Package ‘AnVILWorkflow’

May 8, 2024

Title  Run workflows implemented in Terra/AnVIL workspace
Version 1.4.0
Date 2024-2-27

Description The AnVIL is a cloud computing resource developed in part
by the National Human Genome Research Institute. The main cloud-based
genomics platform deployed by the AnVIL project is Terra. The
AnVILWorkflow package allows remote access to Terra implemented workflows,
enabling end-user to utilize Terra/AnVIL provided resources - such as
data, workflows, and flexible/scalable computing resources - through the
conventional R functions.

Depends R (>= 4.2.0),
Imports utils, AnVIL, httr, methods, jsonlite, dplyr, tibble
Suggests knitr, BiocStyle
License Artistic-2.0
biocViews Infrastructure, Software
Encoding UTF-8
LazyData true
VignetteBuilder knitr
Rooxygen list(markdown = TRUE)
RooxygenNote 7.2.3

BugReports https://github.com/shbrief/AnVILWorkflow/issues
git_url https://git.bioconductor.org/packages/AnVILWorkflow
git_branch RELEASE_3_19
git_last_commit c7d0e1b
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-08

Author Sehyun Oh [aut, cre] (<https://orcid.org/0000-0002-9490-3061>),
Kai Gravel-Pucillo [aut]

Maintainer Sehyun Oh <shbrief@gmail.com>
Contents

- .biobakery_currentInput .................................................. 2
- .get_workflow_fullname .................................................. 3
- .get_workspace_fullname .................................................. 3
- .nonMetadataOutputs ....................................................... 4
- .search_keyword ............................................................ 5
- .stop_quietly ..................................................................... 5
- AnVILBrowse .................................................................... 6
- availableAnalysis .............................................................. 7
- cloneWorkspace .................................................................. 8
- currentInput ..................................................................... 9
- findInputName ................................................................... 10
- getAllDataTables .............................................................. 10
- getAllWorkflows ............................................................... 11
- getAllWorkspaces ............................................................. 12
- getDashboard .................................................................... 12
- getData ............................................................................. 13
- getOutput .......................................................................... 13
- getWorkflowConfig ............................................................ 14
- getWorkflows .................................................................... 15
- getWorkspaces ................................................................... 15
- monitorWorkflow ............................................................... 16
- runWorkflow ...................................................................... 16
- setCloudEnv ...................................................................... 17
- stopWorkflow ..................................................................... 18
- updateInput ....................................................................... 19

Index .................................................................................... 21

---

**.biobakery_currentInput**

*Check the current input arguments*

**Description**

Check the current input arguments

**Usage**

```
.biobakery_currentInput(config)
```

**Arguments**

- **config**  
  Workflow configuration. Output from the `getWorkflowConfig` function.

**Value**

A list length of two, including `inputListPath` and `inputFilePath`. 
Examples

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  config <- avworkflow_configuration_get(
    workflow_namespace = "mtx_workflow_biobakery_version3",
    workflow_name = "mtx_workflow_biobakery_version3",
    workspace_namespace = "waldronlab-terra-rstudio",
    workspace_name = "mtx_workflow_biobakery_version3_template"
  )
  biobakery_inputs <- .biobakery_currentInput(config)
}
```

---

`.get_workflow_fullname`

*Get the workflow namespace and name*

Description

Use this internally when `setCloudEnv` is already run.

Usage

