GenePainter 2 - Documentation

Authors

Stefanie Mühlhausen, Björn Hammesfahr, Florian Odronitz, Marcel Hellkamp, Stephan Waack, Martin Kollmar

How to cite

Stefanie Mühlhausen, Marcel Hellkamp & Martin Kollmar (2014)
GenePainter 2.0 resolves the taxonomic distribution of intron positions.
Bioinformatics, pii: btu798 [Epub ahead of print]

and

Björn Hammesfahr†, Florian Odronitz†, Stefanie Mühlhausen, Stephan Waack & Martin Kollmar (2013)
GenePainter: a fast tool for aligning gene structures of eukaryotic protein families, visualizing the alignments and mapping gene structures onto protein structures. *BMC Bioinformatics* **14**, 77.
http://www.biomedcentral.com/1471-2105/14/77 Open Access Highly accessed

Contact

Dr. Martin Kollmar
Group Systems Biology of Motor Proteins
Department of NMR-based Structural Biology
Max-Planck-Institute for Biophysical Chemistry
Am Faßberg 11
37077 Göttingen

mako@nmr.mpibpc.mpg.de

Homepage

http://www.motorprotein.de/genepainter

Licence

Table of contents

Authors 1
How to cite 1
Contact 1
Homepage 1

Licence 1

Table of contents 2

Introduction 3

Installation 3
Unpack 3
Compilation 3
Ruby version 3

Usage 3
Ruby interpreter 4
Using GenePainter under windows 4

Example 4

Options 4
Text-based output format 4
Graphical output format 5
Taxonomy and statistics 5
General options 7

Input 7
Multiple protein sequence alignment 8
Gene structure files 8

Meaning of the parameters 11
Text-based output 11
Graphical output 16
Gene structures mapped to protein structures 17
History of Intron Gain and Loss Events 18
Statistics 21
Data and output selection 21

Introduction

The conservation of intron positions comprises information useful for de novo gene prediction, protein sequence alignment improvement, and for analyzing the origin of introns. GenePainter is a standalone tool for mapping gene structures onto protein multiple sequence alignments (MSA). Gene structures, as obtained for example by using WebScipio (http://www.webscipio.org), are aligned with respect to the exact positions of the introns (down to nucleotide level) and intron phase. The output can be visualized in various formats, ranging from plain text to complex graphical figures.

Installation

Unpack

Use one of the following methods, depending on the archive file type:

```
$ unzip gene_painter.zip
$ tar -xzf gene_painter.tgz
```

Compilation

No compilation required.

Ruby version

Ruby version 2.0 or higher is required. If necessary, consider using Ruby Version Manager (https://rvm.io/; RVM) to install and work with multiple ruby environments on your machine.

Usage

```
$ ruby gene_painter.rb -i <alignment> -p <yaml-files> \
[<options>]
```

Option		Description
--input <file>	**-i**	Multiple protein sequence alignment in FASTA format.
--path <path>	**-p**	Path to the directory containing the gene structures in YAML- or GFF-format.

A more detailed description of the MSA and the YAML/GFF files as well as the incorporation of both by GenePainter is given in the Input section.

Ruby interpreter

Invoke GenePainter via one of the following options:

1. As a script

```
$ ruby gene_painter.rb
```

2. As a program

```
$ ./gene_painter.rb
```

Important note

GenePainter assumes that `/usr/bin/env ruby` points to the RVM ruby interpreter. While this will work on most UNIX systems, it might not work on windows machines. In case you encounter any "`/usr/bin/env ruby: No such file or directory`" errors, please edit the very first line of script gene_painter.rb and change the specified path to the correct one. For example, with the Ruby interpreter located at `/usr/bin/ruby`, the Shebang should look like this: `#!/usr/bin/ruby.`

Using GenePainter under windows

GenePainter was developed for UNIX and Mac. To use GenePainter under Windows, you might consider installing Cygwin, which comes with a bash and appropriate GNU tools. Please make sure to have GNU tools "`grep`" and "`tar`" installed and in your executable path when using GenePainter with its taxonomy options. If not, you should unpack the NCBI taxonomy dump yourself and add the parameter `--no-grep` to read the taxonomy dump into memory instead of grepping its content. In order to use GenePainter with the parameter `--tree`, please make sure to have "`python`" installed and in your executable path.

Example

```
$ ruby gene_painter.rb \
-i example/coronin_alignment.fas -o coronin \
-p example/coronin_genes --svg 500 1000 \
--pdb example/2AQ5.pdb --pdb-ref-prot HsCoro1A \
--intron-phase
```

Options

Text-based output format

In the standard output format, exons are marked by "-" and introns by "|".

Option	Description
--intron-phase	Mark introns by intron phase instead of by "|".
--phylo	For phylogenetic analysis: Mark exons by "0" and introns by "1"
--spaces	Mark exons by blank (" ") instead of "-" and introns by "|".
--no-standard-output	Suppress standard output-format. Default: standard output is always generated
--alignment	Output the alignment file with additional lines containing intron phases.
--consensus <N>	Additional line marking introns occurring in N percent of all genes. Specify N as decimal number between 0 and 1.
--merge	Additional line marking all introns present in any gene.

Graphical output format

Option	Description
--svg	Create an svg-file of all gene structures
--svg-format <FORMAT>	Format is one of "normal", "reduced" and "both".
	Use "normal" to create a detailed svg (default).
	Use "reduced" to create an svg focused on common introns.
	Use "both" to create both svg files.
--pdb <FILE>	Two scripts for execution in PyMol will be provided. The scripts will colour the consensus exons and splice junctions of consensus exons. Combine with "--consensus", "--merge" or "--pdb-ref-prot-struct" to specify, which introns should be considered.
--pdb-chain <CHAIN>	Mark gene structure for chain CHAIN. Default: chain A
--pdb-ref-prot-struct	Colour only the intron positions occurring in the gene specified by "--pdb-ref-prot".
--pdb-ref-prot <PROT>	Use protein PROT as reference for alignment with pdb sequence. Default: first protein in the input alignment
--tree	Generate newick tree file and SVG representation

