUE, level = 'Mammalia')
hog$members
```

Note that it is also possible to access information on a HOG using the ID of one of its members. Therefore the below will produce the same output.

```
hog <- getHOG(id = 'TRFL_HUMAN', members = TRUE, level = 'Mammalia')
```

We can easily retrieve the Gene Ontology (GO) terms²³ that are associated to each of the members using OmaDB.

```
go_annotatons <- getProtein(hog$members$omaid,
  attribute = 'gene_ontology')
```

The resultant list of GO terms per gene is in the “geneID2GO” format by default, which is used by the topGO¹¹ package.

To compare the function of lactotransferrins with their paralogous counterparts, we can retrieve a background set consisting of all members of the transferring HOG defined at the root of the eukaryotes

```
bgHOG <- getHOG(id = 'TR_HUMAN', members = TRUE, level = 'Eukaryota')
bgAnnot <- getProtein(bgHOG$members$omaid, attribute = 'gene_ontology')
```

We can now construct a topGO object using the getTopGO function as seen below. Note that the background set of terms is set by getTopGO to all terms appearing in the list of annotations. This may not be appropriate in all cases—the choice of background set requires careful consideration²⁴.

```
bgAnnotFormatted = formatTopGO(bgAnnot, format = 'geneID2GO')
library(topGO)
myGO <- getTopGO(annotations = bgAnnotFormatted, format = 'geneID2GO',
  foregroundGenes = hog$members$entry_nr, ontology = 'BP')
myRes <- runTest(myGO, algorithm = 'classic', statistic = 'fisher')
print(GenTable(myGO, myRes))
```

As the output in [Table 1](#) indicates, several enriched terms in the mammalian lactotransferrin are related to bone formation, consistent with previous reports in the literature (e.g. 25). So is the role of lactotransferrin in antimicrobial activity (e.g. 26).

Example 3 - Taxonomic tree visualisation

The taxonomic data obtained using the OmaDB package can easily be plugged into ggtree¹³ for phylogenetic tree visualisation. First, the tree is obtained using the getTaxonomy() function. In this example, the tree is rooted at the Hominoidea taxonomic level. The default format of the object returned is newick.

```
tax <- getTaxonomy(root = 'Hominoidea')
```

Table 1. Gene Ontology enrichment of Biological Process terms associated with mammalian lactotransferrins compared to all eukaryotic transferrins, as obtained from example 2.

GO.ID	Term	P-value
GO:0001501	skeletal system development	<1e-30
GO:0001503	ossification	<1e-30
GO:0001649	osteoblast differentiation	<1e-30
GO:0001816	cytokine production	<1e-30
GO:0001817	regulation of cytokine production	<1e-30
GO:0001818	negative regulation of cytokine production	<1e-30
GO:0002237	response to molecule of bacterial origin	<1e-30
GO:0002682	regulation of immune system process	<1e-30
GO:0002683	negative regulation of immune system process	<1e-30
GO:0002761	regulation of myeloid leukocyte differentiation	<1e-30

The resultant object can directly be used to build a phylogenetic tree using the ggtree package as below:

```
library(ggtree)
tree <- getTree(tax$newick)
mytree <- ggtree(tree)
```

The tree can be further annotated using species silhouettes from PhyloPic (<http://phylopic.org/>). This functionality is already enabled within the ggtree package and just requires obtaining the relevant image codes. The workflow to produce Figure 2 is below.

