

Package ‘qsub’

July 22, 2025

Title Running Commands Remotely on 'Gridengine' Clusters

Version 1.1.3

Description Run lapply() calls in parallel by submitting them to 'gridengine' clusters using the 'qsub' command.

Depends R (>= 3.0)

Imports dplyr, glue, methods, pbapply, processx, purrr, random, readr, ssh, stringr, tidyr, tools, utils

Suggests testthat

URL <https://github.com/rcannood/qsub>

BugReports <https://github.com/rcannood/qsub/issues>

RoxygenNote 7.1.1

Encoding UTF-8

License GPL-3

NeedsCompilation no

Author Robrecht Cannoodt [aut, cre] (ORCID: <<https://orcid.org/0000-0003-3641-729X>>, github: rcannood),
Wouter Saelens [aut] (ORCID: <<https://orcid.org/0000-0002-7114-6248>>, github: zouter)

Maintainer Robrecht Cannoodt <rcannood@gmail.com>

Repository CRAN

Date/Publication 2021-09-23 10:10:02 UTC

Contents

cat_remote	2
cp_remote	3
create_qsub_config	4
create_ssh_connection	7
file_exists_remote	7
get_default_qsub_config	8

instantiate_qsub_config	8
is_job_running	9
is_qsub_config	9
is_remote_local	9
ls_remote	10
mkdir_remote	10
qacct	11
qacct_remote	11
qstat_j	11
qstat_j_remote	12
qstat_remote	12
qsub	12
qsub_lapply	13
qsub_retrieve	14
qsub_run	15
rm_remote	15
rsync_remote	16
run_remote	17
set_default_qsub_config	18
test_qsub_config	19
write_remote	19

Index	20
--------------	-----------

cat_remote	<i>Read from a file remotely</i>
------------	----------------------------------

Description

Read from a file remotely

Usage

```
cat_remote(path, remote = FALSE, verbose = FALSE)
```

Arguments

path	Path of the file.
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

cp_remote	<i>A wrapper around the scp shell command that handles local/remote files and allows copying between remote hosts via the local machine.</i>
-----------	--

Description

A wrapper around the scp shell command that handles local/remote files and allows copying between remote hosts via the local machine.

Usage

```
cp_remote(  
    remote_src,  
    path_src,  
    remote_dest,  
    path_dest,  
    verbose = FALSE,  
    recursively = FALSE  
)
```

Arguments

remote_src	Remote machine for the source file in the format user@machine or an empty string for local.
path_src	Path of the source file.
remote_dest	Remote machine for the destination file in the format user@machine or an empty string for local.
path_dest	Path for the source file; can be a directory.
verbose	Prints elapsed time if TRUE
recursively	Copy a directory recursively?

Examples

```
## Not run:  
## Copy file myfile.csv from the home directory on the remote server to  
## the local working directory.  
  
## on remote server in bash shell:  
# cat myfile.csv  
# [me@myserver ~]$ cat myfile.csv  
# "val","ts"  
# 1,  
# 2,  
# 3,  
# 4,  
# 5,  
# 6,
```

```

# 7,
# 8,
# 9,
# 10,

## on local server in R:
cp_remote(remote_src = "me@myserver", path_src = "~/myfile.csv",
          remote_dest = FALSE, path_dest = getwd(), verbose = TRUE)
# [1] "Elapsed: 1.672 sec"
df <- read.csv("myfile.csv")
df
#   val ts
# 1   1 NA
# 2   2 NA
# 3   3 NA
# 4   4 NA
# 5   5 NA
# 6   6 NA
# 7   7 NA
# 8   8 NA
# 9   9 NA
# 10 10 NA

## End(Not run)

```

create_qsub_config *Create a qsub configuration object.*

Description

Create a qsub configuration object.