Taxonomy and statistics

Option	Description
--statistics	Provides a plain text file with information about

	each intron. Specify “--taxonomy" and “--taxonomy-to-fasta” to include taxonomic information.
--taxonomy <FILE>	FILE is database dump of NCBI taxonomy. As alternative, a file containing an extract of NCBI taxonomy can be specified. Format of such a file: Lineage must be semicolon-separated list of taxa from root to species.
--taxonomy-to-fasta <FILE>	Text-file defining the matching between fasta header and NCBI species One or more genes given as semicolon-separated list and species name. Delimiter between gene list and species name must be a colon. The species name itself must be enclosed by double quotes like this "SPECIES"
--taxonomy-common-to <X,Y,Z>	Mark introns common to taxa X,Y,Z in an additional line. List must consist of at least one NCBI taxon.
--[no-]exclusively-in-taxa	Mark introns occurring (not) exclusively in taxa specified by “--taxonomy-common-to”.
--introns-per-taxon	Mark all introns newly gained in each *last common ancestor* of supplied species by intron phase. Provides a plain text file.
--no-grep	Read the NCBI taxomony dump into RAM. This will require some hundert MBs of RAM additionally. Default: taxomony dump is parsed with 'grep' calls.
--nice	Give grep calls to parse taxonomy dump a lower priority. Please make sure to have 'nice' in your executable path when using this option.

Analysis and output of all or subset of data

Option	Description
--analyse-all-output-all	Analyse all data and provide full output [default]
--analyse-all-output-selection	Analyse all data and provide text-based and graphical output for selection only. All introns are analysed, including those not present in selection
--analyse-selection-output-selection	Analyse selected data and provide output for selection only
--analyse-selection-on-all-data-output-selection	Analyse intron positions of selected data in all data and provide output for selection only. Introns present in selection are analysed in all data.

Selection criteria for data and output selection

Option	Description
--selection-based-on-regex <"REGEX">	Specify a regular expression (without the enclosing "/" and any modifiers). It will be applied on gene structure names.
--selection-based-on-list <X,Y,Z>	List of gene names to be used.
--selection-based-on-species <"SPECIES">	Use all genes associated with that species. Specify also `--taxonomy-to-fasta` to map gene structure file names to species names
--select-all	No selection applied (default)

General options

Option		Description
--outfile <FILE>	**-o**	Prefix of the output file(s). Default: "genepainter"
--path-to-output <path>		Path to location for the output file(s). Default: same location as GenePainter source files
--range <START,STOP>		Restrict genes to alignment range START-STOP
--[no-]delete-range		Delete everything within (outside) the specified range.
--keep-common-gaps		Keep alignment gaps common to all sequences.
--[no-]separate-introns-in-textbased-output		(Not) Separate each consecutive pair of introns by an exon placeholder in text-based output formats. Default: Separate introns unless the output lines get too long.
--help	**-h**	Show help message.

Input

GenePainter expects two types of input:

1. A FASTA-formatted multiple sequence alignment (MSA);
2. A folder containing gene structures in YAML format as specified by WebScipio or in GFF v.3 format.

GenePainter combines information from the alignment with the gene structures. Therefore, the protein names from the MSA must match to the YAML or GFF filenames. Only those genes, which can be matched (i.e. protein name equals the YAML or GFF filename), will be analysed.

Multiple protein sequence alignment

This file must be a multiple protein sequence alignment, in which all sequences are of same length. Protein sequences are matched with the gene structures on the basis of the FASTA header and file names, respectively. To this end, the FASTA header must be exactly like the corresponding YAML or GFF filename for each gene, which should be included in the analysis. For this reason, FASTA headers must not contain any blanks or special characters.

Gene structure files

For the analysis, GenePainter needs gene structure information for each gene. This information must be stored either in the YAML-format as generated by WebScipio (http://www.webscipio.org) or in GFF v.3 format. Moreover, all gene structures must be located in the same directory.

GFF

GFF3 formatted gene structures are parsed for "CDS" features. All "CDS" features are considered, which are linked via the attribute column (*parent tag*) to the first "mRNA" feature specified. If no mRNA feature is given in the GFF file, all CDS features are parsed. GenePainter uses column 1 (*seqid*), columns 4 and 5 (*start* and *end of feature*) and column 8 (*phase*). All other features and columns are ignored. GFF files generated by WebScipio can also be used as input, although they do not strictly follow GFF3 conventions.

YAML

The most convenient way to obtain YAML-formatted files is to use the WebScipio web interface for gene reconstruction and to download the resulting YAML files. For automation of the YAML generation, several scriptable alternatives exist. First, WebScipio can be accessed by its web service API. This can be done within any software program. In the GenePainter package, scripts are included for querying WebScipio with genes belonging to a single species (`generate_yaml_for_species.rb`) or with genes belong to different species (`generate_yaml_for_multiple_species.rb`). Both scripts access WebScipio through the web service and store the resulting YAML files locally. A brief introduction to the usage of the web service can be found at the WebScipio homepage (http://www.webscipio.org/webscipio/web_service). Second, Scipio can be used locally, which requires further software (BLAT, Bioperl, YAML Perl module) and respective genome assembly files.

A list of all species available can be found at
`http://www.webscipio.org/webscipio/genome_list`

Usage of `generate_yaml_for_species.rb`

```
$ ruby tools/ generate_yaml_for_species.rb \
-s 'Species name' -i fasta_sequence.fas
```

Mandatory Arguments

Option		Description
--species <'species_name'>	-s	Species encoding the specified protein(s). Species should be wrapped with " ' " to preserve spaces.
--input <file>	-i	Path to fasta-formatted protein sequence(s). Might be a multiple sequence alignment of sequences encoded by same species.

Options

Option		Description
--outfile <file_name>	-o	Name of the YAML output file. ONLY used if the fasta file contains only one sequence. Default: Use fasta-header of the input protein sequence.
--help	-h	Show help message.

Usage of generate_yaml_for_multiple_species.rb

```
$ ruby tools/ generate_yaml_for_multiple_species.rb \
-s example/fastaheaders2species.txt -i fasta_sequence.fas
```

Mandatory Arguments

Option		Description
--species-to-fasta <file>	-s	Text-based file mapping fasta header to species names. Mandatory line-format: Fastaheader1[,Fastaheader2]:Species
--input <file>	-i	Path to fasta-formatted multiple sequence alignment.