```
library(rphylopic)
labels <- tree$tip.label
labelsFormatted <- sapply(labels, FUN = function(x)
  gsub("_", " ", x, fixed = TRUE))
ids <- sapply(labelsFormatted, FUN = function(x)
  name_search(x)$canonicalName[1,1])
images <- sapply(as.character(ids), FUN = function(x)
  tryCatch(name_images(x)$same[[1]]$uid, error =
  function(w) name_images(x)$supertaxa[[1]]$uid))
d <- data.frame(label = labels, images = as.character(images))
library(dplyr)
library(ggimage)
mytree %<+% d + geom_tiplab(aes(image = images), geom = 'phylopic',
  offset = 2.3, color = 'steelblue') + geom_tiplab(offset = 0.3)
+ ggplot2::xlim(0, 7)
```

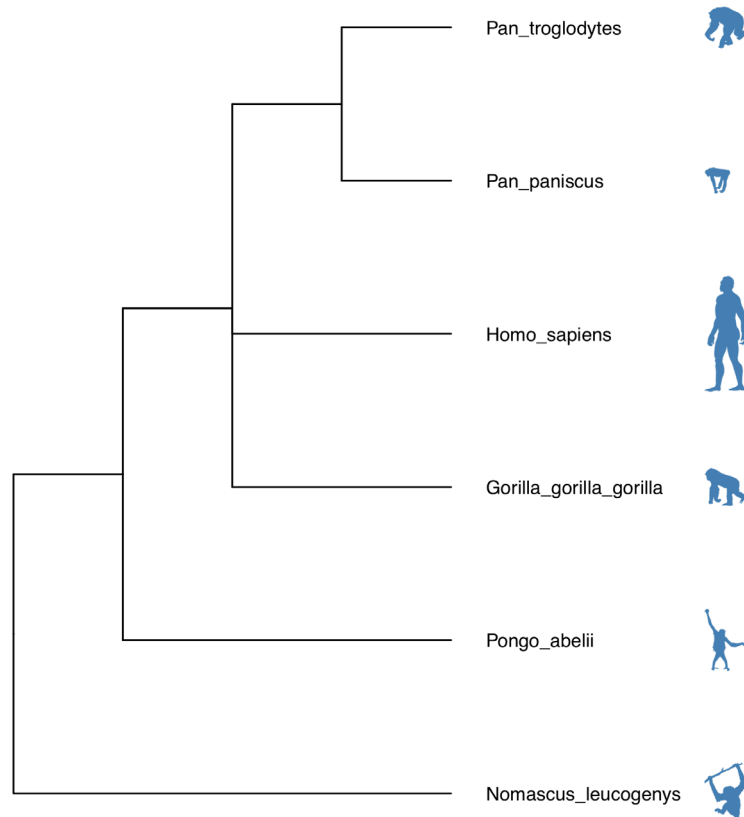


Figure 2. Species taxonomy tree obtained using example 3.

Example 4 - Visualising the distribution of PAM distances in the taxonomic space

To obtain all orthologous pairs between two genomes, we can use the `getGenomePairs()` function. To limit server load, the resultant response is paginated and by default only returns the first page, capped at 100 entries. This is easily adjustable by setting the `'per_page'` parameter to either the number of orthologs required or simply to `'all'`.

In this example, we compare the distribution of PAM distances (Point accepted mutations; 27) between orthologs of two species-pairs, namely human-dog and human-mouse. First, we request the required data:

```
mouse_id = getGenome(id='Mus musculus')$taxon_id
human_id = getGenome(id='Homo sapiens')$taxon_id
dog_id = getGenome(id='Canis lupus familiaris')$taxon_id

human_mouse <- getGenomePairs(genome_id1 = human_id,
                             genome_id2 = mouse_id, rel_type = '1:1')

human_dog <- getGenomePairs(genome_id1 = human_id,
                             genome_id2 = dog_id, rel_type = '1:1')
```

We can then bind the two resultant data frames and plot the results (Figure 3), as so:

