BITACORA:

A comprehensive tool for the identification and annotation of gene families in genome assemblies

Joel Vizueta
Alejandro Sánchez-Gracia
Julio Rozas

Departament de Genètica, Microbiologia i Estadística
Institut de Recerca de la Biodiversitat (IRBio)

Universitat de Barcelona

http://www.ub.es/softevol/bitacora

March 18th, 2020

- Overview

Genome annotation is a critical bottleneck in genomic research, especially for the comprehensive study of gene families in the genomes of non-model organisms. Despite the recent progress in automatic annotation, state-of-the-art tools used for this task often produce inaccurate annotations, such as fused, chimeric, partial or even completely absent gene models for many family copies, errors that require considerable extra efforts to be corrected. Here we present BITACORA, a bioinformatics tool that integrates popular sequence similarity-based search algorithms and Perl scripts to facilitate the curation of these inaccurate annotations and the identification of previously undetected gene family copies directly from genomic DNA sequences. The program creates general feature format (GFF) files, with both curated and newly identified gene models, and FASTA files with all predicted proteins. The output of BITACORA can be easily integrated in genomic annotation editors, greatly facilitating subsequent manual annotation and downstream analyses.

Authors

Joel Vizueta	jvizueta@ub.edu
Alejandro Sánchez-Gracia	elsanchez@ub.edu
Julio Rozas	jrozas@ub.edu

BITACORA Publication

Vizueta, J., Sánchez-Gracia, A. and Rozas, J. 2019. BITACORA: A comprehensive tool for the identification and annotation of gene families in genome assemblies. *bioRxiv*. https://doi.org/10.1101/593889.

