PaleoProPhyler, Supplementary Material

Ioannis Patramanis[1], Jazmin Ramos Madrigal[2], Enrico Cappellini[3], and Fernando Racimo[1,4]

[1]Section for Molecular Ecology and Evolution, Globe Institute, University of Copenhagen
[2]Center for Evolutionary Hologenomics, Globe Institute, University of Copenhagen
[3]Section for GeoGenetics, Globe Institute, University of Copenhagen
[4]Lundbeck GeoGenetics Centre, Globe Institute, University of Copenhagen

December 6, 2022

Detailed Overview of Pipelines

In the next few pages you will find a more in depth, step by step, overview of what is running in each module of the pipelines.

Module 1

Description of Module:

Module 1 is designed to provide the user with a baseline (curated) reference dataset as well as the resources required to perform the *in silico* translation of proteins from mapped whole genomes. The input of this module is a user-provided list of proteins and a list of organisms. Both input lists should be in a simple *.TXT file. The user also has the option of choosing a particular reference build. Utilizing the Ensembl API [1], the module will return 3 different resources for each requested protein and for each requested organism / reference build. The 3 output resources are:

1. The reference protein sequence in FASTA format [2].
2. The location (position and strand) of the gene that corresponds to the protein.
3. The start and end of each exon and intron of that gene/isoform.

The downloaded FASTA sequences are available individually but can also be assembled into species- and protein-specific datasets. They can be immediately used as a reference dataset for either downstream phylogenetic analyses or as an input database for mass spectrometry software, like MaxQuant [3], Pfind [4], PEAKS [5] and others [6, 7, 8, 9]. The gene location information and the exon/intron tables can be utilized automatically by Module 2. For the requested proteins, the module will select the Ensembl canonical isoform by default. Should the user desire a specific isoform or all protein coding isoforms of a protein, they have the ability to specify that as an option in the protein list *.TXT file.

Step by step scripts that run:

1. Python3 Script to fetch Ensembl Gene ID for each Protein - Organism Combination (Link to Script)
2. Python3 Script to fetch Transcript ID (Ensembl-Canonical and Optional Alternative Isoforms) for each Gene ID (Link to Script)
3. Python3 Script to fetch Protein Fasta Sequence for each Transcript ID (Link to Script)
4. Python3 Script to fetch Protein Exon and Intron information for each Transcript ID (and create a table for them) (Link to Script)
5. Python3 Script to fetch Gene location (corrected for Reference version) for each Transcript ID (Link to Script)
6. Python3 Script to combine all FASTA Sequences into datasets (per Protein and per Organism) (Part of main Snakemake Script)

DETAILED OVERVIEW OF PIPELINES

Input of Module 1:

- a TXT file with a list of coded protein names, one per line e.g.:

  AMELX
  ENAM
  AMELY

- Alternatively this list can also contain the names of specific isoforms, or the word ALL to bring all known isoforms e.g.:

  AMELX::AMELX-201
  ENAM::ENAM-202
  AMELY::ALL

- a TXT file with a list of scientific organism names e.g.:

  homo sapiens
  pan troglodytes
  gorilla gorilla

- Alternatively this second list can also contain the names of specific reference versions e.g.:

homo sapiens	GRCh37
pan troglodytes	CHIMP2.1.4
gorilla gorilla	gorGor3.1

Module 2

Description of Module:

Module 2 is designed to utilize the resources generated by Module 1 and to extract, splice and translate genes from whole genome data, into the proteins of interest. Module 2 can handle some of the most commonly used genomic data file formats, including the BAM [10], CRAM [11] and VCF [12] formats. The easiest way to run Module 2 is to first run Module 1 for a set of proteins and a selected organism. This will generate all the necessary files and resources required for protein translation. The selected organism will be used as a reference for the translation process. All genomic data to be translated must be mapped onto the same reference organism. The user can then run Module 2 simply by providing the organism's name (and reference version), as well as a list of the samples to be translated, both in a *.TXT file. Should the user want to translate samples from a VCF file, they will also need to provide a reference genome in FASTA format, to complement the variation information of the VCF file. After executing the module, an initial 'normalization' step is performed, where all input files are formatted and indexed. Once this is complete, the locations of the genes are used to extract their sequence and the exon/intron information is used to splice them. These isolated and spliced genomic sequences are then BLASTed [13] onto the reference protein sequence, and the matching translated amino acids are stitched together into the final translated protein sequences. In the last step, the translated sequences are organized into 3 alternative databases:

1. The 'Per protein' database: a folder containing one FASTA file for each translated protein. Each protein FASTA file contains the sequences of that protein for all samples
2. The 'Per individual' database: a folder containing one FASTA file for each translated sample / individual. Each sample FASTA file contains all of the translated proteins for that sample.
3. The 'All Protein Reference' database: a single FASTA file containing all translated proteins for all samples.

Any of these FASTA files can be instantly merged with an ancient protein dataset and used in Module 3.

Input of the module:

This module has 3 alternative input file options:

- CRAM input:
  - Transformed into BAM file using the command `"samtools view -b"`
  - Follows BAM input path
- BAM input:
  - Headers are renamed, e.g. from 'Chr1' to just '1', using the commands `'samtools view'` and `'samtools reheader'`
  - Renamed BAM file is index with `'samtools index -b'`
  - BAM file is split into chromosome BAM files using `'samtools view -b'`
  - Chromosome BAM files are re-indexed using `'samtools index -b'`
  - BAM files are transformed to FASTA file using

`"angsd -minQ 30 -minMapQ 30 -doFasta 2 -doCounts 1  -basesPerLine 60"`

- VCF input:
  - A reference genome in FASTA format needs to be provided and placed in the `'Dataset_Construction/Reference/'` folder
  - The provided reference genome is renamed to the same standard as the BAM files ('Chr1' to just '1')
  - The VCF file is renamed as well, using `'bcftools view'` , `'bgzip -c'` and `'tabix -C -p vcf'`
  - VCF file is converted to FASTA using a combination of `'samtools faidx'` and `'bcftools consensus --missing ? -s'`

All different inputs are now in the same format and will follow the same workflow:

1. Custom R script that uses Exon / Intron Locations to splice DNA FASTA (Link to Script)
2. Blast Reference Protein onto spliced DNA FASTA with `'makeblastdb -dbtype nucl'` and

   `'tblastn -seg no -ungapped -comp_based_stats F  -outfmt 5'`
3. Custom Python3 script to extract blasted / translated protein and output it in FASTA format (Link to Script).
4. Shell commands to merge together individual proteins into larger datasets (per Protein / per Individual

Module 3

Description of Module:

Module 3 is designed to perform a phylogenetic analysis, with some modifications specifically designed for palaeoproteomic data. The input of the Module is a FASTA file, containing all of the protein sequences from both the reference dataset and the ancient sample(s) to be analyzed. Accompanying this FASTA file should be a *.TXT file that contains the name of the dataset-FASTA file as well as the names of all of the ancient samples included in that dataset. The dataset will automatically be split into protein specific sub-datasets, each of which will be aligned and checked for SAPs.

The alignment is a two step process which includes first isolating and aligning the modern/reference dataset and then aligning the ancient samples onto the modern ones using Mafft [14]. Isobaric amino acids that cannot be distinguished from each other by the Mass Spectrometer are corrected to ensure the downstream phylogenetic analysis can proceed without problems. Specifically, any time an Isoleucine (I) or a Leucine (L) is identified in the alignment, all of the modern sequences are checked for that position. If all of them share one of the 2 amino acids, then the ancient samples are also switched to that amino acid. If both I and L appear on some present-day samples, both present-day and ancient samples are switched to an L. The user also has the option to provide an additional *.TXT file named ‘MASKED’. Using this optional file, the user can ‘mask’ a present-day sample such that it has the same missing sub-sequences as an ancient sample.