```
human_mouse$Species <- 'Mus musculus'
human_dog$Species <- 'Canis lupus familiaris'

all_pairs <- rbind(human_mouse, human_dog)
all_pairs$Species <- as.factor(all_pairs$Species)

library(ggplot2)

g <- ggplot(all_pairs, aes(x = distance, fill = Species)) +
  geom_density(alpha = 0.5) +
  xlab('evolutionary distance [PAM]') +
  theme(legend.position = 'bottom', panel.grid.major =
        element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour =
        'black'))
print(g)
```

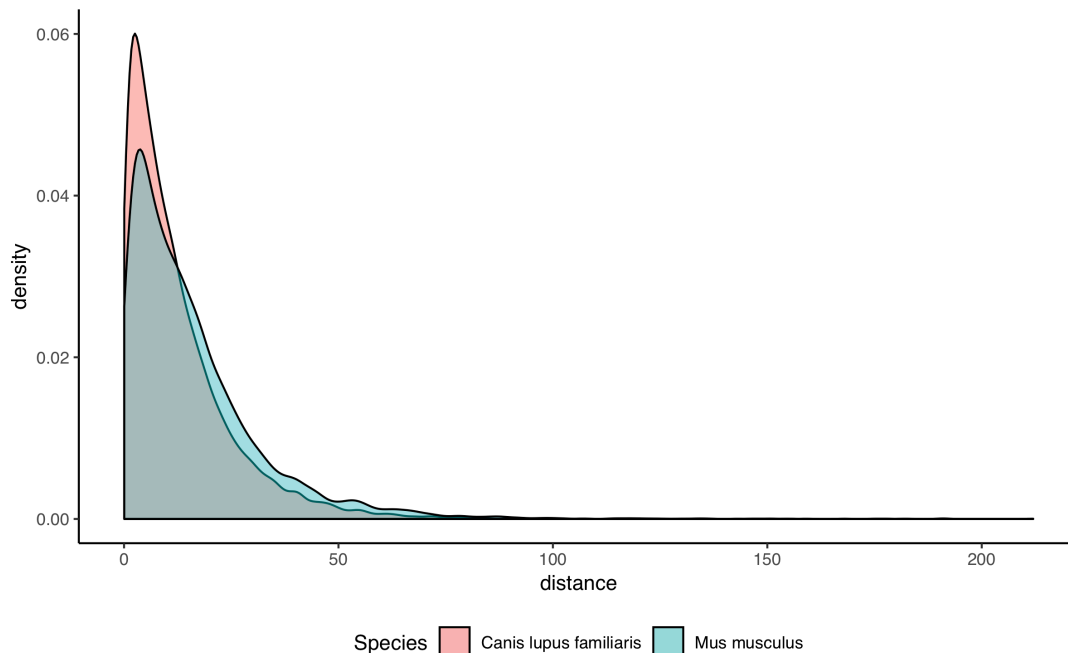


Figure 3. Distribution of evolutionary distances (in PAM units; 27) human-dog (red) and human-mouse (blue) pairs, obtained using example 4.

The two-sample Kolmogorov-Smirnov test can be performed on the two distributions, using the command:

```
ks.test(human_dog$distance, human_mouse$distance)
```

This returns p-value < 2.2e-16. The median distance between dog and human is shorter than that of mouse and human (8.8 vs. 11.8). This is consistent with previous observations that the rodent has a longer branch than humans and carnivores, in part due to their shorter generation time²⁸.

Example 5 - Annotating protein sequences not present in OMA

Although the OMA database currently analyses over 2,100 genomes, many more have been sequenced, and the gap keeps on widening. It is nevertheless possible to use OMA to infer the function of custom protein sequences through a fast approximate search against all sequences in OMA⁴.