Usage

```

create_qsub_config(
  remote,
  local_tmp_path,
  remote_tmp_path,
  name = "r2qsub",
  num_cores = 1,
  memory = "4G",
  max_running_tasks = NULL,
  max_wall_time = "01:00:00",
  batch_tasks = 1,
  compress = c("gz", "bz2", "xz", "none"),
  modules = "R",
  execute_before = NULL,
  verbose = FALSE,
  wait = TRUE,

```

```

    remove_tmp_folder = TRUE,
    stop_on_error = TRUE
)

override_qsub_config(
  qsub_config = get_default_qsub_config(),
  remote = qsub_config$remote,
  local_tmp_path = qsub_config$local_tmp_path,
  remote_tmp_path = qsub_config$remote_tmp_path,
  name = qsub_config$name,
  num_cores = qsub_config$num_cores,
  memory = qsub_config$memory,
  max_running_tasks = qsub_config$max_running_tasks,
  max_wall_time = qsub_config$max_wall_time,
  batch_tasks = qsub_config$batch_tasks,
  compress = qsub_config$compress,
  modules = qsub_config$modules,
  execute_before = qsub_config$execute_before,
  verbose = qsub_config$verbose,
  wait = qsub_config$wait,
  remove_tmp_folder = qsub_config$remove_tmp_folder,
  stop_on_error = qsub_config$stop_on_error
)

```

Arguments

remote	Remote machine specification for ssh, in format such as user@server:port that does not require interactive password entry.
local_tmp_path	A directory on the local machine in which to store temporary files. Should not contain a tilde ('~').
remote_tmp_path	A directory on the remote machine in which to store temporary files. Should not contain a tilde ('~').
name	The name of the execution. This will show up, for instance, in qstat.
num_cores	The number of cores to allocate per element in X in a <code>qsub_lapply</code> (default: 1).
memory	The memory to allocate per core (default: "4G"). If this is set too high without it being required, you might not be able to make optimal use of the remote cluster.
max_running_tasks	limit concurrent array job task execution (default: NULL, infinite). If you have long jobs and there are many other users on the cluster, it is recommended you set this value to a reasonable number, such as 1/4th the total number of nodes * number of cores per node.
max_wall_time	The maximum time each task is allowed to run (default: "01:00:00", 1 hour). If set to NULL, the job will be allowed to run indefinitely. Mind you, this might annoy other users of the cluster.
batch_tasks	How many values in X should be processed per task. Useful for when the 'length(X)' is very large (> 10000).

compress	Compression method to use: "none", "gz" (default), "bz2", or "xz".
modules	Which modules to load (default: "R"). If set to NULL, it will be assumed Rscript will be available in the path through other means.
execute_before	Commands to execute in the bash shell before running R.
verbose	Whether or not to print out any ssh commands.
wait	If TRUE, will wait until the execution has finished by periodically checking the job status.
remove_tmp_folder	If TRUE, will remove everything that was created related to this execution at the end.
stop_on_error	If TRUE, will stop when an error occurs, else returns a NA for errored instances.
qsub_config	A qsub_config to be overridden

Value

A qsub configuration object.

See Also

[qsub_lapply](#), [set_default_qsub_config](#)

Examples

```
## Not run:
qsub_config <- create_qsub_config(
  remote = "myuser@myserver.mylocation.com:22",
  local_tmp_path = "/home/myuser/workspace/.r2gridengine",
  remote_tmp_path = "/scratch/myuser/.r2gridengine"
)
qsub_lapply(1:10, function(x) x + 1, qsub_config = qsub_config)

set_default_qsub_config(qsub_config, permanent = TRUE)
qsub_lapply(1:10, function(x) x + 1)

qsub_lapply(
  X = 1:10,
  FUN = function(x) x + 1,
  qsub_config = override_qsub_config(verbose = TRUE)
)

## End(Not run)
```

create_ssh_connection *Create an SSH connection with remote*

Description

Create an SSH connection with remote

Usage

```
create_ssh_connection(remote)
```

Arguments

remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For the default qsub config remote, use TRUE.
--------	---

file_exists_remote *Checks if a local or remote file exists.*

Description

Checks if a local or remote file exists.

Usage

```
file_exists_remote(file, remote = FALSE, verbose = FALSE)
```

Arguments

file	File path.
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

Value

TRUE or FALSE indicating whether the file exists.