```
.get_workflow_fullname(workspaceName, workflowName = NULL)
```

Arguments

- `workspaceName`: A character. Name of the workspace to use.
- `workflowName`: A character. Name of the workflow to run. If a single workflow is available under the selected workspace, this function will check the input of that workflow under the default (NULL). If there are multiple workflows available, you should specify the workflow.

Value

A character of `workflow_namespace/workflow_name`

---

`.get_workspace_fullname`

*Get the fullname of the workspace*

Description

Get the fullname of the workspace
Usage

.glob_workspace_fullname(workspaceName)

Arguments

workspaceName Character(1). Name of the template workspace name you want to clone. You can provide name or namespace/name.

Value

Character(1) of workspaceNamespace/workspaceName

Examples

library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
    .get_workspace_fullname(workspaceName = "Bioconductor-Workflow-DESeq2")
}

---

.nonMetadataOutputs Subset to non-metadata output files

Description

Subset to non-metadata output files

Usage

.nonMetadataOutputs(workflowOutputs)

Arguments

workflowOutputs

A data frame of workflow outputs with four columns: file, workflow, task, and path. Returned value from avworkflow_files.

Value

A character vector containing the names of non-metadata output files
.search_keyword

Search keywords in a given metadata table

Description
Search keywords in a given metadata table

Usage
.search_keyword(keyword, metadata)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyword</td>
<td>A character(1). Regular expression is accepted. For example, you can search multiple keywords separated by the vertical bar (&quot;</td>
</tr>
<tr>
<td>metadata</td>
<td>A data frame. Metadata table of workspace, workflow, or AnVIL data.</td>
</tr>
</tbody>
</table>

Value
A data frame. A subset of input metadata table with the rows containing the keyword.

.stop_quietly

Stop the execution without error messages

Description
Stop the execution without error messages

Usage
.stop_quietly()

Value
Stop the function call without warning/error messages.
AnVILBrowse

Search AnVIL workspaces using keywords

Description

Search AnVIL workspaces using keywords

Usage

AnVILBrowse(
  keyword,  
  searchFrom = "all",  
  returnFrom = NULL,  
  metaTables = "default",  
  minAge = 0,  
  maxAge = 130,  
  minCount = 0,  
  workspaceTable = NULL,  
  workflowTable = NULL,  
  dataTable = NULL
)

Arguments

keyword  A character(1). Regular expression is accepted. For example, you can search multiple keywords separated by the vertical bar ("|").

searchFrom  Under the default (all), all the workspaces containing keywords in their workspace/workflow/data will be returned. The other available options are workspace, workflow, and data.

returnFrom  Under the default (NULL), the same data type as for searchFrom will be used, while searchFrom = "all" returns workspaces.

metaTables  Under the default (default), all the publicly accessible AnVIL workspaces will be subjected for search. If you want to search in all the workspaces you have access to, set this argument as custom, and provide the inputs for workspaceTable, workflowTable, and dataTable arguments.

minAge  A number. Any data with a maximum participant age lower than this parameter will be excluded from the output. Under the default (0), no data entries will be removed due to the maximum participant age. Data entries with no maximum participant age listed will not be removed by this argument.

maxAge  A number. Any data with a minimum participant age higher than this parameter will be excluded from the output. Under the default (130), no data entries will be removed due to the minimum participant age. Data entries with no minimum participant age listed will not be removed by this argument.
availableAnalysis

minCount
A number. Any data with a participant count fewer than this parameter will be excluded from the output. Under the default (0), no data entries will be removed due to participant count. Data entries with no participant count listed will not be removed by this argument.

workspaceTable
A data frame. This argument is counted only when metaTables = "custom". Provide the output from the getWorkspaces function, to search in all the workspaces you have access to.

workflowTable
A data frame. This argument is counted only when metaTables = "custom". Provide the output from the getWorkflows function, to search in all the workflows you have access to.

dataTable
A data frame. This argument is counted only when metaTables = "custom". Provide the output from the getData function, to search in all the AnVIL data you have access to.

Value
A data frame of AnVIL resources containing keywords. Depending on the returnFrom argument, it can be workspaces, workflows, or data. Under the default returnFrom = NULL, it returns the same data type as specified in searchFrom or workspace for searchFrom = "all".

Examples
AnVILBrowse("malaria")
AnVILBrowse("resistance")
AnVILBrowse("resistance", searchFrom = "workflow")

---

availableAnalysis Find the available analysis

Description
This function shows the available analyses and the brief descriptions of them.

Usage
availableAnalysis(curatedOnly = TRUE, keyword = NULL)

Arguments

  curatedOnly
  Default is TRUE, returning only workspaces that offer simplified input configuration by this package. If it is set to FALSE, all the workspaces

  keyword
  Default is NULL. When this argument is provided as a character(1), it will return only the workspaces containing the keyword and the user has an access to.
Value

A data frame. The analysis columns shows the name of the available analyses, which is the required input (analysis argument) for the functions implemented in AnVILWorkflow package.

Examples

library(AnVIL)
if (gcloud_exists() & nzchar(avworkspace_name())) {availableAnalysis()}

---

closeWorkspace 

Clone template workspace

Description

This function makes your own copy of the existing workspace, selected through templateName or analysis. Your copied/clone workspace name will be workspaceName and any computing cost will be charaged to the billing linked to your billingProjectName. You should provide at least one argument templateName or analysis.

Usage

cloneWorkspace(
  workspaceName,
  templateName = "",
  analysis = NULL,
  accountEmail = gcloud_account(),
  billingProjectName = gcloud_project()
)

Arguments

<table>
<thead>
<tr>
<th>argument</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workspaceName</td>
<td>Name of the workspace you are creating</td>
</tr>
<tr>
<td>templateName</td>
<td>Character(1). Name of the template workspace name you want to clone. You can provide name or namespace/name.</td>
</tr>
<tr>
<td>analysis</td>
<td>Character(1). Name of the analysis you want to clone it's workspace. The list of available analyses can be found using availableAnalysis.</td>
</tr>
<tr>
<td>accountEmail</td>
<td>Character(1). Email linked to Terra account</td>
</tr>
<tr>
<td>billingProjectName</td>
<td>Character(1). Name of the billing project</td>
</tr>
</tbody>
</table>

Value

Name of the cloned workspace
currentInput

Examples

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  cloneWorkspace(workspaceName = "salmon",
                 templateName = "Bioconductor-Workflow-DESeq2")
}
```