BITACORA Web Site

www.ub.edu/softevol/bitacora

Current version: 1.2

0 Workflow & Contents

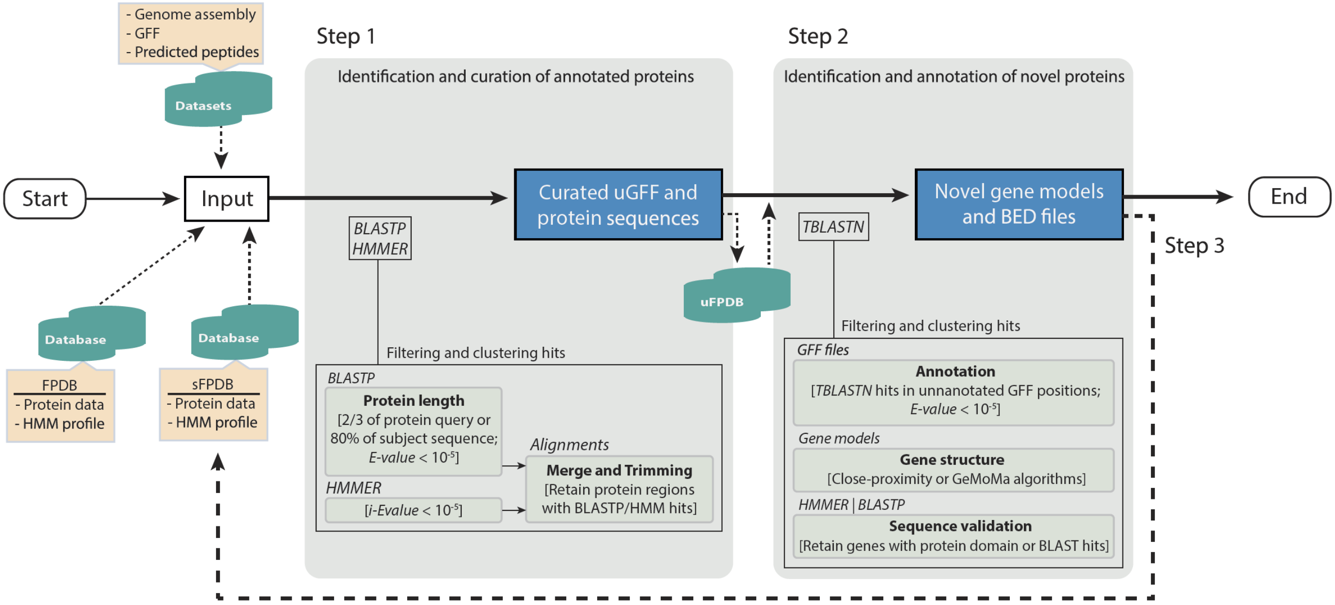

- Genome assembly
- GFF
- Predicted peptides
Datasets
Start
Input
End
Database
FPDB
- Protein data
- HMM profile
Database
sFPDB
- Protein data
- HMM profile
Step 1
Identification and curation of annotated proteins
Curated uGFF and protein sequences
BLASTP
HMMER
Filtering and clustering hits
BLASTP
Protein length
[2/3 of protein query or 80% of subject sequence; E-value < 10-5]
HMMER
[i-Evalue < 10-5]
Alignments
Merge and Trimming
[Retain protein regions with BLASTP/HMM hits]
uFPDB
Step 2
Identification and annotation of novel proteins
Novel gene models and BED files
TBLASTN
Filtering and clustering hits
GFF files
Annotation
[TBLASTN hits in unnanotated GFF positions; E-value < 10-5]
Gene models
Gene structure
[Close-proximity or GeMoMa algorithms]
HMMER | BLASTP
Sequence validation
[Retain genes with protein domain or BLAST hits]
Step 3

Figure 1. Workflow showing the basic steps used in BITACORA

1 Installation

BITACORA is distributed as a multiplatform shell script (`runBITACORA.sh`) that calls several other Perl scripts, which include all functions responsible of performing all pipeline tasks. Hence, it does not require any installation or compilation step.

You can download all package contents from GitHub: https://github.com/molevol-ub/bitacora

To run the pipeline, edit the master script `runBITACORA.sh` variables described in Prerequisites, Data, and Parameters.

2 Prerequisites

- **Perl**: Perl is installed by default in most operating systems. See https://learn.perl.org/installing/ for installation instructions.

- **BLAST**: Download blast executables from: ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/

- **HMMER**: The easiest way to install HMMER in your system is to type one of the following commands in your terminal:

```
% brew install hmmer                # OS/X, HomeBrew
% port install hmmer                # OS/X, MacPorts
% apt install hmmer                 # Linux (Ubuntu, Debian...)
% dnf install hmmer                 # Linux (Fedora)
% yum install hmmer                 # Linux (older Fedora)
% conda install -c bioconda hmmer   # Anaconda
```

Or compile HMMER binaries from the source code: http://hmmer.org/

HMMER and BLAST binaries require to be added to the PATH environment variable. Specify the correct path to bin folders in the master script `runBITACORA.sh`, if necessary.

```
$ export PATH=$PATH:/path/to/blast/bin
$ export PATH=$PATH:/path/to/hmmer/bin
```

- **GeMoMa**: By default, BITACORA reconstructs new gene models using the GeMoMa algorithm (Keilwagen et al., 2016; Keilwagen et al., 2018). The GeMoMa jar file (i.e. `GeMoMa-1.6.2.jar`) must be specified in `GEMOMAP` variable in `runBITACORA.sh`. GeMoMa is implemented in Java using Jstacs and can be downloaded from: http://www.jstacs.de/index.php/GeMoMa.

`GEMOMAP=/path/to/GeMoMa.jar` (within `runBITACORA.sh` script)

3 Computational requirements

BITACORA have been tested in UNIX-based platforms (both in Mac OS and Linux operating systems). Multiple threading can be set in blast searches, which is the most time-consuming step, by editing the option THREADS in `runBITACORA.sh`

For a typical good quality genome (~2Gb in size and ~10,000 scaffolds) and a standard modern PC (16Gb RAM), a full run of BITACORA is completed in less than 24h. This running time, however, will depend on the size of the gene family or the group of genes surveyed in a particular analysis. For gene families of 10 to 100 members, BITACORA spends from minutes to a couple of hours.