Finally a small report is generated for each ancient sample in the dataset, and a maximum likelihood phylogenetic tree is generated for each protein sub-dataset through PhyML [15]. All protein sub-dataset alignments are then also merged together into a concatenated dataset. The concatenated dataset is used to generate a maximum-likelihood species tree [16] through PhyML and a Bayesian species tree [17, 18] through MrBayes [19]. The tree generation is parallelized using Mpirun [20].

Input of the module:

The main input of this module is a FASTA file containing both the ancient sequences to be analysed as well as the full reference data set.

- All proteins for all individuals, the format of fasta sequence labels should be: `>SampleName_ProteinName`
- User must also provide the names of the ancient samples in the analysis. Proteins that are not found in any of the ancient samples but exist inside the input fasta file, will not be included in the analysis.
- Optional - Masking: If the user want to mask a modern sample with the missingness of an ancient one they need to provide a file named ’MASKED’ in the module’s main directory. This file should contain two columns and as many rows as necessary. Each row should contain the names of two samples, first the name of a modern sample to be masked and second, separated by a whitespace, the name of an ancient sample. The missing positions of each ancient sample will be masked on top of the same positions of the modern sample, ‘masking’ it as an ancient sample.

Workflow:

1. The initial FASTA is split into protein-specific FASTAs with a custom R script (Link to Script).
2. Each protein-specific dataset is aligned using Mafft:
3. Modern and Ancient samples are first separated.
4. Modern samples are aligned with

```
‘mafft --ep 0 --op 0.5 --lop -0.5 --genafpair --maxiterate 20000 --bl 80 --fmodel‘
```

5. Ancient samples are merged and aligned with modern ones with the ‘mafft-einsi –addlong’ option.

6. Aligned dataset is trimmed of completely missing positions using ‘trimal -noallgaps’

7. A custom R script generates a small table of statistics for each ancient sample in the dataset after correcting for I/L positions (Link to Script).

8. A custom R script concatenates protein-specific FASTAS into Concatenated FASTA. User has the option to filter proteins under certain coverage. A `‘Partition_Helper’` file is also generated, which contains start and stop positions of each protein and can be utilised by NEXUS format phylogenetic software (Link to Script).

9. A custom R script is used to convert FASTA files to PHYLIP format (Link to Script).

10. Multithreaded version of PhyML is run on each separate protein-specific data set using:

    ```
    mpirun -n threads phyml-mpi -i dataset.phy -d aa -b 100 -m JTT -a e -s BEST -v e -o tlr -f
    m --rand_start --n_rand_starts 4 --r_seed (random_seed_number) --print_site_lnl --print_trace
    --no_memory_check
    ```

11. Concatenated data set is converted to NEXUS format using ‘seqmagick convert –output-format nexus –alphabet protein’ , with minor bash command line fixes to ensure proper formatting.

12. Masking: Optional step of ’masking’ a modern sample with the missingness of an ancient sample, takes palce at this step

13. Multithreaded version of MrBayes is run on the concatenated NEXUS file using the MrBayes commands:

    ```
    prset aamodelpr = mixed;
    mcmc nchains = (number of cores)/4 nruns=(number of cores)/8 ngen = 10000000 samplefreq=100
    printfreq=100 diagnfreq=1000;
    sumt relburnin = yes burninfrac = 0.25;
    sump;
    ```

    and adding the contenet of the generated `‘Partition_Helper’` file and finally running the file by executing `"mpirun -np (number of cores/4) mb-mpi\`

Requirements for running the pipeline

The only true requirement for running any of the 3 modules, is having a Linux machine with Conda installed. All of the required software and packages are downloaded and installed through conda using the provided conda environments. Bellow you can find the full list of software and package used by the pipeline.

