

## Annotation comparison Vetea Jacot

Generated by Doxygen 1.9.1



---

<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List . . . . .	1
<b>2 Namespace Documentation</b>	<b>3</b>
2.1 CDScompR Namespace Reference . . . . .	3
2.1.1 Detailed Description . . . . .	3
2.1.2 Function Documentation . . . . .	3
2.1.2.1 annotation_comparison() . . . . .	4
2.1.2.2 write_results() . . . . .	4
2.2 CDSmulticompR Namespace Reference . . . . .	5
2.2.1 Detailed Description . . . . .	5
2.2.2 Function Documentation . . . . .	5
2.2.2.1 multicompile() . . . . .	6
2.2.2.2 result_to_dict() . . . . .	6
2.2.2.3 write_multi_results() . . . . .	7
<b>Index</b>	<b>9</b>



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

<a href="#">CDScompR</a>	This script is used to compute the distance between two structural annotations of a same genome, one reference and one alternative annotation . . . . .	3
<a href="#">CDSmulticompR</a>	This script is used to compute the distance between multiple structural annotations of a same genome, one reference and one or more alternative annotations . . . . .	5



## Chapter 2

# Namespace Documentation

### 2.1 CDScompR Namespace Reference

This script is used to compute the distance between two structural annotations of a same genome, one reference and one alternative annotation.

#### Functions

- def `write_results` (all\_results, alt\_name, verbose=False)  
*This function writes to a new 'results.csv' file the results of the annotation comparison retrieved from the identities dictionary returned by the `annotation_match` function.*
- def `annotation_comparison` (ref\_path, alt\_path, verbose=False, create\_strings=False, exon\_mode=False)  
*Main function of this program.*
- def `usage` ()
- def `main` ()

#### Variables

- string `script_dir` = `os.path.dirname( __file__ ) + "/python_util/"`

#### 2.1.1 Detailed Description

This script is used to compute the distance between two structural annotations of a same genome, one reference and one alternative annotation.

It expects as input the paths to the annotation files (in GFF format), displays the computed distances between all annotation pairs, and creates a results CSV file detailing the loci comparisons between the annotations

#### 2.1.2 Function Documentation

### 2.1.2.1 annotation\_comparison()

```
def CDScompR.annotation_comparison (
    ref_path,
    alt_path,
    verbose = False,
    create_strings = False,
    exon_mode = False )
```

Main function of this program.

Given a reference and alternative path, gets the corresponding GFF files and compares the two annotations to return their information about their loci's comparison

#### Parameters

<i>ref_path</i>	Path of the GFF file describing the reference annotation
<i>alt_path</i>	Path of the GFF file describing the alternative annotation
<i>verbose</i>	If True, triggers display of more information messages. Default is 'False'
<i>create_strings</i>	Boolean indicating whether to use the 'old' comparison function (old_compare_loci, 'True') or the new one (compare_loci, 'False')
<i>exon_mode</i>	Boolean indicating if the main comparison structures read from the file should be coding sequences (CDS, False) or exons (True). Default is 'False' (CDS comparison)

#### See also

compare\_loci()  
old\_compare\_loci()

#### Returns

Returns a list of lists of dictionaries describing the comparison of the structure identity between the loci of each annotation

### 2.1.2.2 write\_results()

```
def CDScompR.write_results (
    all_results,
    alt_name,
    verbose = False )
```

This function writes to a new 'results.csv' file the results of the annotation comparison retrieved from the identities dictionary returned by the annotation\_match function.

#### See also

annotation\_match()



## Parameters

<i>results</i>	A dictionary of list of list of dictionaries containing results of the annotation comparison, as returned by <code>annotation_match</code>
<i>alt_name</i>	String indicating the name of the original alternative annotation file
<i>verbose</i>	If True, triggers display of more information messages. Default is 'False'

## See also

`annotation_match()`

## Remarks

Results are written in CSV ('Comma-Separated Values') format

## 2.2 CDSmulticompR Namespace Reference

This script is used to compute the distance between multiple structural annotations of a same genome, one reference and one or more alternative annotations.

### Functions

- def `result_to_dict` (result)  
*Extracts the computed identity for each locus of the given result dictionary and add it to a new dictionary.*
- def `write_multi_results` (multi\_results, ref\_path, alt\_paths)  
*Writes the results returned by the function multicomp into a results synthesis CSV file detailing the loci identity for each alternative.*
- def `multicomp` (ref\_path, alt\_paths, verbose, create\_strings, exon\_mode)  
*Compares all alternative annotations given (alt\_paths) to the reference annotation (ref\_path) by calling annotation<sub>↔</sub>\_sort and appends the loci identities to a list.*
- def `usage` ()
- def `main` ()

#### 2.2.1 Detailed Description

This script is used to compute the distance between multiple structural annotations of a same genome, one reference and one or more alternative annotations.

It expects as input the paths to the annotation files (in GFF format), displays the computed distances between all annotation pairs, and creates results CSV files for each alternative and one global synthesis file

#### 2.2.2 Function Documentation

### 2.2.2.1 multicomp()

```
def CDSmulticompr.multicomp (
    ref_path,
    alt_paths,
    verbose,
    create_strings,
    exon_mode )
```

Compares all alternative annotations given (*alt\_paths*) to the reference annotation (*ref\_path*) by calling *annotation\_sort* and appends the loci identities to a list.

#### See also

*annotation\_sort()*

#### Parameters

<i>ref_path</i>	Path to the reference annotation file
<i>alt_path</i>	Path to the alternative annotation file
<i>verbose</i>	If True, triggers display of more information messages. Default is 'False'
<i>create_strings</i>	Boolean indicating whether to use the 'old' comparison function ( <i>old_compare_loci</i> , 'True') or the new one ( <i>compare_loci</i> , 'False')
<i>exon_mode</i>	Boolean indicating if the main comparison structures read from the file should be coding sequences (CDS, False) or exons (True). Default is 'False' (CDS comparison)

#### See also

*compare\_loci()*

*old\_compare\_loci()*

#### Returns

Returns the list of all loci identities (dictionaries) of all alternatives

### 2.2.2.2 result\_to\_dict()

```
def CDSmulticompr.result_to_dict (
    result )
```

Extracts the computed identity for each locus of the given result dictionary and add it to a new dictionary.

#### Parameters

<i>result</i>	Result dictionary, as returned by <a href="#">CDScompR</a> 's <i>annotation_compare()</i> function
---------------	--

### Returns

Returns a dictionary with the locus name as key, and the comparison identity as value

#### 2.2.2.3 write\_multi\_results()

```
def CDSmulticompR.write_multi_results (
    multi_results,
    ref_path,
    alt_paths )
```

Writes the results returned by the function multicomp into a results synthesis CSV file detailing the loci identity for each alternative.

### See also

[multicomp\(\)](#)

### Parameters

<i>multi_results</i>	List of loci identities (dictionaries), as returned by multicomp
<i>ref_path</i>	Path to the reference annotation GFF file



# Index

annotation\_comparison  
    CDScompR, [3](#)

CDScompR, [3](#)  
    annotation\_comparison, [3](#)  
    write\_results, [4](#)

CDSmulticompR, [5](#)  
    multicomp, [5](#)  
    result\_to\_dict, [6](#)  
    write\_multi\_results, [7](#)

multicomp  
    CDSmulticompR, [5](#)

result\_to\_dict  
    CDSmulticompR, [6](#)

write\_multi\_results  
    CDSmulticompR, [7](#)

write\_results  
    CDScompR, [4](#)