```
# Our mystery sequence is cystic fibrosis transmembrane conductance
# regulator in the Emperor penguin (UniProt ID: A0A087RGQ1_APTFO)
mysterySeq <-
'FFFLLRWTKPIILRKGYRRRLELSDIYQIPADSADNLSEKLEREWRELATSKKKPKLINALRRCFFWKFMFYGIIL
YLGEVTKSVQPLLLGRIIASYDPDNDERSIAYYLAIGLCLLFLVRLTLLIHPAIFGLHHIGMQMRIAMFSLIYKIKL
LSSRVLDKISTGQLVLSLLSNLNKFDEGLALAHFVWIAPLQVALLMGLLWDMLEASAFSGLAFLIVLAFFQAWLGQRM
MKYRNKRAGKINERLVITSEIIENIQSVKAYCWEDAMEKMIESIRETELKLRKAAAYVRYFNSSAFFSFFVFLAV
LPYAVIKGIILRKIFTTISFCIVLRMTVTRQFPGSVQTYWDSIGAINKIQDFLLKKEYKSLEYNLTTTGVELDKVTA
WDEGIGELFVKANQENNSKAPSTDNNLFFSNFPLHASPVLDINFKIEKGQLLAVSGSTGAGKTSLLMLIMGELEPS
QGRLLKHSGRISFSPQVSWIMPGTIKENIIFGVSYDEYRYKSVIKACQLEEDISKFPDKDYTVLGDGGIILSGGQRARI
SLARAVYKDADLYLLDSPFGHLDFTEKEIFESCVCCKLMANKTRILVTSKLEHLKIADKILILHEGSCYFYGTSELQ
GQRPDFSSELMGFDSFDQFSAERRNSILTETLRRFSIEGEGTGSRNEIKKQSFQKTSDFNDKRRKNSIIINPLNASRKF
SVVQRNGMQVNGIEDGHNDPPERFSLVPDLEQGDVGLLRSSMLNTDHILOGRRRQSVLNLMTGTSVNYGPNFSKKS
TFRKMSMVPQTNLSEIDIYTRRLSRDSVLDITDEINEEDLKECFTDDAESMGTVTTWNTYFRYVTIHKNLIFVLIL
CVTVFLVEVAASLAGLWFLKQTALKANTTQSENSTSDKPPVIVTVTSSYYIIYIVGVADTLLAMGIFRGLPLVHTLI
TVSKTLHQKMHAVLHAPMSTFNSWKAGMLNRFKSDTAVLDDLLPLTVDFIQLILIVIGAITVVSILQPYIFLASV
PVIAAFILLRAYFLHTSQQLKQLESEARSPIFTHLVTSLKGLWTLRAFRQPYFETLFHKALNLHTANWFLYLSTLRW
FQMRIEMIFVVFVAVAFISIVTTGDGSGKVGIIITLAMNIMGTQWAVNSSIDVDSLMSVGRIFKFDMPTEEMKN
IKPHKNNQFSDALVIENRHAKEEKNWPSGGQMTVKDLTAKYSEGGAAVLENISFSSISGQRVGLLGRGTSGKSTLLFA
FLRLNTEGDIQIDGVSWSTVSVQQRKAFGVIPOKVFIFSGTFRMNLDPYQWVNDDEIWKVAEEVGLKSVIEQFPGQ
LDFVLVDGGCVLSHGKQLMCLARSVLSKAKILLDEPSAHLDPVTSQVIRKTLKHAFANCTVILSEHRLEAMLECQR
FLVIEDNKLRQYESIQKLLNEKSSFRQAI SHADRLKLLPVHHRNSSKRKPRPKITALQEETEEVEVQETRL'
```

```
myAnnotations <- getAnnotation(mysterySeq)
```

This results in 54 GO annotations. By comparison, this sequence has merely 15 GO annotations in UniProt-GOA²⁹ — all of which are also predicted by this method in OMA.

Example 6 - Combining OmaDB with BgeeDB for gene expression

We go back to the lactotransferrin gene family from Example 2. We can use OmaDB in conjunction with the BgeeDB Bioconductor package³⁰ to retrieve expression data from the Bgee database³¹ as follows.