OS Requirements:

- Linux

List of Software and version used by the pipeline:

- Snakemake v7.3.6 [21]
- Conda v22.9.0 [22]
- Samtools v1.15 [10]
- BCFtools v1.15 [23]
- Blast v2.12.0+ [24]
- Angsd v0.937 [25]
- Mafft v7.490 [14, 26]
- Trimal v1.4.rev15 [27]
- Mpirun v4.1.1 [20]
- PhyML v3.3.20200621 [15]
- MrBayes v3.2.7 [19]
- Seqmagick v0.8.4 https://github.com/fhcrc/seqmagick
- R v4.1.0 [28]
- Python 3 v3.9.6 [29]

R packages

- Bioconductor - ShortRead v1.50.0 [30]
- Phyclust v0.1.30 [31]
- Stringr v1.4.0 [32]

Python packages

- Biopython v1.79 [33]
- OS package [29]
- Sys package [29]
- Requests package v2.26.0 [34]
- RE package v2.2.1[35]

Palaeo proteomic hominid reference dataset

This section of the supplementary file is dedicated to describing how the 'Palaeo proteomic hominid reference dataset' was generated. The dataset is available at: (REF)

Choosing and preparing the list of proteins.

We selected 6 publications cataloging proteins identified in either teeth or bone tissue[36, 37, 38, 39, 40, 41]. From these publications we compiled a list of 1696 unique protein names, which are provided in this file,in the main repository: `\Reference_Protein_List.txt"` We modified this list and used it as an input for Module 1 of the pipeline. From these 1696 proteins, 153 could not be matched to an Ensembl Gene ID. This left a total of 1543 proteins that were successfully translated.

Choosing and preparing samples for translation.

Samples were chosen and used from the following publications:

- (1) 1KG high coverage[42].
- (2) Great ape genomes project[43].
- (3) Morphometric, Behavioral, and Genomic Evidence for a New Orangutan Species[44].
- (4) A high-coverage Neandertal genome from Vindija Cave in Croatia[45].
- (5) A high-coverage Neandertal genome from Chagyrskaya Cave[46].

Choosing individuals for the data set:

For publications 2,3 and 5 all available samples were used. For publication 1, only a maximum of 20 individuals from each population were used, resulting in a final X number of individuals. For publication 4 , the individual named 'Mezmaiskaya' was removed from the data set due to a high amount of predicted unique SAPs. We believe that due to the low coverage of the individual, a high number of variants might be miss-called.

Reference Genomes:

We chose to use the human reference genome as the basis for our translations, due to its higher level of annotation. For this purpose, all individuals were mapped onto either GRCh37[47] or GRCh38[48]. Individuals from datasets 1,4 and 5 were already mapped onto a human reference genome. For datasets 2 and 3, raw fastq files were downloaded and mapped onto the GRCh38 human reference. The mapping workflow is provided in the form of a snakemake python script, along with a conda environment containing all software necessary to run the script.

The re-mapped bam files are available upon request.

Final execution:

Both BAM files and VCF files were then used as input for Module 2, as exemplified by the Tutorial.

references

Bibliography

[1] Andrew Yates et al. “The Ensembl REST API: Ensembl data for any language”. In: *Bioinformatics* 31.1 (2015), pp. 143–145.

[2] David J Lipman and William R Pearson. “Rapid and sensitive protein similarity searches”. In: *Science* 227.4693 (1985), pp. 1435–1441.

[3] Jürgen Cox and Matthias Mann. “MaxQuant enables high peptide identification rates, individualized ppb-range mass accuracies and proteome-wide protein quantification”. In: *Nature biotechnology* 26.12 (2008), pp. 1367–1372.

[4] Hao Chi et al. “Open-pFind enables precise, comprehensive and rapid peptide identification in shotgun proteomics”. In: *BioRxiv* (2018), p. 285395.

