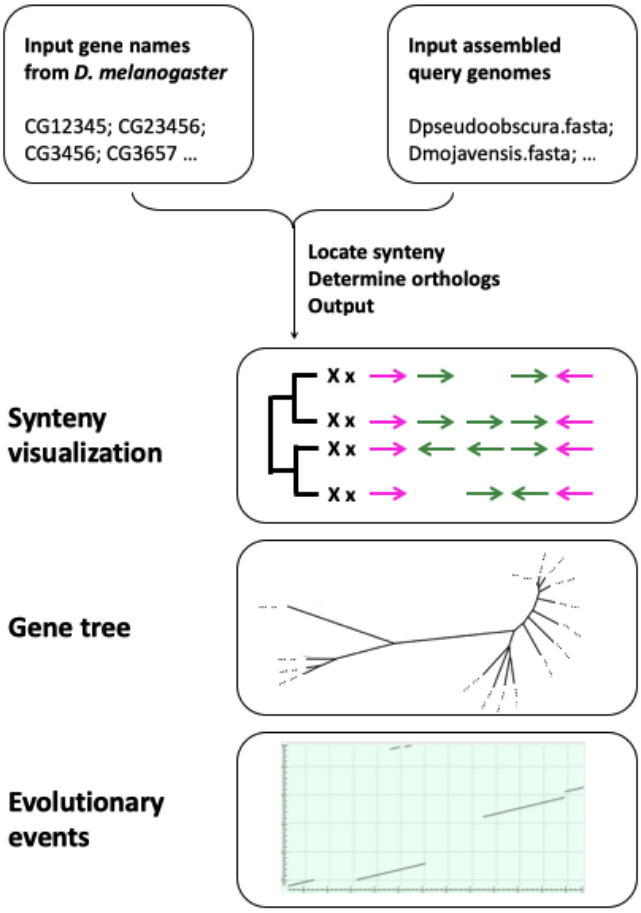
Zinan Wang
CSME890 Final Report

FlyOrtho: a BLAST-based synteny search tool for rapidly evolved orthologs among fruit flies

Abstract

Genes are rapidly evolving with complicated events occurring, e.g., the gain and loss resulted from gene duplication. To functionally study how genes evolve across species, both sequence and synteny homology are important to accurately locate orthologs. However, currently available tools for synteny-based search/comparison cannot satisfy the purpose of single gene search, possibly due to either an over-large scale analyses or low quality genomic annotation. In this class project, I propose to develop a model-species-centric pipeline which integrates local BLAST tools and synteny search for protein coding sequences in assembled genomes regardless of annotation quality. I expect that this new method could focus on single gene or gene family; and compare and analyze the evolutionary relationship among different orthologous genes across species.

Graphic Abstract



1. Introduction

Genes are evolving rapidly across species (Kondrashov 2012, Albalat and Cañestro 2016). This evolution can be resulted from various evolutionary events such as gene duplication and gene loss. In the example of gene duplication, the paralogs could also accumulate mutations and therefore, lead to a rapid and complicated evolution. One consequence from these complicated evolutionary events could result in independent evolution of genes with similar sequence or function. Although BLAST based on the homology of genes can help resolve the evolutionary history at certain degree, lack of information about the synteny could generate false prediction, especially in the cases of gene duplication, gene loss and other more complicated scenarios (Kuzniar et al. 2008). Synteny denotes conserved collocation of genomic elements. To correctly identify the orthologs among species, the first step is to correctly identify the synteny of targeted gene. Genes with high similarity but not in a conserved synteny cannot be determined as the orthologs. With the confirmation of synteny, future analyses can be efficiently conducted to determine the evolutionary events, e.g., gene duplication and loss.