```
BiocManager::install("BgeeDB")
library(BgeeDB)

# Bgee uses Ensembl gene IDs, obtainable using OmaDB's cross-references.
trfl_xrefs <- getProtein(id='TRFL_HUMAN')$xref
trfl_ens_id <- subset(trfl_xrefs, source == 'Ensembl Gene')$xref
# The Ensembl gene IDs need to be without version suffix
trfl_ens_id <- strsplit(trfl_ens_id, '.', fixed=TRUE)[[1]][1]

my_stage <- 'UBERON:0034920' # Infant stage
bgee.expr <- Bgee$new(species='Homo_sapiens')
expr.data <- loadTopAnatData(bgee.expr, stage = my_stage)
gene.expr.tissue.ids <-
  unlist(expr.data$gene2anatomy[trfl_ens_id], use.names = F)
tissues <- expr.data$organ.names
print(tissues[tissues$ID %in% gene.expr.tissue.ids, ])
```

Among the tissues in which lactotransferrin is expressed according to Bgee (Table 2), we note the bone marrow and the palpebral conjunctiva (the eyelid inner surface). This is consistent with the aforementioned involvement of lactotransferrin in bone formation and anti-microbial activity.

Table 2. Human tissues in which lactotransferrin is expressed in infant stage, according to the Bgee database version 14 (output of Example 6).

ID	Name
UBERON:0001812	palpebral conjunctiva
UBERON:0000178	blood
UBERON:0002371	bone marrow
UBERON:0001154	vermiform appendix
UBERON:0002084	heart left ventricle

Further tutorials on the OmaDB package can be found in the accompanying vignettes:

```
browseVignettes('OmaDB')
```

Discussion and outlook

Orthology is used for various purposes, such as species tree inference, gene evolution dynamic, or protein function prediction. The retrieval of orthologs is thus typically just the starting point of a larger analysis. Therefore, this overhaul and expansion of the OMA programmatic interface will facilitate the incorporation of OMA data in such larger analyses

Our R package will continue to be maintained in line with the biannual Bioconductor releases. Further work to improve the package includes improvement in performance. For example, the responses are currently fully loaded into an R object of choice which, depending on the response size, may create some time lag in the response. We will also continue to update the package and API to incorporate new functionalities of OMA, such as support for local synteny which is currently under development.

Likewise, we will also maintain and further develop the Python package. In particular, we will explore the possibility of further integration with the BioPython library³².

More generally, in OMA we will keep supporting the various ways of accessing the underlying data, including the interactive web browser and flat files in a variety of formats. The REST API is also complemented by a new SPARQL interface that enables highly specific queries, as well as federated queries over multiple resources⁴. However, the query language is more complex.

We very much welcome feedback and questions from the community. We also highly appreciate contributions to the code in the form of pull requests. Our preferred channel for support is the BioStar website³³, where we monitor all posts with keyword “oma”.

Software availability

Please note that this manuscript uses version 2.0 of the OmaDB R package, which is in the **development version** of Bioconductor (v.3.9). Until the release of Bioconductor v.3.9 in Spring 2019, there are two possible ways of installing it:

- 1) Install the development version of R (v.3.6) — required for Bioconductor v.3.9 — and install OmaDB using the command:

```
BiocManager::install('OmaDB', version = 'devel')
-or-
```

- 2) Install OmaDB 2.0 directly from the github repo using the devtools R package:

```
install.packages('devtools')
library(devtools)
install_github('dessimozlab/omadb')
```

REST API available from: <https://omabrowser.org/api>

Documentation available from: <https://omabrowser.org/api/docs>

R OmaDB package available from: <http://bioconductor.org/packages/OmaDB/>

Source code available from: <https://github.com/DessimozLab/OmaDB/>

Archived source code as at time of publication: <http://doi.org/10.5281/zenodo.2530253>³⁴

License: GPL-2

omadb Python package available from: <https://pypi.org/project/omadb/>

Source code available from: <https://github.com/DessimozLab/pyomadb/>

Archived source code as at time of publication: <http://doi.org/10.5281/zenodo.2530250>³⁵

License: LGPL-3

Grant information

We acknowledge support by Swiss National Science Foundation grant 150654, UK BBSRC grant BB/M015009/1, the Swiss State Secretariat for Education, Research and Innovation (SERI), as well as a UCL Genetics, Evolution and Environment Departmental Summer Bursary (to KK).

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Acknowledgements

We thank Natasha Glover for helpful feedback on the manuscript, and Frédéric Bastian for help on the example involving BgeeDB.

References

- Fitch WM: **Distinguishing homologous from analogous proteins.** *Syst Zool.* 1970; **19**(2): 99–113.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Sonnhammer EL, Gabaldón T, Sousa da Silva AW, *et al.*: **Big data and other challenges in the quest for orthologs.** *Bioinformatics.* 2014; **30**(21): 2993–8.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Forslund K, Pereira C, Capella-Gutiérrez S, *et al.*: **Gearing up to handle the mosaic nature of life in the quest for orthologs.** *Bioinformatics.* 2017.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Altenhoff AM, Glover NM, Train CM, *et al.*: **The OMA orthology database in 2018: retrieving evolutionary relationships among all domains of life through richer web and programmatic interfaces.** *Nucleic Acids Res.* 2018; **46**(D1): D477–85.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Schmitt T, Messina DN, Schreiber F, *et al.*: **Letter to the editor: SeqXML and OrthoXML: standards for sequence and orthology information.** *Brief Bioinform.* 2011; **12**(5): 485–8.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Han MV, Zmasek CM: **phyloXML: XML for evolutionary biology and comparative genomics.** *BMC Bioinformatics.* 2009; **10**: 356.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Huber W, Carey VJ, Gentleman R, *et al.*: **Orchestrating high-throughput genomic analysis with Bioconductor.** *Nat Methods.* 2015; **12**(2): 115–21.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Django Software Foundation.** Django. [cited 2018].
[Reference Source](#)
- Folk M, Heber G, Koziol Q, *et al.*: **An overview of the HDF5 technology suite and its applications.** *Proceedings of the EDBT.* 2011.
[Publisher Full Text](#)
- Altenhoff AM, Gil M, Gonnet GH, *et al.*: **Inferring hierarchical orthologous groups from orthologous gene pairs.** *PLoS One.* 2013; **8**(1): e53786.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- Alexa A, Rahnenfuhrer J: **topGO: Enrichment analysis for Gene Ontology.** R package version 2.28.0. *Bioconductor.* 2016.
- Pagès H, Aboyou P, Gentleman R, *et al.*: **Biostrings: Efficient manipulation of biological strings.** R Package Version. 2017; **2**(0).
[Reference Source](#)
- Yu G, Smith DK, Zhu H, *et al.*: **ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data.** McInerney G editor. *Methods Ecol Evol.* 2017; **8**(1): 28–36.
[Publisher Full Text](#)
- Lawrence M, Huber W, Pagès H, *et al.*: **Software for computing and annotating genomic ranges.** *PLoS Comput Biol.* 2013; **9**(8): e1003118.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
- McKinney W: **pandas: a foundational Python library for data analysis and statistics.** *Python for High Performance and Scientific Computing.* 2011; 1–9.
[Reference Source](#)