Options

Option		Description
--help	-h	Show help message.

YAML files will be named like the corresponding fasta-headers. YAMLs are only generated for those sequences, for which a species is specified.

Structure of YAML files

Scipio and WebScipio store gene structure information in YAML format. This format comprises a collection of key – value pairs, an associative array. However, the accurate gene structure representation requires more keys than necessary for the alignment of the gene structures. Thus, GenePainter ignores some data included in the YAML files. Accordingly, these additional keys need not be included in manually reconstructed YAML files. A minimal working example YAML file is defined in Figure 1. An exhaustive description of all keys used by

WebScipio can be found at the WebScipio homepage (http://www.webscipio.org/help/scipio#description).

```
---
- matchings:
    type: exon
    nucl_start: 0
    nucl_end: 198
    dna_start: -72598366
    dna_end: -72598168
    prot_start: 0
    prot_end: 66
    translation: MSRQVVRSSKFRHVFGQPAKADQCYEDVRVSQTTWDSGFCAVNPKFVALICEASGGGAFLVLPLGK
    seq: atgagccggcaggtggtccgctccagcaagttccgccacgtgtttggacagccggccaaggccgaccagtgctatgaagatgtgcgcgtct

    type: intron
    nucl_start: 198
  - dna_start: -72598168
    dna_end: -72596978
    seq: gtgagccccctgaggccctgaggggggcagctcctccaccgagcccatgactctgtgcaggtcctgattaggtgacggtcaccaggccttcc
...
    type: intron
    nucl_start: 1065
  - dna_start: -72594999
    dna_end: -72594999
    seq: tcggacctgttccaggaggacctgtacccacccaccgcagggcccgaccctgccctcacggctgaggagtggctgggggtcgggatgctg

    type: exon
    nucl_start: 1281
    nucl_end: 1383
    dna_start: -72594716
    dna_end: -72594614
    prot_start: 427
    prot_end: 461
    translation: DAVSRLEEEMRKLQATVQELQKRLDRLEETVQAK
    seq: gatgccgtgtctcggctggaggaggagatgcggaagctccaggccacggtgcaggagctccagaagcgcttggacaggctggaggagacag
  ID: 720
  status: auto
```

Figure 1 - Excerpt from the YAML file describing HsCoro1A. All exon and intron descriptions but the very first and last ones have been omitted (marked in yellow). Blank lines were added to separate exon and intron descriptions. Additionally, green boxes highlight exons and blue boxes highlight introns. Only those key – value pairs, which are relevant for GenePainter are shown. The original YAML file is part of the test data included in the package.

The list of exons and introns ("matchings") must start with the keyword "`- matchings:`". The order of keys describing the exons and introns is not important. Mandatory keys for successful incorporation in GenePainter are listed in the following tables:

YAML keys	Description
type	"intron", "intron?", "exon", or "gap". "intron?" is used for uncertain introns (unusual splice patterns found)
nucl_start	Location in the query (in nucleotide coordinates).
seq	DNA sequence of the feature.

YAML keys that appear only in exons	Description
nucl_end	Location in the query (in nucleotide coordinates).

Structure of file mapping fasta header to species names

```
HsCoro1A,HsCoro1B,HsCoro1D,HsCoro1C,HsCoro2B,HsCoro2A,HsCoro3:"Homo sapiens"
GgCoro1C:"Gallus gallus"
GgCoro2A:"Gallus gallus"
...

```

Figure 2 - For taxonomy options as well as to generate YAML files, a mapping between genes described in MSA and corresponding species must be established. To this end, fasta header (separated by "`,`") are mapped to species names. Fasta header and species names are separated by "`:`".

Genes that are linked to the corresponding species are considered for taxonomic computations or generation of YAML files, respectively. All other genes are omitted from these computations. Species names must match NCBI taxonomy.

Meaning of the parameters

The following figures illustrate some of GenePainters output formats and options. All figures were generated with the test data comprising coronin genes as included in the archive `gene_painter.zip`.

Text-based output

Basis intron pattern representations

The basic output format is a plain text-file where exons are represented as minus signs and introns as vertical bars (Figure 3A). Using the `--spaces` option (Figure 3B), shows only introns. A more detailed output including intron phases can be obtained by using the `--intron-phase` option (Figure 3C). Options `--merge` and `--consensus` are applied to all generated output.

In all text-based output formats, consecutive introns are by default (or by setting `--separate-introns-in-textbased-output`) separated by exon placeholders ("`-`"). By setting the option `--no-separate-introns-in-textbased-output`, delimiters between consecutive introns are omitted (Figure 4).

Introns next to alignment gaps

A special case arises when intron positions map between amino acids that are separated by alignment gaps. In this case, the intron could be positioned on both sites of the alignment gap. This becomes relevant when introns are mapped to other sequences in the same region. By default, GenePainter assumes that these regions represent less conserved parts of the proteins as they appear in loop

regions, and aligns the intron located in the alignment gap to the first respective intron of the other sequences (Figure 5A).

However, this alignment might be wrong, if the alignment gap is not caused by natural sequence length variation but by a sequence gap caused by e.g. a gap in the respective genome assembly. In this case, the intron alignment can be turned of by setting `--no-best-position-intron` (Figure 5B).

Enhancing multiple sequence alignments, output for phylogenetic analyses

Moreover, intron phases can be included as additional lines in the given alignment (option `--alignment`). Common gaps are removed by default. To keep them, please specify `--keep-common-gaps` (Figure 6A and B). If two or three distinct introns are at the same amino acid alignment position, the will be displayed as `'?'` instead of listing their intron phase (because of space limitations; figure not shown). Option `--phylo` generates an alignment based on the presence (`1`) and absence (`0`) of introns for further phylogenetic analyses (Figure 6C).