In case of larger or very fragmented genomes, BITACORA should be used in a computer cluster or workstation given the increase of RAM memory and time required.

4 Usage modes

4.1. Full mode

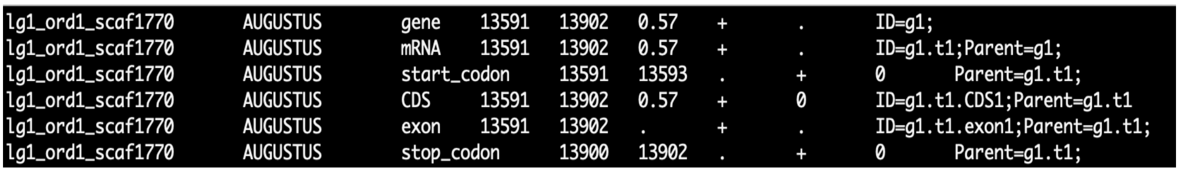
BITACORA has been designed to work with genome sequences and protein annotations (full mode). However, the pipeline can also be used either with only protein or only genomic sequences (protein and genome modes, respectively). These last modes are explained in next subsections.

Preparing the data: The input files (in plain text) required by BITACORA to run a full analysis are (update the complete path to these files in the master script `runBITACORA.sh`):

I. File with genomic sequences in FASTA format

II. File with structural annotations in GFF3 format. [*NOTE: mRNA* or *transcript*, and *CDS* are mandatory fields].

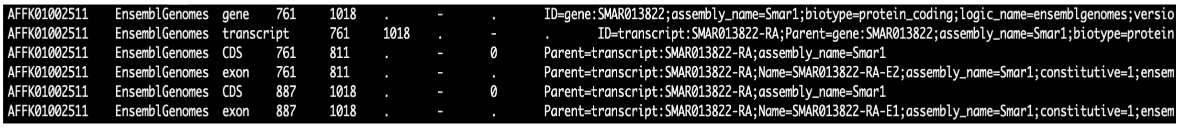
--------------------- GFF3 example



```
lg1_ord1_scaf1770       AUGUSTUS        gene    13591   13902   0.57    +       .       ID=g1;
lg1_ord1_scaf1770       AUGUSTUS        mRNA    13591   13902   0.57    +       .       ID=g1.t1;Parent=g1;
lg1_ord1_scaf1770       AUGUSTUS        start_codon     13591   13593   .       +       0       Parent=g1.t1;
lg1_ord1_scaf1770       AUGUSTUS        CDS     13591   13902   0.57    +       0       ID=g1.t1.CDS1;Parent=g1.t1
lg1_ord1_scaf1770       AUGUSTUS        exon    13591   13902   .       +       .       ID=g1.t1.exon1;Parent=g1.t1;
lg1_ord1_scaf1770       AUGUSTUS        stop_codon      13900   13902   .       +       0       Parent=g1.t1;
```

BITACORA also accepts other GFF formats, such as Ensembl GFF3 or GTF. [NOTE: GFF formatted files from NCBI can cause errors when processing the data, use the supplied script "`reformat_ncbi_gff.pl`" (located in the folder `/Scripts/Tools`) to make the file parsable by BITACORA]. See Troubleshooting in case of getting errors while parsing your GFF.