- Train C-M, Pignatelli M, Altenhoff A, *et al.*: **iHam & pyHam: visualizing and processing hierarchical orthologous groups.** *Bioinformatics.* 2018.
[PubMed Abstract](#) | [Publisher Full Text](#)
- Sukumaran J, Holder MT: **DendroPy: a Python library for phylogenetic computing.** *Bioinformatics.* 2010; **26**(12): 1569–71.
[PubMed Abstract](#) | [Publisher Full Text](#)

18. Huerta-Cepas J, Serra F, Bork P: **ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data.** *Mol Biol Evol.* 2016; **33**(6): 1635–8.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
19. Klopfenstein DV, Zhang L, Pedersen BS, *et al.*: **GOATOOLS: A Python library for Gene Ontology analyses.** *Sci Rep.* 2018; **8**(1): 10872.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
20. Kluyver T, Ragan-Kelley B, Pérez F, *et al.*: **Jupyter Notebooks—a publishing format for reproducible computational workflows.** In: *ELPUB.* 2016; 87–90.
[Publisher Full Text](#)
21. Wickham H: **httr: Tools for Working with URLs and HTTP.** 2018.
[Reference Source](#)
22. Lambert LA, Perri H, Meehan TJ: **Evolution of duplications in the transferrin family of proteins.** *Comp Biochem Physiol B Biochem Mol Biol.* 2005; **140**(1): 11–25.
[PubMed Abstract](#) | [Publisher Full Text](#)
23. Ashburner M, Ball CA, Blake JA, *et al.*: **Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.** *Nat Genet.* 2000; **25**(1): 25–9.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
24. Gaudet P, Dessimoz C: **Gene Ontology: Pitfalls, Biases, and Remedies.** *Methods Mol Biol.* In: Dessimoz C, Škunca N, editors. *The Gene Ontology Handbook.* New York, NY: Springer New York; 2017; 189–205.
[PubMed Abstract](#) | [Publisher Full Text](#)
25. Naot D, Grey A, Reid IR, *et al.*: **Lactoferrin—a novel bone growth factor.** *Clin Med Res.* 2005; **3**(2): 93–101.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
26. Orsi N: **The antimicrobial activity of lactoferrin: current status and perspectives.** *Biometals.* 2004; **17**(3): 189–96.
[PubMed Abstract](#) | [Publisher Full Text](#)
27. Dayhoff MO, Schwartz RM, Orcutt BC: **A model of evolutionary change in proteins.** In: *Atlas of Protein Sequence and Structure.* 1978; 345–52.
[Reference Source](#)
28. Easteal S: **Generation time and the rate of molecular evolution.** *Mol Biol Evol.* 1985; **2**(5): 450–3.
[PubMed Abstract](#) | [Publisher Full Text](#)
29. Huntley RP, Sawford T, Mutowo-Meullenet P, *et al.*: **The GOA database: gene Ontology annotation updates for 2015.** *Nucleic Acids Res.* 2015; **43**(Database issue): D1057–63.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
30. Komljenovic A, Roux J, Wollbrett J, *et al.*: **BgeeDB, an R package for retrieval of curated expression datasets and for gene list expression localization enrichment tests [version 2; referees: 2 approved, 1 approved with reservations].** *F1000Res.* 2016; **5**: 2748.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
31. Bastian F, Parmentier G, Roux J, *et al.*: **Bgee: Integrating and Comparing Heterogeneous Transcriptome Data Among Species.** In: *Data Integration in the Life Sciences.* (Lecture Notes in Computer Science). Springer Berlin Heidelberg. 2008; 124–31.
[Publisher Full Text](#)
32. Cock PJ, Antao T, Chang JT, *et al.*: **Biopython: freely available Python tools for computational molecular biology and bioinformatics.** *Bioinformatics.* 2009; **25**(11): 1422–3.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
33. Parnell LD, Lindenbaum P, Shameer K, *et al.*: **BioStar: an online question & answer resource for the bioinformatics community.** *PLoS Comput Biol.* 2011; **7**(10): e1002216.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
34. klarakaleb, Altenhoff A, bioc-gitadmin, *et al.*: **DessimozLab/OmaDB: v1.99.1 (Version 1.99.1).** *Zenodo.* 2019.
<http://www.doi.org/10.5281/zenodo.2530253>
35. Alex WV, Altenhoff A: **DessimozLab/pyomadb: v2.0.0 (Version 2.0.0).** *Zenodo.* 2019.
<http://www.doi.org/10.5281/zenodo.2530250>

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research