The intuitive idea of determining orthologous synteny regions (or synteny blocks) is to confirm aligned anchoring genes surrounding the gene of interest. Different algorithms have been developed to accomplish this task by either clustering neighboring matching gene pairs, e.g., i-ADHoRe 3.0 (Proost et al. 2011) and SynFind (Tang et al. 2015); or applying dynamic programming to chains of pair-wise collinear genes, e.g., *MCScanX* (Wang et al. 2012) and Genomicus 2018 (Nguyen et al. 2017). Specific scoring systems can then be applied to determine conserved synteny blocks across species. Based on these algorithms/pipelines, some user-friendly software or websites with visualization tools have been developed (with/out modification) for more smoothing experience in identifying syntenies at varied scales from gene families to whole genomes for various types of organisms. For example, MultiSyn develops a web application that determines and visualizes multiple synteny which use algorithms derived from *MCScanX* (Baek et al. 2016).

These methods can help locate large conserved synteny regions across different taxa, however, few constraints limit the uses from molecular biologists with little computational

skills. The poor annotation quality in these large sequences is the major constraint. Though most functional studies usually start from important gene families in model species, molecular biologists have started extending their work to non-model insects with limited information and investigating tools, for example, different species in *Drosophila*. For these less known species, a genomic sequencing is usually one of first steps in investigation. With nowadays cheaper sequencing techniques and more accessible assembly and annotation tools (van Dijk et al. 2018), genomes of any species can be easily sequenced and posted in different database but with various quality. The sequence information of these genomes is valuable, especially for molecular biologists, but difficult to accurately and correctly identify genes of interests for future functional studies. In addition, if manually mining data from these genomic database with poor annotation/operational instruction, the task would be time- and labor-consuming. On the other hand, most methods described here determine synteny regions at a much larger scale compared with few single genes. For the simple purpose of locate few orthologous genes, currently available synteny-search algorithms and methods seem to be too complicated and superfluous, which waste computational resources and provide unnecessary information.

To circumvent these constraints, I propose to design a model-species-centric pipeline which integrates BLAST and synteny search based on protein coding sequences in assembled genomes regardless of annotation quality. Comparing with contemporary methods which use full genome sequences for synteny search, I propose to only use the protein coding sequences (CDS). CDSs are relatively more conserved than the regulatory regions, so synteny search based on CDSs can provide more accurate output. The users can input any genome(s) as they want for the search. At last, I propose to add two extra modules for synteny visualization and evolutionary event analyses. In future versions, this pipeline can be integrated in a graphical interface for easy operations. Fatty acid elongase genes (ELO family) are one of rapidly evolved gene families. For the model insects, ELO genes have been annotated well. However, the gain and loss of these genes in other species have not been well documented. In some poorly annotated genomes, the duplication has been observed being missed from automatic annotation program. The ELO gene family is a perfect model study the birth-and-death model of rapidly evolving genes and will be used as the case study to examine the pipeline efficiency.

2. Data & Methods

2.1 Data

This pipeline is centric to the genome of the vinegar fruit fly, *Drosophila melanogaster*, the model insect which has been most often studied since more than a century ago. For most molecular biology studies, demonstrations of gene functions usually start from experiments in *D. melanogaster* and then extend to other non-model species. Protein coding sequences are more conserved compared with regulatory regions (Cai et al. 2009), which are used in this method to determine synteny blocks. Well-annotated sequence data with improved annotation (Hoskins et al. 2015) were downloaded from FlyBase (https://www.FlyBase.org), including the dataset of all annotation (dmel-all-no-analysis-r6.26.gff), all gDNA sequences (dmel-all-gene-r6.26.fasta), and all protein coding sequences (CDS; dmel-all-CDS-r6.26.fasta) annotated in *D. melanogaster* will be used. The whole genome sequences of all other *Drosophila* species were downloaded from FlyBase or NCBI Genome Database Genomes (https://www.ncbi.nlm.nih.gov/genome) for testing the efficiency of the algorithm and case study.