Examples

```
## Not run:  
file_exists_remote("~/myfile.csv", remote = "me@myserver")  
# [1] TRUE  
  
## End(Not run)
```

`get_default_qsub_config`*Retrieve a default qsub_config.*

Description

Will prefer the temporary default over the permanent default. You should typically not require this function.

Usage

```
get_default_qsub_config(config_file = config_file_location())
```

Arguments

`config_file` The file in which a permanent default config is stored.

`instantiate_qsub_config`*Create an instance of the qsub_config.*

Description

This function generates the paths for the temporary files.

Usage

```
instantiate_qsub_config(qsub_config)
```

Arguments

`qsub_config` A valid `qsub_config` object.
@export # you should typically not require to call this function manually.

is_job_running	<i>Check whether a job is running.</i>
----------------	--

Description

Check whether a job is running.

Usage

```
is_job_running(qsub_config)
```

Arguments

qsub_config	The qsub configuration of class qsub_configuration, as returned by any qsub execution
-------------	---

is_qsub_config	<i>Returns whether the passed object is a qsub_config object.</i>
----------------	---

Description

Returns whether the passed object is a qsub_config object.

Usage

```
is_qsub_config(object)
```

Arguments

object	The object to be tested
--------	-------------------------

is_remote_local	<i>Tests whether the remote is a local host or not.</i>
-----------------	---

Description

Tests whether the remote is a local host or not.

Usage

```
is_remote_local(remote)
```

Arguments

remote	A putative remote machine. This function will return true if remote == FALSE.
--------	---

ls_remote	<i>View the contents of a directory remotely</i>
-----------	--

Description

View the contents of a directory remotely

Usage

```
ls_remote(path, remote = FALSE, verbose = FALSE)
```

Arguments

path	Path of the directory.
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

mkdir_remote	<i>Creates a remote directory with the specified group ownership and permissions.</i>
--------------	---

Description

Creates a remote directory with the specified group ownership and permissions.

Usage

```
mkdir_remote(path, remote = FALSE, verbose = FALSE)
```

Arguments

path	Directory path. If using remote, this should be a full path or a path relative to the user's home directory.
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

qacct	<i>Run qacct on remote</i>
-------	----------------------------

Description

Run qacct on remote

Usage

qacct(qsub_config)

Arguments

qsub_config	The config
-------------	------------

qacct_remote	<i>Run qacct on remote</i>
--------------	----------------------------

Description

Run qacct on remote

Usage

qacct_remote(job_id, remote = FALSE)

Arguments

job_id	The job_id of the job
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.

qstat_j	<i>Run qstat on remote</i>
---------	----------------------------

Description

Run qstat on remote

Usage

qstat_j(qsub_config)

Arguments

qsub_config	The config
-------------	------------

qstat_j_remote	<i>Run qstat on remote</i>
----------------	----------------------------

Description

Run qstat on remote

Usage

```
qstat_j_remote(job_id, remote = FALSE)
```

Arguments

job_id	The job_id of the job
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.

qstat_remote	<i>Show the status of Grid Engine jobs and queues</i>
--------------	---

Description

Show the status of Grid Engine jobs and queues

Usage

```
qstat_remote(remote = NULL, verbose = FALSE)
```

Arguments

remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

qsub	<i>Running Commands Remotely on Gridengine Clusters</i>
------	---

Description

Run 'lapply()' calls in parallel by submitting them to 'gridengine' clusters using the 'qsub' command.

qsub_lapply

Apply a Function over a List or Vector on a gridengine system!

Description

Apply a Function over a List or Vector on a gridengine system!

Usage

```
qsub_lapply(
  X,
  FUN,
  object_envir = environment(FUN),
  qsub_config = NULL,
  qsub_environment = NULL,
  qsub_packages = NULL,
  ...
)
```

Arguments

X	A vector (atomic or list) or an expression object. Other objects (including classed objects) will be coerced by <code>base::as.list</code> .
FUN	The function to be applied to each element of X.
object_envir	The environment in which to go looking for the <code>qsub_environment</code> variables, if these are characters.
qsub_config	The configuration to use for this execution.
qsub_environment	NULL, a character vector or an environment. Specifies what data and functions will be uploaded to the server.
qsub_packages	The packages to be loaded on the cluster.
...	optional arguments to FUN.