currentInput | Check the current input arguments

Description

Check the current input arguments

Usage

`currentInput(workspaceName, config, requiredInputOnly = TRUE, analysis = NULL)`

Arguments

- `workspaceName` Name of the workspace
- `config` Workflow configuration. Output from the `getWorkflowConfig` function.
- `requiredInputOnly` Under the default (TRUE), only the required inputs are returned.
- `analysis` If specified, only the minimally required inputs for a given workflow will be returned.

Value

A data.frame for the inputs defined in a workflow configuration.

Examples

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  workspaceName <- "Bioconductor-Workflow-DESeq2"
  config <- getWorkflowConfig(workspaceName)
  currentInput(workspaceName = workspaceName, config = config)
}
```
**findInputName**  
*Find the root entity name*

**Description**

Find the root entity name

**Usage**

```r
findInputName(workspaceName, rootEntity = "", nameOnly = TRUE)
```

**Arguments**

- `workspaceName`: Name of the workspace
- `rootEntity`: A character. Type of root entity for Terra’s data model. For example, `participant`, `participant_set`, `sample`, etc.
- `nameOnly`: Under the default (TRUE), only the names of a given root entity type will be returned.

**Value**

A character vector of input names under the given root entity.

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  .findInputName(
    workspaceName = "Bioconductor-Workflow-DESeq2",
    rootEntity = "participant_set"
  )
}
```

---

**getAllDataTables**  
*Get all the data tables*

**Description**

Get all the data tables

**Usage**

```r
ggetAllDataTables(workspaces = NULL)
```
**getAllWorkflows**

**Arguments**

workspaces  A character vector. Under the default (NULL), all the data tables from all the workspaces user has access to will be returned. If you specify this, the data tables only from the specified workspace(s) will be returned.

**Value**

A Data Frame of all the data tables

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  allDataTables <- getAllDataTables()
}
```

---

**getAllWorkflows**  
*Collect workflows from all workspaces a user has access to*

**Description**

Collect workflows from all workspaces a user has access to

**Usage**

```
getAllWorkflows(workspaces = NULL)
```

**Arguments**

workspaces  Under the default (NULL), workflows from all the workspaces a user has access to will be collected.

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  allWorkflows <- getAllWorkflows()
}
```
getAllWorkspaces  
*Get AnVIL workspaces*

**Description**

Different from `avworkspaces` [https://drive.google.com/drive/u/0/folders/1NNAzcNRBx4nPfcdfqjKPeUJlVE7lhxXM6L](https://drive.google.com/drive/u/0/folders/1NNAzcNRBx4nPfcdfqjKPeUJlVE7lhxXM6L)

**Usage**

```r
getAllWorkspaces()
```

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
    allWorkspaces <- getAllWorkspaces()
}
```

---

getDashboard  
*Print out Dashboard contents*

**Description**

This function prints out the Dashboard contents of the target workspace. You can provide either `workspaceName` or `analysis`. If both values are provided, this function will use `workspaceName` argument over `analysis` argument.

**Usage**