2.2 Computational methods/approach

Using different *Drosophila* species as the targeted species here, this pipeline will be designed being centric to *Drosophila melanogaster* and tentatively includes the following steps. A database of all genes, CDSs and related position information of *D. melanogaster* was first built for extracting important gene information.

When the gene (id from *D. melanogaster* genome) being input, 20 individual genes of *D. melanogaster* upstream and downstream of the input gene will then be located and correlated CDSs will be extracted for tBLASTn search. A series of tBLASTn search will then be conducted on query genomes. A cutoff of E-value (e-6) will be applied to first filter tBLASTn output, noting that after applying the cutoff, the tBLASTn search could still result in multiple output from different scaffolds. All BLAST results will be treated as a dot indexed by the position in the query genomes. The last step of the pipeline will be connecting the most plausible orthologs of all

recorded genes in the targeted genome using networking algorithm, which selecting the shortest route to connect at least one of these dots. The shortest route will be the orthologous synteny region. Targeted genes or gene families will be BLAST from this synteny region and output. Other modules for data visualization and evolutionary analyses can be used here once we identified the conserved syntenies. The details require more comprehensive considerations.

2.3 Evaluations

In FlyBase, a well-developed genome browser has been developed on six different *Drosophila* species except for *D. melanogaster*. The pipeline will use fatty acid elongase gene family (ELO family) as the case study to analyze all elongase genes on these non-model *Drosophila* species. The accuracy of orthologous search will be manually examined on these species. In addition, all ELO genes will be searched for all available 48 *Drosophila* species. The birth-and-death evolution of these genes will then be analyzed.

3. Results

Due to my naïve ability of using Python, this project requires extra time to accomplish. Here I present current progress including database and sequence parser.

A database was first built which contains all coding sequences and correlated position information of *D. melanogaster*. This database includes two feature tables and two .fasta files. The first feature table has seven columns including gene_id (gene identifier), chrom (chromosome number), start (starting position of the gene), end (ending position of the gene), number_of_cds, gene_name (gene name used in FlyBase), and name_alias (other alias for specific genes) (Fig. 1). In the culture of *Drosophila* studies, the scientists who demonstrated the functions of genes are allowed to name these genes. So all alias of genes are included in the first feature table. The second feature table provides information for the CDSs of D. melanogaster including four columns: CDS_name, CDS_loc (location information for the full CDS), length, and gene_name (Fig. 1).