See Also

[create_qsub_config](#), [set_default_qsub_config](#)

Examples

```
## Not run:
# Initial configuration and execution
qsub_config <- create_qsub_config(
  remote = "myserver",
  local_tmp_path = "/home/myuser/workspace/.r2gridengine",
  remote_tmp_path = "/scratch/myuser/.r2gridengine"
)
```

```

qsub_lapply(
  X = seq_len(3),
  FUN = function(i) { Sys.sleep(1); i+1 },
  qsub_config = qsub_config
)

# Setting a default configuration and short hand notation for execution
set_default_qsub_config(qsub_config, permanent = T)
qsub_lapply(seq_len(3), function(i) { Sys.sleep(1); i+1 })

# Overriding a default qsub_config
qsub_lapply(seq_len(3), function(i) i + 1,
  qsub_config = override_qsub_config(name = "MyJob"))

# Don't wait for results, get a handle instead and retrieve later.
handle <- qsub_lapply(seq_len(3), function(i) i + 1,
  qsub_config = override_qsub_config(wait = F))

# Wait until results have been generated on the remote

# Retrieve results
qsub_retrieve(handle)

## End(Not run)

```

qsub_retrieve	<i>Retrieve the results of a qsub execution.</i>
---------------	--

Description

Retrieve the results of a qsub execution.

Usage

```
qsub_retrieve(qsub_config, wait = TRUE, post_fun = NULL)
```

Arguments

qsub_config	The qsub configuration of class <code>qsub_configuration</code> , as returned by any qsub execution
wait	If TRUE, wait until the execution has finished in order to return the results, else returns NULL if execution is not finished.
post_fun	Apply a function to the output after execution. Interface: <code>function(index, output)</code>

qsub_run	<i>Run a Function on a gridengine system!</i>
----------	---

Description

Run a Function on a gridengine system!

Usage

```
qsub_run(FUN, qsub_config = NULL, qsub_environment = NULL, ...)
```

Arguments

FUN	the function to be executed.
qsub_config	The configuration to use for this execution.
qsub_environment	NULL, a character vector or an environment. Specifies what data and functions will be uploaded to the server.
...	optional arguments to FUN.

See Also

[create_qsub_config](#), [set_default_qsub_config](#)

rm_remote	<i>Remove a file or folder</i>
-----------	--------------------------------

Description

Remove a file or folder

Usage

```
rm_remote(path, remote, recursive = FALSE, force = FALSE, verbose = FALSE)
```

Arguments

path	Path of the file/folder
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
recursive	Whether to work recursively
force	Whether to force removal
verbose	If TRUE prints the command.

rsync_remote

*Rsync files between machines***Description**

A wrapper around the rsync shell command that allows copying between remote hosts via the local machine.

Usage

```
rsync_remote(
    remote_src,
    path_src,
    remote_dest,
    path_dest,
    compress = TRUE,
    delete = "no",
    exclude = NULL,
    verbose = FALSE
)
```

Arguments

remote_src	Remote machine for the source, see the section below 'Specifying a remote'.
path_src	Path of the source file.
remote_dest	Remote machine for the destination, see the section below 'Specifying a remote'.
path_dest	Path for the source file; can be a directory.
compress	Whether or not to compress the data being transferred.
delete	Whether or not to delete files at the target remote. Use "yes" to delete files at the remote.
exclude	A vector of files / regexs to be excluded.
verbose	Prints elapsed time if TRUE.

Specifying a remote

A remote can be specified in one of the following ways:

- A character vector in format user@ipaddress:port,
- The name of a Host in the ~/.ssh/config file,
- FALSE for the local machine,
- TRUE for the default remote specified as get_default_qsub_config()\$remote.

run_remote	run_remote - <i>Runs the command locally or remotely using ssh.</i>
------------	---

Description

In `run_remote` the remote commands are enclosed in wrappers that allow to capture output. By default `stderr` is redirected to `stdout`. If there's a genuine error, e.g., the remote command does not exist, the output is not captured. In this case, one can see the output by setting `intern` to `FALSE`. However, when the command is run but exits with non-zero code, `run_remote` intercepts the generated warning and saves the output.