```r
getDashboard(workspaceName = "", analysis = NULL)
```

**Arguments**

- `workspaceName`: The name of the workspace you want to get the overview provided through the Dashboard.
- `analysis`: The name of the analysis use want to check the Dashboard of. The list of available analyses can be found with `availableAnalysis()`.

**Value**

The last modified date as a message, followed by the Dashboard contents from the target workspace.
Examples

```r
library(AnVIL)
if (gcloud_exists() && nzchar(awworkspace_name())) {
  getDashboard(analysis = "salmon")
  getDashboard(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

**getData**

*Creates a metadata table of data from all workspaces provided*

**Description**

This function usually takes a long time to run due to the large volume of AnVIL data.

**Usage**

```r
getData(allWorkspaces)
```

**Arguments**

- `allWorkspaces` - A data frame of all the workspaces you have access to. An output from the `getWorkspaces` function.

**getOutput**

*Download output files from Terra*

**Description**

Download output files from Terra

**Usage**

```r
getOutput(
  workspaceName,
  submissionId = NULL,
  keyword = NULL,
  dest_dir = ".",
  dry = TRUE
)
```
Arguments

workspaceName   Name of the workspace
submissionId   Submission Id. If it’s not provided, the most recent submission id with the 'succeeded' status will be used.
keyword   A character string containing a regular expression to be matched in the output file name. Under the default NULL, all the outputs from the workflow, including log files, will be returned.
dest_dir   Path to the directory where downloaded files are saved
dry   To download the output data, set dry = FALSE.

Value

If "dry=TRUE", this function will return a data frame with two columns named 'filename' and 'name'.

- filename: Name of the actual output files.
- name: Name of the output defined in your workflow script. This is how configuration refers the outputs.

Examples

library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  getOutput(workspaceName = "Bioconductor-Workflow-DESeq2")
}

---

getWorkflowConfig   Check the workflow configuration

Description

Check the workflow configuration

Usage

getWorkflowConfig(workspaceName, workflowName = NULL)

Arguments

workspaceName   Name of the workspace
workflowName   Name of the workflow to run. If a single workflow is available under the specified workspace, this function will check the input of that workflow under the default (NULL). If there are multiple workflows available, you should specify the workflow.
**Value**

A data.frame for the inputs defined in a workflow configuration.

**Examples**

```r
library(AnVIL)
if (gccloud_exists() && nzchar(avworkspace_name())) {
  config <- getWorkflowConfig(workspaceName = "Bioconductor-Workflow-DESeq2")
  config
}
```

---

**getWorkflows**

*Creates a metadata table of workflows from all workspaces provided*

**Description**

Creates a metadata table of workflows from all workspaces provided

**Usage**

```r
getWorkflows(allWorkspaces)
```

**Arguments**

- `allWorkspaces` A data frame of all the workspaces you have access to. An output from the `getWorkspaces` function.

---

**getWorkspaces**

*Creates a metadata table of all workspaces*

**Description**

Creates a metadata table of all workspaces

**Usage**

```r
getWorkspaces()
```
runWorkflow

monitorWorkflow  

**Description**

Check the status of submitted jobs

**Usage**

```
monitorWorkflow(workspaceName)
```

**Arguments**

- **workspaceName**  
  Character(1). Name of the workspace

**Value**

A tibble summarizing submitted workflow jobs. Contains information such as submission Id, submission date, and submission status.

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  monitorWorkflow(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

runWorkflow

**Description**

Launch Terra workflow

**Usage**

```
runWorkflow(
  workspaceName,
  config,
  workflowName = NULL,
  useCallCache = TRUE,
  inputName = NULL
)
```
setCloudEnv

**Arguments**

- **workspaceName**: Name of the workspace that contains the workflow(s) you want to launch.
- **config**: Workflow configuration. Output from the `getWorkflowConfig` function.
- **workflowName**: Name of the workflow to run. If this input is not provided but there is only a single workflow available, the function will automatically use the only workflow.
- **useCallCache**: A logical. Under the default condition (TRUE), call cache will be used.
- **inputName**: Name of you input entity. If the workflow is using Terra’s data model, this is required. The available entities can be found using the `findInputName` function.