[5] Bin Ma et al. “PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrometry”. In: *Rapid communications in mass spectrometry* 17.20 (2003), pp. 2337–2342.

[6] Vadim Demichev et al. “DIA-NN: Neural networks and interference correction enable deep coverage in high-throughput proteomics”. In: *bioRxiv* (2018), p. 282699.

[7] Andy T Kong et al. “MSFragger: ultrafast and comprehensive peptide identification in mass spectrometry–based proteomics”. In: *Nature methods* 14.5 (2017), pp. 513–520.

[8] Stefan K Solntsev et al. “Enhanced global post-translational modification discovery with MetaMorpheus”. In: *Journal of proteome research* 17.5 (2018), pp. 1844–1851.

[9] David N Perkins et al. “Probability-based protein identification by searching sequence databases using mass spectrometry data”. In: *ELECTROPHORESIS: An International Journal* 20.18 (1999), pp. 3551–3567.

[10] Heng Li et al. “The sequence alignment/map format and SAMtools”. In: *Bioinformatics* 25.16 (2009), pp. 2078–2079.

[11] James K Bonfield. “CRAM 3.1: advances in the CRAM file format”. In: *Bioinformatics* 38.6 (2022), pp. 1497–1503.

[12] Petr Danecek et al. “The variant call format and VCFtools”. In: *Bioinformatics* 27.15 (2011), pp. 2156–2158.

[13] Christiam Camacho et al. “BLAST+: architecture and applications”. In: *BMC bioinformatics* 10.1 (2009), pp. 1–9.

[14] Kazutaka Katoh and Daron M Standley. “MAFFT multiple sequence alignment software version 7: improvements in performance and usability”. In: *Molecular biology and evolution* 30.4 (2013), pp. 772–780.

[15] Stéphane Guindon et al. “New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0”. In: *Systematic biology* 59.3 (2010), pp. 307–321.

[16] Joseph Felsenstein. “Evolutionary trees from DNA sequences: a maximum likelihood approach”. In: *Journal of molecular evolution* 17.6 (1981), pp. 368–376.

[17] Bruce Rannala and Ziheng Yang. “Probability distribution of molecular evolutionary trees: a new method of phylogenetic inference”. In: *Journal of molecular evolution* 43.3 (1996), pp. 304–311.

[18] Bob Mau and Michael A Newton. "Phylogenetic inference for binary data on dendograms using Markov chain Monte Carlo". In: *Journal of Computational and Graphical Statistics* 6.1 (1997), pp. 122–131.

[19] John P Huelsenbeck and Fredrik Ronquist. "MRBAYES: Bayesian inference of phylogenetic trees". In: *Bioinformatics* 17.8 (2001), pp. 754–755.

[20] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard Version 4.0.* June 2021. URL: https://www.mpi-forum.org/docs/mpi-4.0/mpi40-report.pdf.

[21] Felix Mölder et al. "Sustainable data analysis with Snakemake". In: *F1000Research* 10 (2021).

[22] *Anaconda Software Distribution.* Version Vers. 2-2.4.0. 2020. URL: https://docs.anaconda.com/.

[23] Heng Li. "A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data". In: *Bioinformatics* 27.21 (2011), pp. 2987–2993.

[24] National Center for Biotechnology Information (US) and Christiam Camacho. *BLAST (r) Command Line Applications User Manual.* National Center for Biotechnology Information (US), 2008.

[25] Thorfinn S. Korneliussen, Anders Albrechtsen, and Rasmus Nielsen. "ANGSD: Analysis of Next Generation Sequencing Data". en. In: *BMC Bioinformatics* 15.1 (Nov. 2014), p. 356. ISSN: 1471-2105. DOI: 10.1186/s12859-014-0356-4. URL: http://www.biomedcentral.com/1471-2105/15/356/abstract (visited on 11/26/2014).