Usage

```
run_remote(
  command,
  remote = FALSE,
  args = character(),
  verbose = FALSE,
  shell = FALSE
)
```

Arguments

<code>command</code>	Command to run. If run locally, quotes should be escaped once. If run remotely, quotes should be escaped twice.
<code>remote</code>	Remote machine specification for <code>ssh</code> , in format such as <code>user@server</code> that does not require interactive password entry. For local execution, pass <code>FALSE</code> (default). For execution on the default <code>qsub</code> config <code>remote</code> , use <code>TRUE</code> .
<code>args</code>	Character vector, arguments to the command.
<code>verbose</code>	If <code>TRUE</code> prints the command.
<code>shell</code>	Whether to execute the command in a shell

Details

The remote command will be put inside double quotes twice, so all quotes in `cmd` must be escaped twice: `\\`". However, if the command is not remote, i.e., `remote` is `NULL` or empty string, quotes should be escaped only once.

If the command itself redirects output, the `stderr_redirect` flag should be set to `FALSE`.

Value

A list with components:

- `status` The exit status of the process. If this is `NA`, then the process was killed and had no exit status.
- `stdout` The standard output of the command, in a character scalar.

- `stderr` The standard error of the command, in a character scalar.
- `elapsed_time` The number of seconds required before this function returned an output.

Warnings are really errors here so the error flag is set if there are warnings.

set_default_qsub_config

Set a default qsub_config.

Description

If permanent, the `qsub_config` will be written to the specified path. Otherwise, it will be saved in the current environment.

Usage

```
set_default_qsub_config(  
  qsub_config,  
  permanent = TRUE,  
  config_file = config_file_location()  
)
```

Arguments

<code>qsub_config</code>	The <code>qsub_config</code> to use as default.
<code>permanent</code>	Whether or not to make this the default <code>qsub_config</code> .
<code>config_file</code>	The location to which to save the permanent <code>qsub_config</code> .

See Also

[qsub_lapply](#), [create_qsub_config](#)

Examples

```
## Not run:  
qsub_config <- create_qsub_config(  
  remote = "myserver",  
  local_tmp_path = "/home/myuser/workspace/.r2gridengine",  
  remote_tmp_path = "/scratch/myuser/.r2gridengine"  
)  
set_default_qsub_config(qsub_config, permanent = T)  
qsub_lapply(1:10, function(x) x + 1)  
  
## End(Not run)
```

test_qsub_config	<i>Tests whether the passed object is a qsub_config object.</i>
------------------	---

Description

Tests whether the passed object is a qsub_config object.

Usage

```
test_qsub_config(object)
```

Arguments

object	The object to be tested
--------	-------------------------

write_remote	<i>Write to a file remotely</i>
--------------	---------------------------------

Description

Write to a file remotely

Usage

```
write_remote(x, path, remote = FALSE, verbose = FALSE)
```

Arguments

x	The text to write to the file.
path	Path of the file.
remote	Remote machine specification for ssh, in format such as user@server that does not require interactive password entry. For local execution, pass FALSE (default). For execution on the default qsub config remote, use TRUE.
verbose	If TRUE prints the command.

Index

cat_remote, 2
cp_remote, 3
create_qsub_config, 4, 13, 15, 18
create_ssh_connection, 7

file_exists_remote, 7

get_default_qsub_config, 8

instantiate_qsub_config, 8
is_job_running, 9
is_qsub_config, 9
is_remote_local, 9

ls_remote, 10

mkdir_remote, 10

override_qsub_config
 (create_qsub_config), 4

qacct, 11
qacct_remote, 11
qstat_j, 11
qstat_j_remote, 12
qstat_remote, 12
qsub, 12
qsub_lapply, 5, 6, 13, 18
qsub_retrieve, 14
qsub_run, 15

rm_remote, 15
rsync_remote, 16
run_remote, 17

set_default_qsub_config, 6, 13, 15, 18

test_qsub_config, 19

write_remote, 19