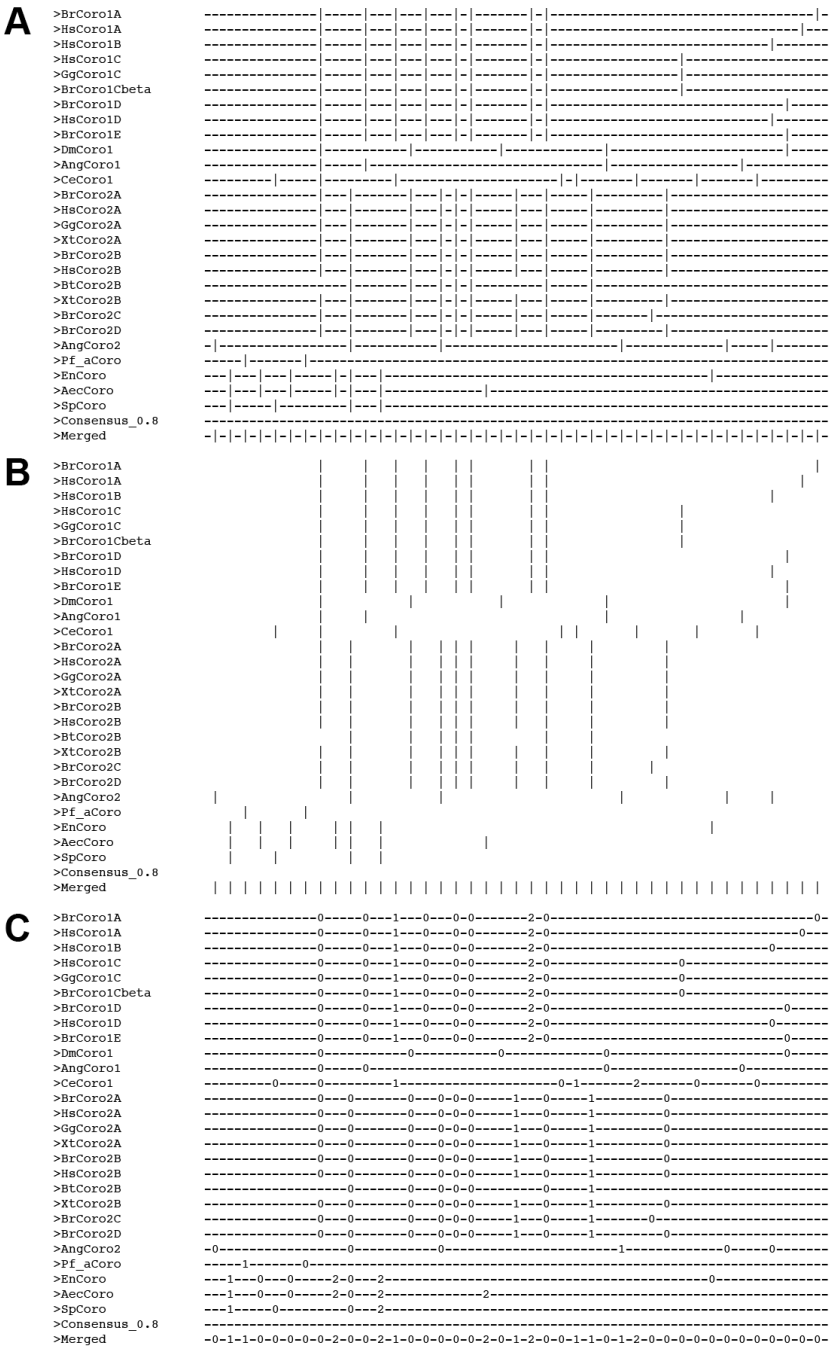

Figure 3 - Basic output formats.

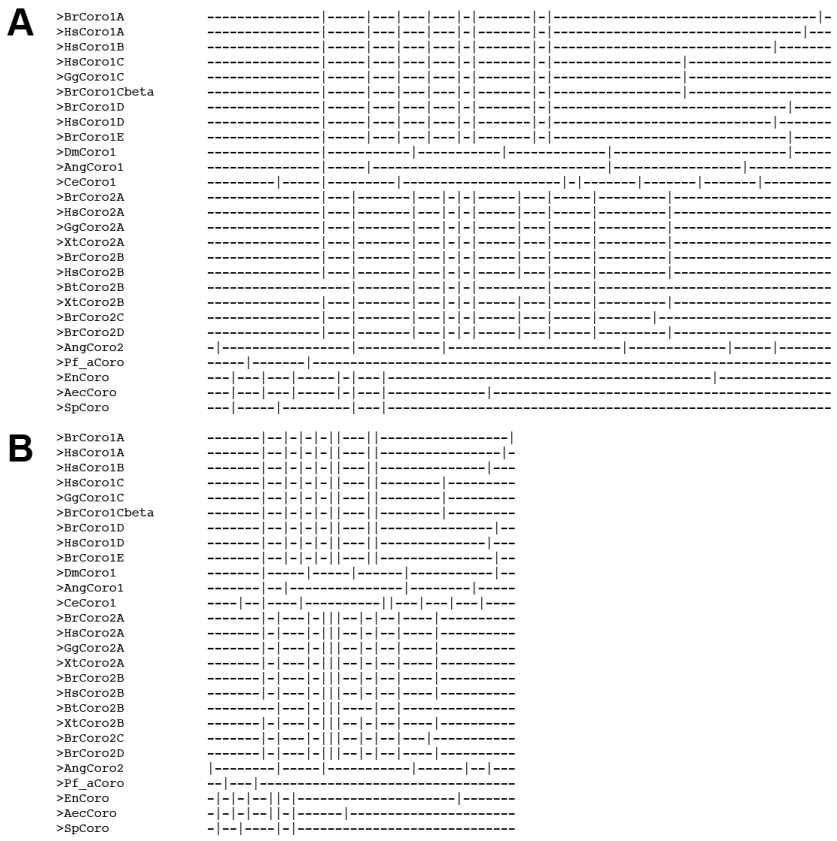

Figure 4 - Effect of `--no-separate-introns-in-textbased-output` option.