--------------------- Ensembl GFF3 example



```
AFFK01002511    EnsemblGenomes  gene    761     1018    .       -       .       ID=gene:SMAR013822;assembly_name=Smar1;biotype=protein_coding;logic_name=ensemblgenomes;versio
AFFK01002511    EnsemblGenomes  transcript      761     1018    .       -       .       ID=transcript:SMAR013822-RA;Parent=gene:SMAR013822;assembly_name=Smar1;biotype=protein
AFFK01002511    EnsemblGenomes  CDS     761     811     .       -       0       Parent=transcript:SMAR013822-RA;assembly_name=Smar1
AFFK01002511    EnsemblGenomes  exon    761     811     .       -       .       Parent=transcript:SMAR013822-RA;Name=SMAR013822-RA-E2;assembly_name=Smar1;constitutive=1;ensem
AFFK01002511    EnsemblGenomes  CDS     887     1018    .       -       0       Parent=transcript:SMAR013822-RA;assembly_name=Smar1
AFFK01002511    EnsemblGenomes  exon    887     1018    .       -       .       Parent=transcript:SMAR013822-RA;Name=SMAR013822-RA-E1;assembly_name=Smar1;constitutive=1;ensem
```

III. File with predicted proteins in FASTA format. BITACORA requires identical IDs for proteins and their corresponding mRNAs or transcripts IDs in the GFF3. [NOTE: we recommend using genes but not isoforms in BITACORA; isoforms can be removed or properly annotated after BITACORA analysis]

IV. Specific folder with files containing the query protein databases (`YOURFPDB_db.fasta`) and HMM profiles (`YOURFPDB_db.hmm`) in FASTA and hmm format, respectively, where the "YOURFPDB" label is your specific data file name. The addition of "_db" to the database name with its proper extension, fasta or hmm, is mandatory.

BITACORA requires one protein database and profile per surveyed gene family (or gene group). See Example/DB files for an example of searching for two different gene families in BITACORA: OR, Odorant Receptors; and CD36-SNMP.

[NOTE: profiles covering only partially the proteins of interest are not recommended]

Notes on HMM profiles:

HMM profiles are found in InterPro or PFAM databases associated to known protein domains. If you don't know if your protein contains any described domain, you can search in InterPro (http://www.ebi.ac.uk/interpro/) using the protein sequence of one of your queries to identify domains.

For example, for the chemosensory proteins (CSPs) in insects, you can download the HMM profile from pfam (Curation & model PFAM submenu):

http://pfam.xfam.org/family/PF03392#tabview=tab6

In the case of searching for proteins with not described protein domains, or with domains not covering most of the protein sequence, it should be performed an alignment of the query proteins to create a specific HMM profile.

Example of building a protein profile (it requires an aligner, here we use mafft as example):

```
$ mafft --auto FPDB_db.fasta > FPDB_db.aln

$ hmmbuild FPDB_db.hmm FPDB_db.aln
```

Notes on the importance of selecting a confident curated database:

The proteins included in the database (FPDB) and that will be used as query in similarity-based searches are really important. The inclusion of proteins or protein fragments unrelated with the focal gene family will lead to false positive identifications.

On the other hand, if possible, we recommend including proteins from phylogenetically close species to increase the power of similarity-based searches, particularly in fast-evolving and old gene families. If the organism of interest does not have an annotated genome of a close related species, we recommend to perform a second BITACORA round (step 3 described in the manuscript; Figure 1), by including in the query database (sFPDB) all sequences identified in the first round along with a new HMM profile built with these updated protein set.

4.2. Protein mode

BITACORA can also run on a set of proteins (e.g., proteins predicted from transcriptomic data; script `runBITACORA_protein_mode.sh`) by using the input files described in points **III** and **IV** of the section **4.1**.

Under this mode, BITACORA identifies, curates when necessary, and report all members of the surveyed family among the predicted proteins. The original protein sequences (not being curated) are also reported (located in `Intermediate_Files` if cleaning output is active).

4.3. Genome mode

BITACORA can also run on raw genome sequences (e.g., not annotated genomes; script `runBITACORA_genome_mode.sh`), by using the input files described in points **I** and **IV** of the section **4.1**.

Under this mode, BITACORA carries out a *de novo* identification of all members of the focal family and returns a BED file with the coordinates of all sequences encoding these members, a GFF3 file with structural annotations and a FASTA file with the predicted proteins.

