

API Documentation

API Documentation

October 25, 2016

Contents

Contents	1
1 Script script-EveryGO	2
1.1 Functions	2
1.2 Variables	3
2 Script script-GOslimmer_xml	4
2.1 Functions	4
2.2 Variables	4
3 Script script-GOtrimmer	5
3.1 Functions	5
3.2 Variables	5
4 Script script-GoXML	6
4.1 Functions	6
4.2 Variables	8
5 Script script-VCFtoGO	9
5.1 Functions	9
5.2 Variables	11

1 Script script-EveryGO

EveryGO workflow. In order to perform GO-terms enrichment analysis for one or more nodes/comparisons of the phylogeny stored in the XML file.

Requires:

- python 2.7¹ (tested with 2.7.6)
- lxml²
- GOslimmer_xml
- GOwalker
- GoView

1.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(*type=parser object*)

makeuniverse(xmlname)

Make universe of GO-terms from the XML file

Parameters

xmlname: the xml filename

(*type=string*)

everygo(compvalues, view, xmlname, Rpath)

Make enrichment analysis for each comparison values

Parameters

compvalues: comparisons values returned by getcomp()

(*type=string*)

view: -view argument -> if view != 0 , graphical representation of enrichment will be processed

(*type=int*)

xmlname: the xml filename

(*type=string*)

Rpath: the R full path

(*type=string*)

¹<https://www.python.org/downloads/>

²<https://github.com/lxml/lxml>

main()

1.2 Variables

Name	Description
<code>__doc__</code>	Value: ...
<code>__package__</code>	Value: None

2 Script script-GOslimmer_xml

GOslimmer script. In order to append prokaryotic terms only in XML output file, based on GOslim (GO-basic.obo)

Requires:

- VCFtoGO output file (GO-terms universe)
- GO-basic.obo file (GOslim)

2.1 Functions

add_object(*d*)

collect prokaryotic terms in GO-basic.obo

Parameters

d: go-term related data collected from GO-basic.obo
(*type=object data*)

main()

2.2 Variables

Name	Description
__doc__	Value: ...
fname	Value: 'db/go-basic.obo'
term_head	Value: '[Term]'
all_objects	Value: {'GO:0000001': [], 'GO:0000002': [], 'GO:0000003': ['gosl...']}
__package__	Value: None

3 Script script-GOtrimmer

3.1 Functions

get_parser()
Arguments setting and parsing
Return Value
arguments list
<i>(type=parser object)</i>

add_object(<i>d</i>)
collect prokaryotic terms in GO-basic.obo
Parameters
d: go-term related data collected from GO-basic.obo
<i>(type=object data)</i>

3.2 Variables

Name	Description
<code>__doc__</code>	Value: ...
<code>parser</code>	Value: <code>get_parser()</code>
<code>Arguments</code>	Value: <code>parser.parse_args()</code>
<code>fname</code>	Value: <code>"../db/go-basic.obo"</code>
<code>term_head</code>	Value: <code>"[Term]"</code>
<code>all_objects</code>	Value: <code>{}</code>
<code>output</code>	Value: <code>open("goprok.txt", "w")</code>
<code>gofull</code>	Value: <code>str(Arguments.input)</code>
<code>outputuniv</code>	Value: <code>open("VCFtoGOresults/univers.txt", "w")</code>
<code>current</code>	Value: <code>defaultdict(list)</code>

4 Script script-GoXML

GoXML workflow. In order to add GO-terms from a XML file and generate a non-enriched universe of GO-terms.

Requires:

- python 2.7³ (tested with 2.7.6)
- PyVCF⁴
- lxml⁵
- GOslimmer_xml

4.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

readDBFile(fichier)

Read NP-EBI local storage file

Parameters

fichier: NP-EBI_table.tsv file

(type=string)

Return Value

dictionnary of NP identifiers

(type=dictionnary)

readDBGOF(fichier)

Read EBI-GO local storage file

Parameters

fichier: EBI-GO_table.tsv file

(type=string)

Return Value

dictionnary of GO identifiers

(type=dictionnary)

³<https://www.python.org/downloads/>

⁴<https://github.com/jamescasbon/PyVCF>

⁵<https://github.com/lxml/lxml>

extractNP(*path*)

Retrieve NP ids from XML file

Parameters

path: relative path to the XML file
(*type=string*)

Return Value

dictionary of NP identifiers
(*type=dictionnary*)

idmapping(*nplist*, *dicoNP*, *NPfile*)

Make ID mapping from NP identifiers to Uniprot identifiers requesting Online(Uniprot database) or Offline(NP-EBI_databases.tsv)

Parameters

nplist: list used in ONLINE request -> values = NP identifiers
(*type=list*)

dicoNP: dictionary used in OFFLINE request -> keys = NP identifiers ,
values = uniprot id retrieved
(*type=dictionnary*)

NPfile: Storage of uniprot identifiers to NP identifiers (NP-EBI_databases.tsv)
(*type=string*)

Return Value

list of Uniprot identifiers
(*type=list*)