[26] HaiXia Long, ManZhi Li, and HaiYan Fu. "Determination of optimal parameters of MAFFT program based on BAliBASE3. 0 database". In: *SpringerPlus* 5.1 (2016), pp. 1–9.

[27] Salvador Capella-Gutiérrez, José M Silla-Martínez, and Toni Gabaldón. "trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses". In: *Bioinformatics* 25.15 (2009), pp. 1972–1973.

[28] R Core Team et al. "R: A language and environment for statistical computing". In: (2013).

[29] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual.* Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[30] Martin Morgan et al. "ShortRead: a bioconductor package for input, quality assessment and exploration of high-throughput sequence data". In: *Bioinformatics* 25.19 (2009), pp. 2607–2608.

[31] Wei-Chen Chen. "Overlapping codon model, phylogenetic clustering, and alternative partial expectation conditional maximization algorithm". PhD thesis. Iowa State University, 2011.

[32] Hadley Wickham. *stringr: Simple, Consistent Wrappers for Common String Operations.* http://stringr.tidyverse.org, https://github.com/tidyverse/stringr. 2022.

[33] Peter JA Cock et al. "Biopython: freely available Python tools for computational molecular biology and bioinformatics". In: *Bioinformatics* 25.11 (2009), pp. 1422–1423.

[34] Rakesh Vidya Chandra and Bala Subrahmanyam Varanasi. *Python requests essentials.* Packt Publishing Ltd, 2015.

[35] Guido Van Rossum. *The Python Library Reference, release 3.8.2.* Python Software Foundation, 2020.

[36] Gina A Castiblanco et al. "Identification of proteins from human permanent erupted enamel". In: *European journal of oral sciences* 123.6 (2015), pp. 390–395.

[37] Rodrigo DAM Alves et al. "Unraveling the human bone microenvironment beyond the classical extracellular matrix proteins: a human bone protein library". In: *Journal of proteome research* 10.10 (2011), pp. 4725–4733.

[38] Yahya Acil et al. "Detection of mature collagen in human dental enamel". In: *Calcified tissue international* 76.2 (2005), pp. 121–126.

[39] Cristiane R Salmon et al. "Global proteome profiling of dental cementum under experimentally-induced apposition". In: *Journal of proteomics* 141 (2016), pp. 12–23.

[40] Michal Jágr et al. “Comprehensive proteomic analysis of human dentin”. In: *European journal of oral sciences* 120.4 (2012), pp. 259–268.

[41] Eun-Sung Park et al. “Proteomics analysis of human dentin reveals distinct protein expression profiles”. In: *Journal of proteome research* 8.3 (2009), pp. 1338–1346.

[42] M Byrska-Bishop et al. “High Coverage Whole Genome Sequencing of the Expanded 1000 Genomes Project Cohort Including 602 Trios. bioRxiv. 2021”. In: *Publisher Full Text* ().

[43] Javier Prado-Martinez et al. “Great ape genetic diversity and population history”. In: *Nature* 499.7459 (2013), pp. 471–475.

[44] Alexander Nater et al. “Morphometric, behavioral, and genomic evidence for a new orangutan species”. In: *Current Biology* 27.22 (2017), pp. 3487–3498.

[45] Kay Prüfer et al. “A high-coverage Neandertal genome from Vindija Cave in Croatia”. In: *Science* 358.6363 (2017), pp. 655–658.

[46] Fabrizio Mafessoni et al. “A high-coverage Neandertal genome from Chagyrskaya Cave”. In: *Proceedings of the National Academy of Sciences* 117.26 (2020), pp. 15132–15136.

[47] Deanna M Church et al. “Modernizing reference genome assemblies”. In: *PLoS biology* 9.7 (2011), e1001091.

[48] Valerie A Schneider et al. “Evaluation of GRCh38 and de novo haploid genome assemblies demonstrates the enduring quality of the reference assembly”. In: *Genome research* 27.5 (2017), pp. 849–864.