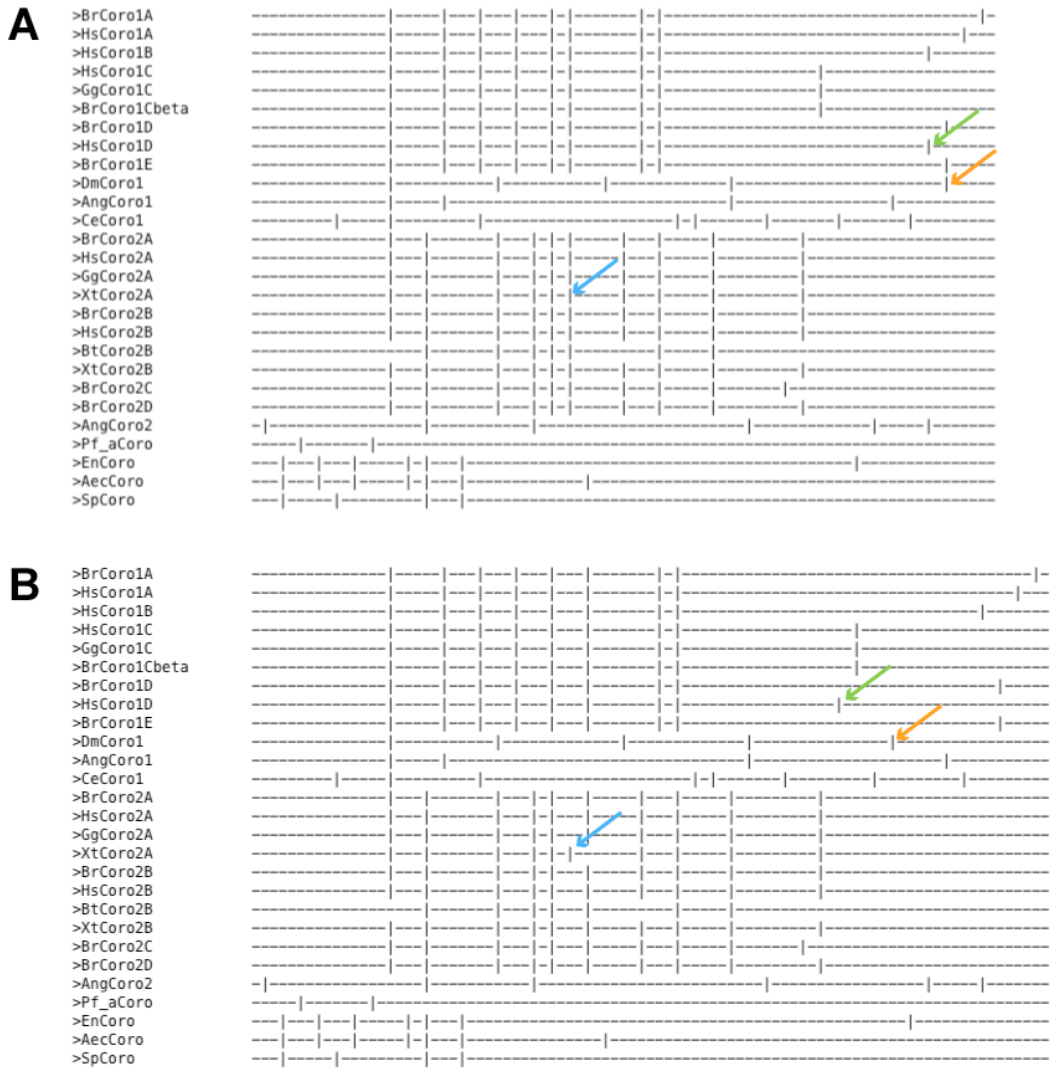

Figure 5 - Effect of `--no-best-position-introns` option.