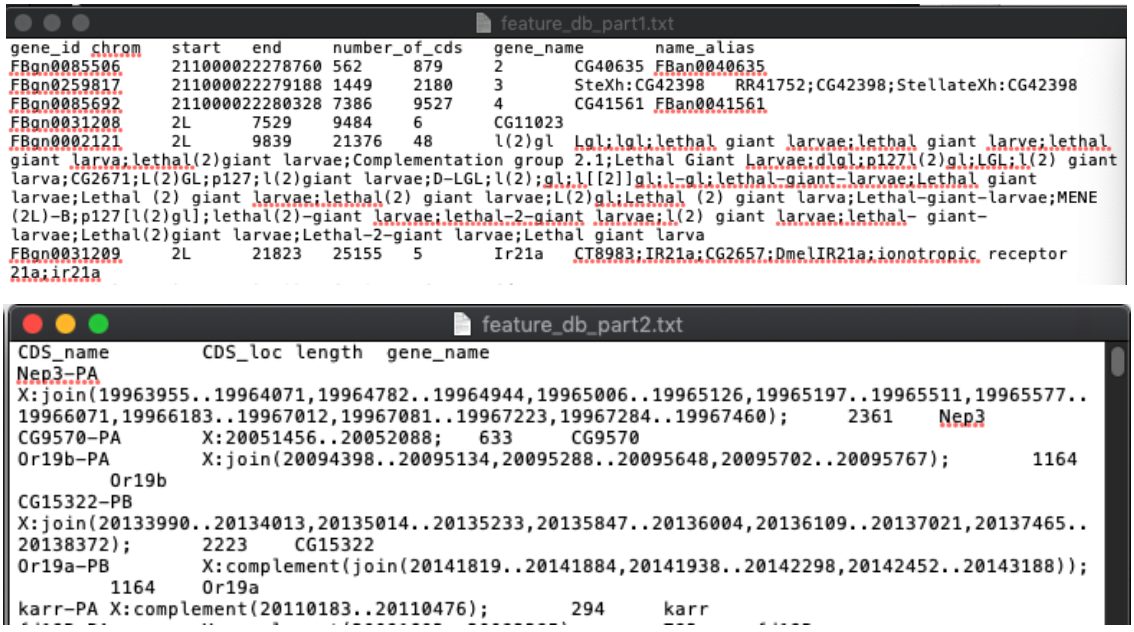


Figure 1. Screenshots of the database for indexing all protein coding sequence from the model insect, *Drosophila melanogaster*. All protein coding sequences are arranged in a sequential order for further analyses, so the nearby genes can be easily located.

As I learned recently about the property of being object-oriented programming in Python, I generated a sequence parser for obtaining information about input gene and flanking genes from the database. A demonstration is in Fig. 2.

```
In [7]: test=Gene(gene_name="CG18609")
        test.get_cds_fasta()

Out[7]: '>CG18609-PA\nATGCTCCGATACTTGCGCATACCTCAAGCGGATCCTAACCCAATTCCGCTGGCTGGATCACCATGGCCTATCACACTGA
        TTTTGATAGCATATCTGTTGTTTGTCCTTAAATTGGGCAAGATCTTTATGAGAAACCGGAAACCATATGATTTGAAAACGGTCTTGAAGGTCT
        ACAATCTATTTCAGGTGCTATACAATGGTCTCTACTTCGGAATGGTTTTTTATTATCTCTTCATCGTGGGCATTTGCAATCTGCACTGCATAG
        AAAGCTTTCCCGAGGGCCATGAACGCAAACAATTGGAACGAGTATTGCATGCCGCCATATCTGCTGAACAAGGTCCTCGATCTTATGGATACGG
        TGTTCTTTGTGCTGCGAAAGAGCTATAAGCAGATCACCTTCCTGCACATATATCACCACGTGTTCATGTCCTTTGGAAGTTATGCCCTAACCC
        GTTACTATGGAACTGGAGGCCATGTCAATGCCGTTGGACTGCTGAACTCCTTGGTGCACACGGTCATGTATTTCTACTACTTTCTGTCTTCAG
        AATATCCCGGAGTGAGGGCCAATATCTGGTGGAAGAAGTATATAACATTGACACAGCTCTGCCAGTTCTTCATGCTGCTCAGTTATGCCATCT
        ATGTGCGATTCTTTTCACCGAATTGTGGCGTTCCACGCGGTCTTCTCTATCTAAATATGGTGCAAGGCGTTGTGTTCATTTATCTGTTTGGTA
        AATTCTATATCGACAACTATCTGAGACCTCCGAAGGCGAAAATCAACGCAAAGCAATCGTAG'
```