[NOTE: By default, BITACORA applies GeMoMa software to generate accurate gene models, but it requires the additional installation of some dependencies. Otherwise, BITACORA can use the "close-proximity" algorithm to generate gene models, although these models are only semi-automatic predictions that could require further manual annotation (e.g. using genomic annotation editor). Under this mode, we highly recommend running a second iterative search round.]

5 Parameters

- The option CLEAN can be used to create the `Intermediate_files` directory where all intermediate files will be stored (see output section).

```
CLEAN=T  #T=true, F=false
```

- BLAST and HMMER hits are filtered with a default cut-off E-value of 10^{-5} (in addition to an internal parameter for filtering the length covered by the alignment).

E-value can be modified in the master script runBITACORA.sh:

```
EVALUE=10e-5    #Default
```

- Number of threads to be used in blast searches, default is 1.

```
THREADS=1    #Default
```

- BITACORA can generate new gene models (for those putative genes not included in the input GFF) using two different methods. By default (`GEMOMA=T`), BITACORA will use the GeMoMa software to predict novel genes from TBLASTN alignments (the user must specify the PATH to jar file in `GEMOMAP` variable described in prerequisites). Otherwise (`GEMOMA=F)`, BITACORA will predict new genes by exon proximity (close-proximity method described in the manuscript).

```
GEMOMA=T    #Default
```

- For the close proximity method (`GEMOMA=F` option), BITACORA uses by default an upper limit value of 15 kb to join putative exons from separate but contiguous (and in the same scaffold) genome hits to build a gene model

This value can be modified in the master script runBITACORA.sh:

```
MAXINTRON=15000    #Default
```

- New generated gene models are subsequently evaluated for the presence of the specific protein family domain using either HMMER or BLASTP searches against the proteins in FPDB (`GENOMICBLASTP=T` option). Otherwise (`GENOMICBLASTP=F`), BITACORA will only retain all gene models exhibiting the protein domain using HMMER. Despite that the first option is more sensitive, poor quality annotations in the FPDB could lead to a loss of specificity.

```
GENOMICBLASTP=F    #Default
```

Notes about using GeMoMa:

We highly recommend the use of GeMoMa (set as default) to construct more accurate gene models. This method incorporates intron position conservation as evidence in gene predictions. Nevertheless, in the presence of sequencing or assembly errors and pseudogenes (e.g. highly fragmented assemblies or point mutations either real or introduced by the sequencing process), this algorithm might fail to identify some genes; in these cases, the close proximity method is able to report these gene models regardless of whether they are true or artifactual pseudogenes or fragmented copies. Therefore, we encourage the user to apply both methods in order to ensure the identification of all gene family copies, functional or not.

Notes on the parameter MAXINTRON:

Estimating the intron length distribution in your genome:

MAXINTRON is a key parameter affecting the quality of gene models built using the close-proximity algorithm in BITACORA step 2. BITACORA is distributed with a script (`get_intron_size_fromgff.pl`) that computes the mean, median, and 95% and 99% upper limits of intron length in a specific genome (from the GFF of the genome) or for a specific gene family (from the GFF generated in BITACORA output).

As default, BITACORA uses a (conservative) high intron length for the algorithm, in order to ensure joining all exons of a same gene. However, a large value of MAXINTRON parameter can generate gene fusions. These possible gene fusions are tagged with the label "`Ndom`" in the corresponding proteins in the output file, being N the number of predicted domains (i.e., likely different genes).

The number of predicted gene fusions identified *de novo* (i.e, in not annotated DNA regions) can be obtained using the following command in the terminal:

```
$ grep '>.*dom' DB/DB_genomic_and_annotated_proteins_trimmed.fasta
```

6 Running BITACORA

After preparing the data as indicated in sections 4 (Usage) and 5 (Parameters), you can execute BITACORA with the following command:

```
$ bash runBITACORA.sh
```

7 Output

BITACORA creates an output folder for each query database, and three files with the number of proteins identified in each step, including a summary table. For the genome and protein modes, only one summary table will be reported with the number of identified genes.