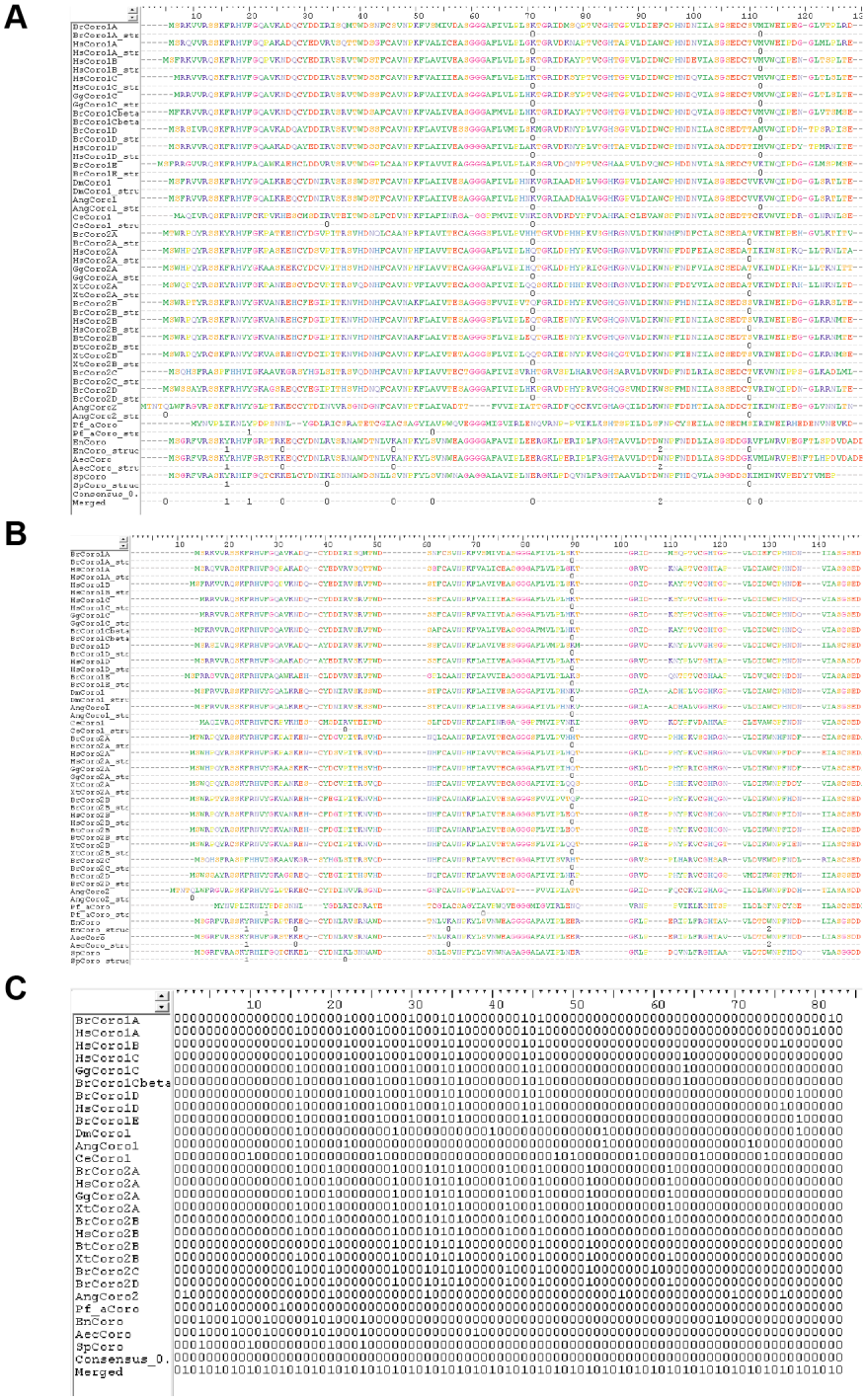

Figure 6 - Fasta-formatted output formats.

Graphical output

Apart from text based output formats, a graphical output can be generated. To this end, SVG parameters need to be set with `--svg`. The SVG can be drawn in two formats, `normal` and `reduced` (set option with `--svg-format`). In each format, common introns are drawn underneath each other. While exons and introns are both drawn to scale with format `normal`, intron length is fixed within format `reduced` (Figure 7). Both formats can be requested at the same time by specifying `both`.

Figure 7 - Graphical output options.

Gene structures mapped to protein structures

Additionally, if a pdb file is specified via `--pdb example/2Aq5.pdb`, intron positions and phases are mapped onto protein structure. Figure 8A demonstrates mapping of the exons of the human coronin HsCoro1A gene (`--pdb-ref-prot HsCoro1A`) onto the protein structure of mouse coronin MmCoro1A gene (the pdb file is part of the test data set). While for this figure all exons present in the reference sequence (`--pdb-ref-prot-struct`) are considered, Figure 8B displays all exons that are conserved in at least 80% of all proteins (default). Accordingly, splice sites are shown in Figure 8C and Figure 8D. In this output, attention is drawn to intron phases. A three-color scheme and numbers denote phases.

Part of the underlying algorithm is the calculation of a global alignment between reference and pdb sequence. The implementation of the Needleman-Wunsch algorithm was adapted from Michael Ryan, Copyright (c) 2011 (part of the ruby gem `align`, downloaded from https://rubygems.org/gems/align at 01-15-2014).

Plotting intron positions onto the protein structure

GenePainter writes two scripts for execution in PyMOL. In order to invoke them, first open the PDB file. Then execute the following commands in *the PyMOL command line*:

```
$ run <path_to/script_name>
$ <script_name>
```

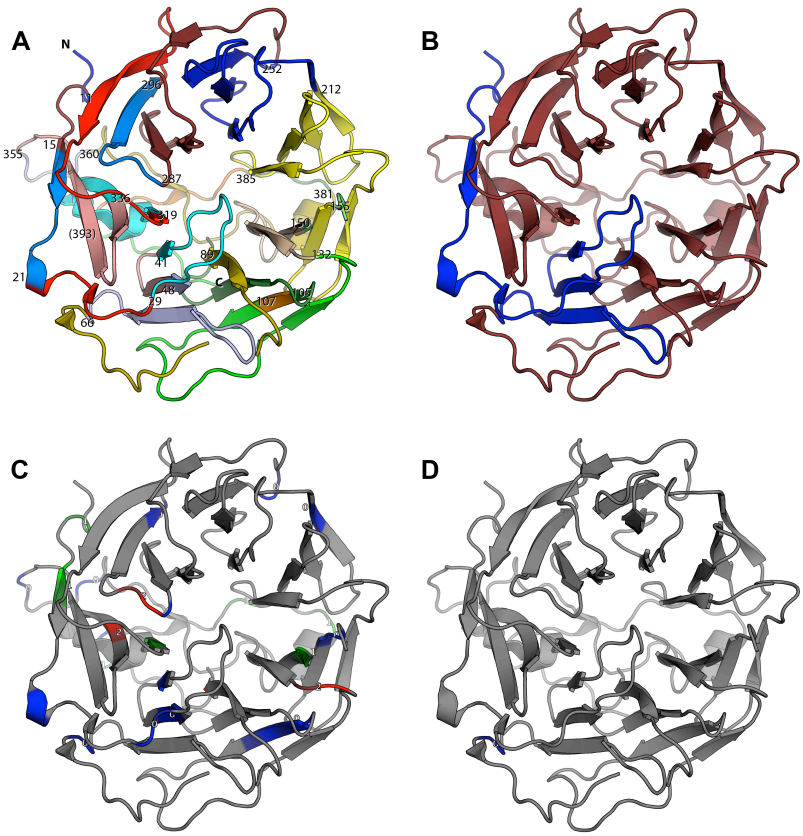

Figure 8 - Mapping of conserved exons (9A, 9B) and intron position and phase (9C, 9D) onto protein structure.

History of Intron Gain and Loss Events

Plotting intron gain and loss patterns onto a phylogentic tree

Specifying parameter `--tree` in combination with taxonomy options `--taxonomy` and `--taxonomy-to-fasta` can generate a phylogenetic tree of the species analysed. Taxonomic relations of the species named in `--taxonomy-to-fasta` as defined by NCBI taxonomy are converted to a phylogenetic tree in newick file format and printed in an additional SVG file. The python script "`phb2svg.py`" does the conversion of newick tree to SVG. To use `--tree` option, the python interpreter needs to be in your executable path.

The generated phylogenetic tree focuses on gain and loss of intron positions. The generated tree file can be viewed using standard tree viewer software. Branch labels contain the branch name and the number of intron positions that were gained (first number) and lost (second number) at this node. Leafs are annotated with the number of intron positions occurring in that taxon (first number) and the number of genes analysed (second number). If introns were gained or lost in a leaf, these numbers are included in the leaf name (separated by an underscore). In addition, an SVG representation of the tree is generated (Figure 9). In the SVG file, branches are annotated with taxon name (written above the branch) and number of intron losses and (coloured in red and green).