**Value**

This function will print out whether the call for workflow launching was successful or not.

**Examples**

```r
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  if ("salmon" %in% avworkspaces()$name)
    runWorkflow(workspaceName = "salmon")
}
```

---

**Description**

Setup Google Cloud Account and Project

**Usage**

```r
setCloudEnv(
  accountEmail = gcloud_account(),
  billingProjectName = gcloud_project(),
  message = TRUE
)
```

**Arguments**

- **accountEmail**: Character(1). Email linked to your Terra account.
- **billingProjectName**: Character(1). Name of the billing project, which is the gcloud account.
- **message**: Under the default (TRUE), this function will print out Google Cloud Account and Billing Project set in the working environment.
stopWorkflow

Value
Terra/AnVIL working environment - Google Cloud billing account and the billing project name - will be printed out.

Examples
library(AnVIL)
if (gcloud_exists()) {
  setCloudEnv()
}

stopWorkflow
Abort submitted job

Description
Abort submitted job

Usage
stopWorkflow(workspaceName, submissionId = NULL, dry = TRUE)

Arguments

workspaceName  Name of the workspace
submissionId   A character. Submission ID you want to abort. You can find the submission id using monitorWorkflow function. If it is not defined, the most recent submission will be aborted.
dry            Logical(1) when TRUE (default), report the consequences but do not perform the action requested. When FALSE, perform the action.

Value
This function will print out whether the call for workflow abortion was successful or not. In case it was unsuccessful, the diagnosis will be suggested as a part of the message.

Examples
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  if ("salmon" %in% avworkspaces()$name)
    stopWorkflow(workspaceName = "salmon")
}
updateInput

Update the input

Description
Update the input

Usage
updateInput(
  workspaceName,
  inputs,
  config,
  workflowName = NULL,
  dry = TRUE,
  verbose = TRUE
)

Arguments

workspaceName Name of the workspace
inputs A tibble containing new input values. Provide the modified version of the current
input table, which is the output from currentInput function. IMPORTANT: all
the attributes should be provided as a character vector format and any string type
attributes (inputType = String) should have escaped quotation mark (\"").
config Workflow configuration. Output from the getWorkflowConfig function.
workflowName Name of the workflow to run. If a single workflow is available under the spec-
ified workspace, this function will check the input of that workflow under the
default (NULL). If there are multiple workflows available, you should specify the
workflow.
dry Logical(1). When TRUE (default), report the updated configuration but do not
perform the action requested in Terra. When FALSE, inputs in Terra/AnVIL will
updated.
verbose Logical(1). When TRUE (default), this function will print the updated input.

Value
With verbose=TRUE, a list of updated inputs will be printed. A successful execution of the function
will update the input configuration of the target workflow in Terra/AnVIL.

Examples

library(AnVIL)
if (gccloud_exists() && nzchar(avworkspace_name())) {
  if ("salmon" %in% awworkspaces()$name) {
    config <- getWorkflowConfig(workspaceName = "salmon")
inputs <- currentInput("salmon", config)
## Modify the contents of 'inputs' table for your analysis
updateInput("salmon", inputs, config)
Index

* internal
  .biobakery_currentInput, 2
  .get_workflow_fullname, 3
  .get_workspace_fullname, 3
  .nonMetadataOutputs, 4
  .stop_quietly, 5
  .biobakery_currentInput, 2
  .get_workflow_fullname, 3
  .get_workspace_fullname, 3
  .nonMetadataOutputs, 4
  .search_keyword, 5
  .stop_quietly, 5

AnVILBrowse, 6
availableAnalysis, 7, 8
avworkflow_files, 4
avworkspaces, 12

cloneWorkspace, 8
currentInput, 9, 19

findInputName, 10

getAllDataTables, 10
getAllWorkflows, 11
getAllWorkspaces, 12
getDashboard, 12
data, 13
getOutput, 13
getWorkflowConfig, 2, 9, 14, 17, 19
getWorkflows, 15
getWorkspaces, 15

monitorWorkflow, 16
runWorkflow, 16

setCloudEnv, 3, 17
stopWorkflow, 18

updateInput, 19