In each folder, there are the following **main files** (considering you chose to clean output directory. If not, all files will be found in the same output folder):

- `YOURFPDB_genomic_and_annotated_genes_trimmed.gff3`: GFF3 file with information of all curated models.

- `YOURFPDB_genomic_and_annotated_proteins_trimmed.fasta`: A fasta file containing the protein sequences corresponding to the curated models.

Non-redundant data: Relevant information excluding identical proteins or those considered as false positives (e.g. duplicated scaffolds, isoforms…).

- `YOURFPDB_genomic_and_annotated_genes_trimmed_nr.gff3`: GFF3 file containing all non-redundant curated models.

- `YOURFPDB_genomic_and_annotated_proteins_trimmed.fasta`: A fasta file containing the non-redundant protein sequences corresponding to the curated models.

BED files with the location of all putative identified exons from TBLASTN hits in the genome sequence:

- `YOURFPDBtblastn_parsed_list_genomic_positions.bed`: BED file with genomic coordinates of putative exons located in non-annotated regions (i.e. absent in the input GFF).

- `YOURFPDBtblastn_parsed_list_genomic_positions_nogff_filtered.bed`: BED file with the genomic coordinates of all putative exons.

Intermediate files: BITACORA generates the following Intermediate files (located into `Intermediate_files` folder if `CLEAN=T`). These files contain information of some intermediate steps of the analysis, such as the **original or untrimmed gene models**, and multiples files stored for debugging or as controls:

-`YOURFPDB_annot_genes.gff3` and `YOURFPDB_proteins.fasta`: GFF3 and FASTA file containing the original untrimmed models for the identified proteins in the input GFF (from BITACORA Step 1).

- `YOURFPDB_annot_genes_trimmed.gff3` and `YOURFPDB_proteins_trimmed.fasta`: GFF3 and fasta containing only the curated model for the identified annotated proteins (trimming exons if not aligned to query FPDB sequences or splitting putative fused genes in Step 1).

- YOURFPDB_genomic_genes.gff3: GFF3 containing newly identified and untrimmed proteins in genomic sequences (from BITACORA Step 2).

- YOURFPDB_genomic_genes_trimmed.gff3: GFF3 containing newly identified proteins in the genomic mode, curated by the positions identified in the HMM profile (from Step 2).

- Hmmer folder containing the output of HMMER searches against the annotated proteins and novel proteins identified in the genome.

- GeMoMa folder containing the obtained raw gene models.

- YOURFPDBgfftrimmed.cds.fasta and YOURFPDBgfftrimmed.pepfasta: Files containing CDS and protein sequences translated directly from YOURFPDB_annot_genes_trimmed.gff3 (used for debugging purposes)

- YOURFPDBgffgenomictrimmed.cds.fasta and YOURFPDBgffgenomictrimmed.pep.fasta: Files containing CDS and protein sequences translated directly from YOURFPDB_genomic_genes_trimmed.gff3

- YOURFPDB_blastp.outfmt6: BLASTP output of the search of the query FPDB against the annotated proteins.

- YOURFPDB_tblastn.outfmt6: TBLASTN output of the search of the query FPDB against the genomic sequence.

- YOURFPDB_blastp_parsed_list.txt; YOURFPDB_hmmer_parsed_list.txt; YOURFPDB_allsearches_list.txt; YOURFPDB_combinedsearches_list.txt: Parsed coordinate files combining all hits from BLASTP and HMMER outputs.

- YOURFPDB_tblastn_parsed_list_genomic_positions.txt (and _notgff_filtered) : File containing the positions identified after parsing the TBLASTN search.