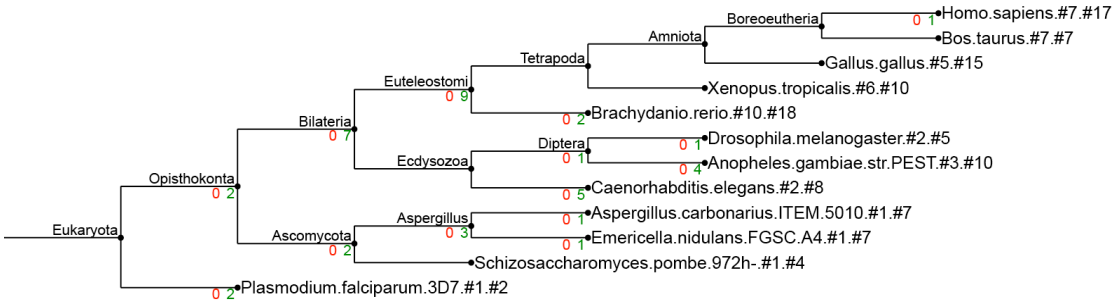

Figure 9 - Phylogenetic tree of those braches, at which introns were lost (coloured red) or gained (coloured in green). The numbers displayed behind the species comprise the total number of genes (first number) and intron positions present in these genes (second number)

Adding common intron patterns to the standard output

To project intron positions onto the species they occur in, the path to NCBI taxonomy database dump as well as the mapping between fasta headers and NCBI species names must be specified with parameters `--taxonomy` and `--taxonomy-to-fasta`.

GenePainter can extract the taxonomic lineages from NCBI taxonomy database dump or from a user-provided text-file containing taxonomic lineages. In this file, one lineage should be given per line. Taxa should be ordered from root to species and be separated from each other by semicolons. Optionally, a blank may follow the semicolon (Figure 10).

The file containing the mapping of fasta headers and species names must be in a specific format: Exactly one pair of fasta header and species name per line, which is separated by ":". The species name must be enclosed by double quotes. Blanks immediately before and after ";" are permitted, they are ignored by the program. More than one fasta header belonging to the same species might be given in one line, the list of fasta headers needs to be comma-separated or semicolon-separated (Figure 2).

```
1  Cellular organisms; Eukaryota; Opisthokonta; [...] Hominidae; Homininae; Homo; Homo sapiens
2  ...
3
```

Figure 10 - Excerpt of file providing a user-defined taxonomic lineage. For displaying purpose, parts of the lineage were omitted.

In addition, one of the taxonomic output formats needs to be specified: Parameter `--taxonomy-common-to <x,y,z>` adds a merged exon-intron pattern to standard output containing all introns occurring in any of the species belonging to the specified taxa (Figure 11A). By default, introns need not be exclusively for the specified taxa. To change this behaviour and see only those introns occurring only in the specified taxa, add parameter `--exclusively-in-taxa` (Figure 11B).

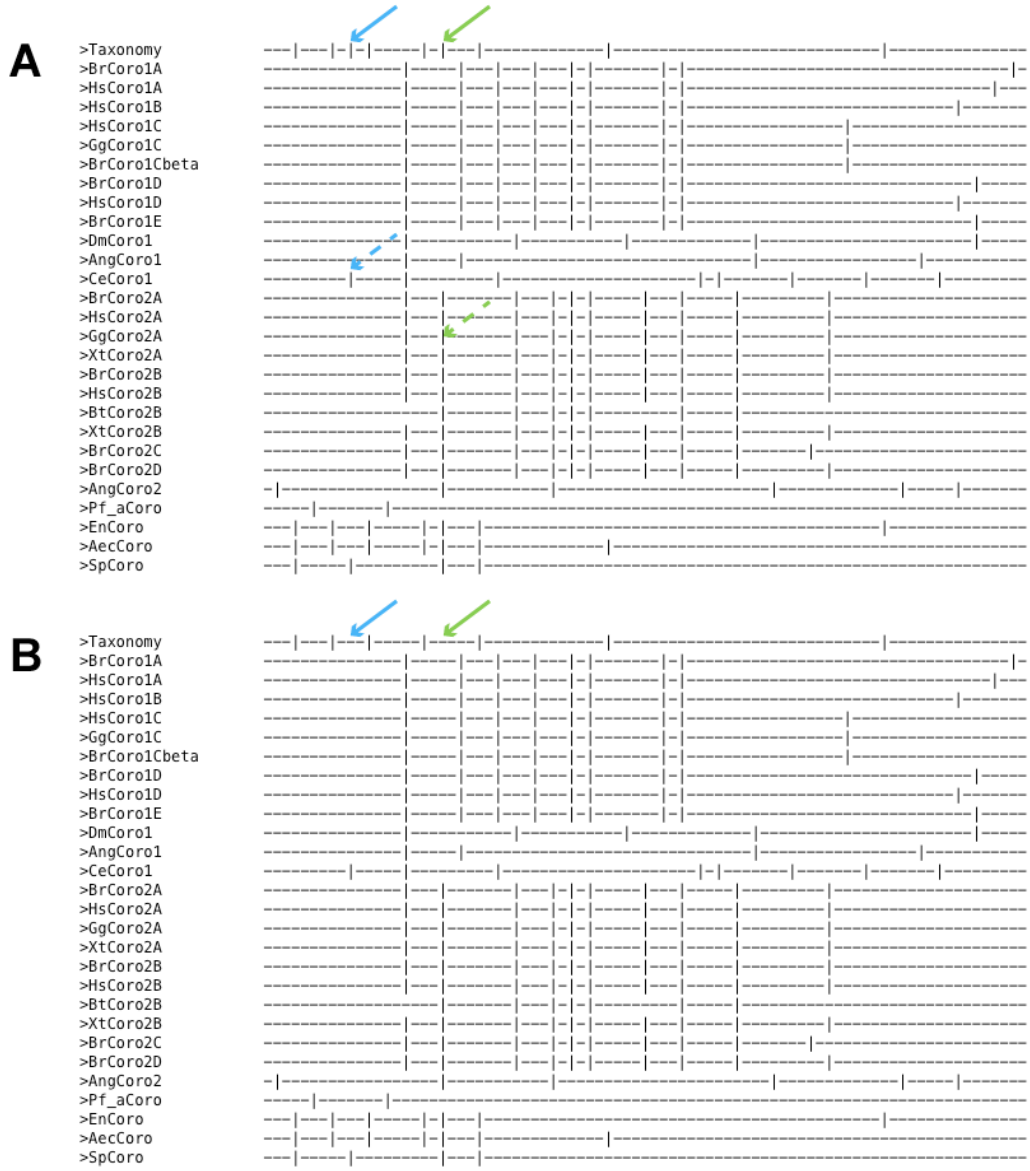

Figure 11 – A) Introns, which are common to fungi are included in the bottom-most exon-intron pattern. B) Only those introns that are exclusive for fungi are included in the bottom-most exon-intron pattern.