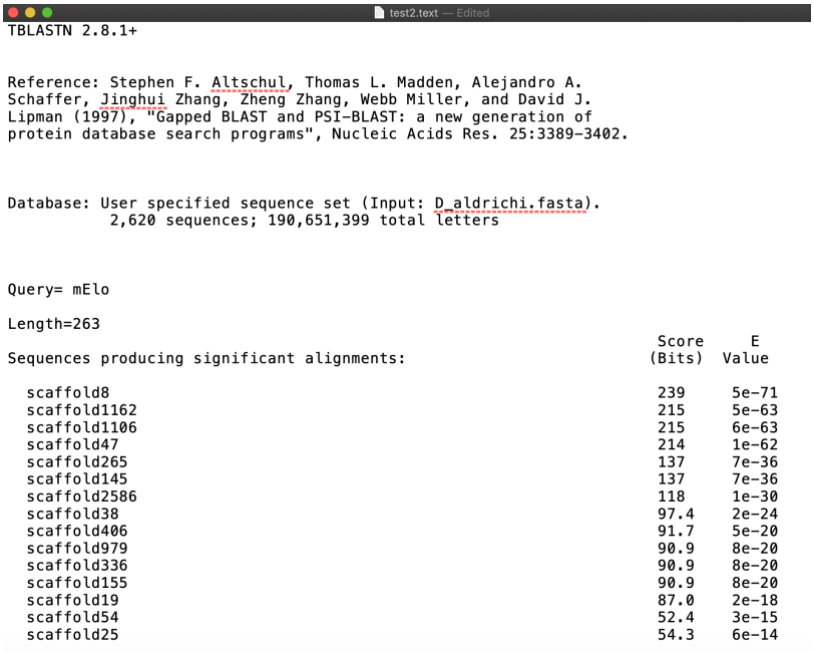
```
In [11]: # LEFT FLANK
         test.get_neighbor_records()[0]
```

Out[11]:

	gene_id	chrom	start	end	number_of_cds	gene_name	name_alias
12959	FBgn0267043	X	19914482	19915281	2	CG45487	NaN
12960	FBgn0267044	X	19919641	19920435	2	CG45488	NaN
12961	FBgn0267045	X	19928416	19929178	2	CG45489	NaN
12962	FBgn0267046	X	19933541	19934294	2	CG45490	NaN
12963	FBgn0267047	X	19938232	19938994	2	CG45491	NaN
12964	FBgn0267048	X	19943357	19944147	2	CG45492	NaN
12965	FBgn0267049	X	19948079	19948873	2	CG45493	NaN
12966	FBgn0267050	X	19952814	19953608	2	CG45494	NaN
12967	FBgn0264309	X	19957548	19958342	2	CG43784	CR43784
12968	FBgn0031080	X	19958393	19959403	2	CG12655	NaN

Figure 2. Demonstration for sequence parser to obtain sequence and position information for targeted genes from the database built in this project.

tBLASTn of translated CDS in genomes of other insect species showed multiple hits at different scaffolds (Fig. 3). My next steps will focus on transform these hits into nodes and then use Nextworkx to locate the minimum path for the synteny search.



test2.text — Edited

```
TBLASTN 2.8.1+


Reference: Stephen F. Altschul, Thomas L. Madden, Alejandro A.
Schaffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J.
Lipman (1997), "Gapped BLAST and PSI-BLAST: a new generation of
protein database search programs", Nucleic Acids Res. 25:3389-3402.


Database: User specified sequence set (Input: D_aldrichi.fasta).
           2,620 sequences; 190,651,399 total letters


Query= mElo

Length=263
                                                                      Score     E
Sequences producing significant alignments:                          (Bits)  Value

  scaffold8                                                           239     5e-71
  scaffold1162                                                        215     5e-63
  scaffold1106                                                        215     6e-63
  scaffold47                                                          214     1e-62
  scaffold265                                                         137     7e-36
  scaffold145                                                         137     7e-36
  scaffold2586                                                        118     1e-30
  scaffold38                                                          97.4    2e-24
  scaffold406                                                         91.7    5e-20
  scaffold979                                                         90.9    8e-20
  scaffold336                                                         90.9    8e-20
  scaffold155                                                         90.9    8e-20
  scaffold19                                                          87.0    2e-18
  scaffold54                                                          52.4    3e-15
  scaffold25                                                          54.3    6e-14
```

Figure 3. A demonstration of tBLASTn of the gene 'mElo' in the genome of *Drosophila aldrichi* showed multiple hits in different scaffolds.