- YOURFPDB_prots_VsGFF_badannot_list.txt and YOURFPDB_goodannot_list.txt: Debugging files: These files are for checking that the identified proteins and the protein models in the GFF3 codify the same protein. If the file badannot_list.txt contains some identifier, it means that the GFF3 annotation is incorrect pointing to a bad annotation in the original GFF3. Please, try to translate the CDS for that protein into the 3 reading frames and check if the 2nd or 3rd frame codify for the protein in question stored in "YOURFPDB_genomic_and_annotated_proteins_trimmed_nr.fasta". If correct, modify the GFF3 by adding 1 or 2 nucleotide position in the start of the GFF3 (take into account if it is transcribed from forward or reverse strand). If negative, please report the error via GitHub.

- YOURFPDB_genomic_genes_proteins.fasta: It contains all merged exons from putative novel proteins identified in the genome before filtering those without the protein domain identified with HMMER. This file could be useful in case of using an HMM profile not trained with your sequences which cannot detect divergent sequences, such as remote homologs.

- YOURFPDB_genomic_exon_proteins.fasta: contains the exon sequences joined into genes in the aforementioned file.

- Additional generated files are stored for pipeline debugging and controls.

Notes on BITACORA output:

The proteins identified with BITACORA can be used either for further prospective analyses or to facilitate a more curated annotation using genome annotation editors (such as Apollo). In the first case, it is recommended to perform some validation of the newly identified proteins, especially when using the close proximity algorithm. It is important to determine if gene models need to be split or joined (i.e., if the parameter MAXINTRON needs to be modified). For highly divergent gene families, we also recommend using the parameter `BLASTPHMMER=T` in order to identify distant homologs. In addition, it would be also useful to build an MSA, constructing a tree of the gene family including members of other closely related species, or checking for characteristic structural features of the protein (i.e. the presence of transmembrane domains, signal peptides, etc.). See Vizueta *et al.* (2018) for some examples.

In the second case, BITACORA is also designed to facilitate the annotation of the generated gene models in editors as Apollo (Figure 2). The specific files that can be used for this purpose are (see an example in Documentation/example_Apollo.png):

- Original GFF3
- Final GFF3 with curated models for the annotated proteins
- BED files from TBLASTN search