The parameter `--introns-per-taxon` creates an additional exon-intron pattern of all taxa with assigned introns. The file extension is `-taxonomy` (Figure 12).

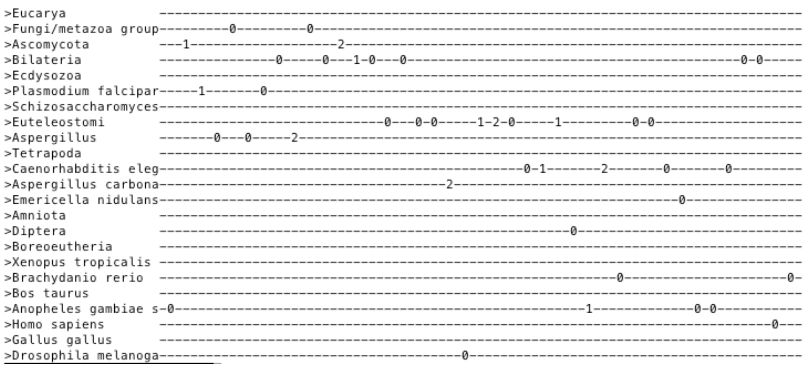

Figure 12 - Each intron is assigned to the last common ancestor of those species it occurs in. The introns are then combined to exon-intron patterns for each taxon.

Statistics

Adding `--statistics` generates statistics about each intron position. These statistics include the total number of introns at each position as well as the last common ancestor of all species harbouring the respective intron. In addition, the distribution of intron counts onto direct descendants of that last common ancestor is given. For a better overview, only intron positions occurring in human coronin genes were included in the figure (Figure 13).

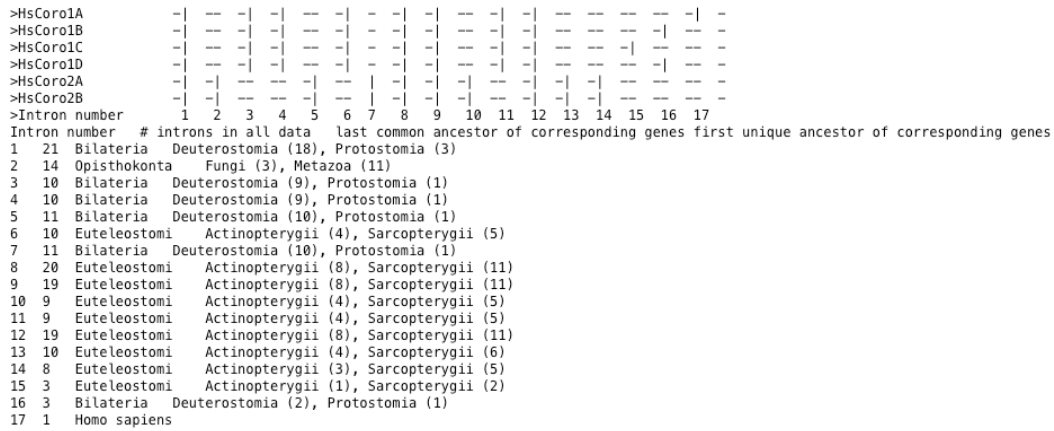

Figure 13 - Statistics per intron position, including the last common ancestor of all species encoding that intron

Data and output selection

GenePainter includes all genes in its analysis, for which both an aligned protein sequence and the gene structure are present. Assuming that the multiple sequence alignment contains all sequences of the dataset, the easiest way to analyse only a subset of all data is to copy the corresponding gene structure files to a new folder and specify that folder. A more elegant way to accomplish this task is to use the various data and output selection options GenePainter offers.

Subsets of data for analyse and visualization can be specified in various ways:

- A list of gene names corresponding to gene structure file names and fasta headers in the multiple sequence alignment (option `--selection-based-on-list`).
- Specifying a species name by using option `--selection-based-on-species`. This option is only possible together with `--taxonomy-to-fasta` to correctly associate genes with the selected species.
- A regular expression, which is then applied on gene names (option `--selection-based-on-regex`). The regular expression needs to be enclosed by quotation marks and specify only the regular expression itself, not the surrounding ("`/`"). Modifiers such as "`i`" (for case insensitivity) are not allowed. The validity of the regular expression can be checked on online platforms such as http://rubular.com/.

Analysing All Data, Visualizing All Data

This is the default. However, some of the output files might be confusing for large datasets. This is identical to setting `--analyse-all-output-all`.

Analysing All Data, Visualizing a Subset

For many purposes it is useful to focus the visualization on a specific subset of the data while all data is analysed. This is done by setting `--analyse-all-output-selection`.
For `--analyse-all-output-selection`, the complete dataset, i.e. introns occurring in all data, is analysed, but only gene structures of selected data are included in the output. For example, the dataset contains hundreds of genes from species all across the eukaryotic tree of life. Here, it might be better to focus the visualization on only the human or mammalian genes.

Analysing a Subset, Visualizing a Subset

In might be important to focus the analysis on the introns of a single species or a subset of species. For example, it might be interesting to analyse the evolution of the introns present in certain human genes, while the introns conserved in plants but not present in humans should be ignored. In this case, only the conservation of the introns present in this selection will be determined and visualized
(option `--analyse-selection-on-all-data-output-selection`).

To analyse and visualize only a selection of the data, use `--analyse-selection-on-all-data-output-selection`.