4. Discussion

This project requires more time to accomplish. For now, I have built a database including needed information from *D. melanogaster* genome for the synteny search. I also made a sequence parser so I can obtain sequence and position information from the database. Further, tBLASTn trials showed successful search using the CDSs of *D. melanogaster* in the genomes of other species. Next steps will be using these output to build a network and then locate the conserved synteny for the genes.

Learning a computer language is a struggling process. As a molecular biologist, I have spent most of my time in the laboratory and learn different strategies making transgenic insects. All the concepts, ideas and strategies in computational biology is relatively new to me. In this project, I learned a lot. It helped me make the first step in using a computational language to solve problems. By using Python, I learned different logistics in 'translating' human language to computer language (though Python has a relatively simpler syntax and grammar comparing with other computer languages such as C or Java). I also learned the property of

object-oriented programming when making the class for gene parser. This is a very handful managing strategy.

Another thing I have learned from this project (and the whole course) is the way how computational biologist frame and solve questions. During the whole semester, I learned different algorithms and methods to solve questions, especially some original ideas developed from scratch decades ago. Initially, I thought most computational methods only provide correlation and cannot be practical. However, I realized these methods can be powerful for prediction or assisting molecular experiments.

In my case, I did try different other ways before starting this project (as I present in the class). Those naïve methods did waste me plenty of time. For this project, if I were to start it again, another option for me is to modify current tools and make them suitable to my needs.

References:

Albalat, R., and C. Cañestro. 2016. Evolution by gene loss. Nature Reviews Genetics 17: 379-391.

Baek, J.-H., J. Kim, C.-K. Kim, S.-H. Sohn, D. Choi, M. B. Ratnaparkhe, D.-W. Kim, and T.-H. Lee. 2016. MultiSyn: A Webtool for Multiple Synteny Detection and Visualization of User's Sequence of Interest Compared to Public Plant Species. Evolutionary Bioinformatics 12: EBO. S40009.

Cai, X., H. Hu, and X. Li. 2009. A new measurement of sequence conservation. BMC genomics 10: 623.

Hoskins, R. A., J. W. Carlson, K. H. Wan, S. Park, I. Mendez, S. E. Galle, B. W. Booth, B. D. Pfeiffer, R. A. George, and R. Svirskas. 2015. The Release 6 reference sequence of the Drosophila melanogaster genome. Genome research 25: 445-458.

Kondrashov, F. A. 2012. Gene duplication as a mechanism of genomic adaptation to a changing environment. Proceedings of the Royal Society B: Biological Sciences 279: 5048-5057.

Kuzniar, A., R. C. van Ham, S. Pongor, and J. A. Leunissen. 2008. The quest for orthologs: finding the corresponding gene across genomes. Trends in Genetics 24: 539-551.

Nguyen, N. T. T., P. Vincens, H. Roest Crollius, and A. Louis. 2017. Genomicus 2018: karyotype evolutionary trees and on-the-fly synteny computing. Nucleic acids research 46: D816-D822.

Proost, S., J. Fostier, D. De Witte, B. Dhoedt, P. Demeester, Y. Van de Peer, and K. Vandepoele. 2011. i-ADHoRe 3.0—fast and sensitive detection of genomic homology in extremely large data sets. Nucleic acids research 40: e11-e11.

Tang, H., M. D. Bomhoff, E. Briones, L. Zhang, J. C. Schnable, and E. Lyons. 2015. SynFind: Compiling Syntenic Regions across Any Set of Genomes on Demand. 7: 3286-3298.

van Dijk, E. L., Y. Jaszczyszyn, D. Naquin, and C. Thermes. 2018. The third revolution in sequencing technology. Trends in Genetics 1481: 1-16.

Wang, Y., H. Tang, J. D. DeBarry, X. Tan, J. Li, X. Wang, T.-h. Lee, H. Jin, B. Marler, and H. Guo. 2012. MCScanX: a toolkit for detection and evolutionary analysis of gene synteny and collinearity. Nucleic acids research 40: e49-e49.