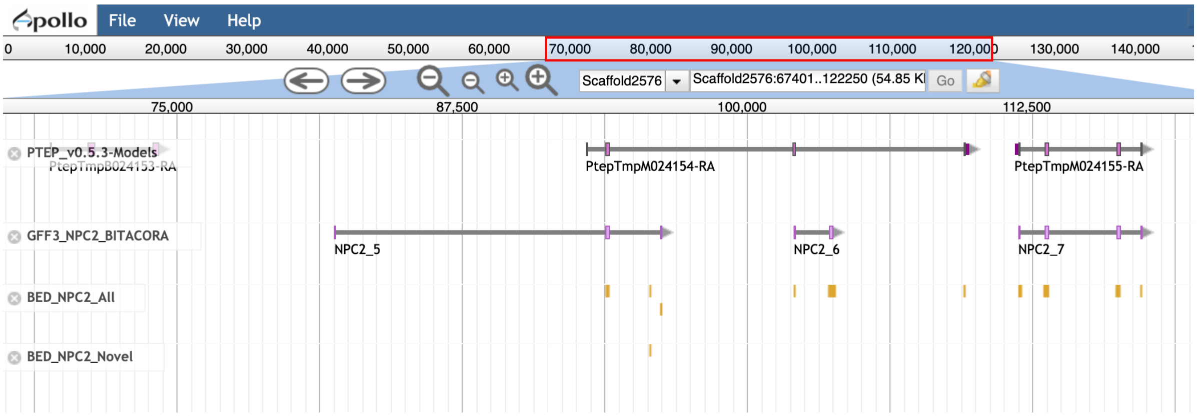


Figure 2. Example of the visualization in Apollo genome editor of the BITACORA output. The example includes the annotation features of three genes encoding NPC2 proteins that are arranged in tandem in the genome of the spider *P. tepidariorum*. The existing automatic annotation of this region, obtained with MAKER2 (track PTEP_v0.5.3-Models), produced a chimeric gene model (PtepTmpM024154-RA; a fusion of two genes), which can be effectively curated with BITACORA (NPC2_5 and NPC2_6 gene models). "GFF3_NPC2_BITACORA" track shows the final BITACORA output with information about the gene models identified and curated by this program; "BED_NPC2_All" and "BED_NPC2_Novel" tracks show information about all putative coding sequences of the focal gene family (here the NPC2 gene family) identified by BITACORA. Note that a novel coding sequence (not predicted in automatic annotations) is predicted by the program.
For clarity, the name of BITACORA tracks have been renamed in the figure: GFF3_NPC2_BITACORA, BED_NPC2_All and BED_NPC2_Novel, correspond to `NPC2_genomic_and_annotated_genes_trimmed_nr.gff3`, `NPC2tblastn_parsed_list_genomic_positions_nogff_filtered.bed` and `NPC2tblastn_parsed_list_genomic_positions.bed`, respectively.

8 Example

An example to run BITACORA can be found in `Example` folder. First, unzip the `Example_files.zip` file to obtain the necessary files for BITACORA. In this example, two chemosensory-related gene families in insects: Odorant receptors (ORs), and the CD36-SNMP gene family; will be searched in the chromosome 2R of *Drosophila melanogaster*. The GFF3 and protein files are modified from original annotations, deleting some gene models, to allow that BITACORA can identify novel not-annotated genes.

To run the example, edit the master script `runBITACORA.sh` to add the path to BLAST and HMMER binaries and run the script. It will take around 1 minute with 2 threads.

```
$ bash runBITACORA.sh
```

9 Citation

Joel Vizueta, Alejandro Sánchez-Gracia, and Julio Rozas. 2019. BITACORA: A comprehensive tool for the identification and annotation of gene families in genome assemblies. *bioRxiv*. https://doi.org/10.1101/593889

Joel Vizueta, Julio Rozas, Alejandro Sánchez-Gracia; Comparative Genomics Reveals Thousands of Novel Chemosensory Genes and Massive Changes in Chemoreceptor Repertories across Chelicerates, *Genome Biology and Evolution*, Volume 10, Issue 5, 1 May 2018, Pages 1221–1236, https://doi.org/10.1093/gbe/evy081

Moreover, if you use GeMoMa, please cite:

J. Keilwagen, M. Wenk, J. L. Erickson, M. H. Schattat, J. Grau, and F. Hartung. Using intron position conservation for homology-based gene prediction. *Nucleic Acids Research*, 2016. doi: 10.1093/nar/gkw092

J. Keilwagen, F. Hartung, M. Paulini, S. O. Twardziok, and J. Grau Combining RNA-seq data and homology-based gene prediction for plants, animals and fungi. *BMC Bioinformatics*, 2018. doi: 10.1186/s12859-018-2203-5

10 Troubleshooting

When BITACORA detects any error related to input data, it stops and prints the description of the error. Please check the error and your data.

If you are getting errors related to parsing the GFF file, take into account that BITACORA expects proteins ID to be as ID in mRNA rows from GFF3.

In case of protein ID and mRNA ID causing error as they are not the named equally, first, you can use the script located in `Scripts/Tools/get_proteins_notfound_ingff.pl` to check which proteins are not found in the GFF3 file, as detailed in the Error message. You could use only those proteins found in the GFF3 in BITACORA.

If all proteins are named differently in the GFF3, you can obtain a protein file from the GFF3 using the script `Scripts/gff2fasta_v3.pl` and use that protein file as input to BITACORA.

You could also modify the perl module `Readgff.pm` to allow BITACORA to read your data. Otherwise, modify the GFF, preferably, as GFF3 format.

If you cannot solve the error, create an issue in Github specifying the error and all details as possible.