UNIPROTtoGO(*path*, *output*, *protlist*, *dicoGO*, *GOFile*)

Make ID mapping from Uniprot identifiers to GO identifiers requesting Online(Quick-GO database) or Offline(EBI-GO_databases.tsv)

Parameters

path: relative path to the XML file
(*type=string*)

output: Universe of GO identifiers (univers.txt)
(*type=string*)

protlist: list used in ONLINE request -> values = UNIPROT identifiers
(*type=list*)

dicoGO: dictionary used in ONLINE requests -> keys = Uniprot identifiers ,
values = GO identifiers list
(*type=dictionnary*)

GOFile: Storage of uniprot identifiers to GO identifiers
(EBI-GO_databases.tsv)
(*type=string*)

main()

4.2 Variables

Name	Description
<code>__doc__</code>	Value: ...
<code>__package__</code>	Value: None

5 Script script-VCFtoGO

VCFtoGO workflow. In order to generate non-enriched universe of GO-terms from a VCF.

Requires:

- python 2.7⁶ (tested with 2.7.6)
- PyVCF⁷
- GOftrimmer

5.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(*type=parser object*)

readDBFile(fichier)

Read NP-EBI local storage file

Parameters

fichier: NP-EBI_databases.tsv file

(*type=.tsv file*)

Return Value

dictionnary of NP identifiers

(*type=dictionnary*)

readDBGOFFile(fichier)

Read EBI-GO local storage file

Parameters

fichier: EBI-GO_databases.tsv file

(*type=.tsv file*)

Return Value

dictionnary of GO identifiers

(*type=dictionnary*)

⁶<https://www.python.org/downloads/>

⁷<https://github.com/jamescasbon/PyVCF>

extractinfopos(*fichier, pos*)

Parse NP identifiers from VCF file

Parameters

fichier: VCF file
(type=.vcf file)

pos: positions of interest file
(type=.txt file)

Return Value

dictionary of NP identifiers
(type=dictionnary)

idmapping(*dico, dicoNP, NPfile*)

Make ID mapping from NP identifiers to Uniprot identifiers requesting Online(Uniprot database) or Offline(NP-EBI_databases.tsv)

Parameters

dico: dictionnary used in ONLINE request -> keys = NP identifiers , values = uniprot id retrieved
(type=dictionnary)

dicoNP: dictionnary used in OFFLINE request -> keys = NP identifiers , values = uniprot id retrieved
(type=dictionnary)

NPfile: Storage of uniprot identifiers to NP identifiers (NP-EBI_databases.tsv)
(type=.tsv file)

UNIPROTtoGO(*dico, output, pos, dicoGO, GOfile*)

Make ID mapping from Uniprot identifiers to GO identifiers requesting Online(Quick-GO database) or Offline(EBI-GO_databases.tsv)

Parameters

dico: dictionnary used in OFFLINE requests -> keys = Uniprot identifiers , values = GO identifiers list
(type=dictionnary)

output: Universe of GO identifiers (univers.txt)
(type=.txt file)

pos: positions requested if -pos argument was used
(type=int)

dicoGO: dictionnary used in ONLINE requests -> keys = Uniprot identifiers , values = GO identifiers list
(type=dictionnary)

GOfile: Storage of uniprot identifiers to GO identifiers (EBI-GO_databases.tsv)
(type=.tsv file)

main()

5.2 Variables

Name	Description
<code>__doc__</code>	Value: ...
<code>__package__</code>	Value: None

Index

- script-EveryGO (*script*), 2–3
 - script-EveryGO.everygo (*function*), 2
 - script-EveryGO.get_parser (*function*), 2
 - script-EveryGO.main (*function*), 2
 - script-EveryGO.makeuniverse (*function*), 2
- script-GOslimmer_xml (*script*), 4
 - script-GOslimmer_xml.add_object (*function*), 4
 - script-GOslimmer_xml.main (*function*), 4
- script-GOtrimmer (*script*), 5
 - script-GOtrimmer.add_object (*function*), 5
 - script-GOtrimmer.get_parser (*function*), 5
- script-GoXML (*script*), 6–8
 - script-GoXML.extractNP (*function*), 6
 - script-GoXML.get_parser (*function*), 6
 - script-GoXML.idmapping (*function*), 7
 - script-GoXML.main (*function*), 7
 - script-GoXML.readDBFile (*function*), 6
 - script-GoXML.readDBGOFfile (*function*), 6
 - script-GoXML.UNIPROTtoGO (*function*), 7
- script-VCFtoGO (*script*), 9–11
 - script-VCFtoGO.extractinfopos (*function*), 9
 - script-VCFtoGO.get_parser (*function*), 9
 - script-VCFtoGO.idmapping (*function*), 10
 - script-VCFtoGO.main (*function*), 10
 - script-VCFtoGO.readDBFile (*function*), 9
 - script-VCFtoGO.readDBGOFfile (*function*), 9
 - script-VCFtoGO.UNIPROTtoGO (*function*), 10