Package: NHSRwaitinglist (via r-universe)

October 24, 2024

Title R-package to implement a waiting list management approach

Version 0.0.0.9001

Maintainer Tom Smith <thomas.smith5@nuh.nhs.uk>

Description R-package to implement the waiting list management approach described in this paper by Fong et al 2022.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

VignetteBuilder knitr

URL https://nhs-r-community.github.io/NHSRwaitinglist/

Config/testthat/edition 3

Imports cli, dplyr, rlang, purrr, utils, stats

Suggests ggplot2, knitr, rmarkdown, testthat (>= 3.0.0)

Depends R (>= 2.10)

LazyData true

Repository https://nhs-r-community.r-universe.dev

RemoteUrl https://github.com/nhs-r-community/NHSRwaitinglist

RemoteRef main

RemoteSha 861cf52f64a70b43e8293c052cb325d711bf66a4

Contents

`calc_queue_load` *Calculate Queue Load*

Description

Calculates the queue load. The queue load is the number of arrivals that occur for every patient leaving the queue (given that the waiting list did not empty). It could also be described as the rate of service at the queue. The queue load is calculated by dividing the demand by the capacity: queue_load = demand / capacity.

Usage

```
calc_queue_load(demand, capacity)
```

Arguments

`demand`	Numeric value of rate of demand in same units as target wait - e.g. if target wait is weeks, then demand in units of patients/week.
`capacity`	Numeric value of the number of patients that can be served (removals) from the waiting list each week.

Value

Numeric value of load which is the ratio between demand and capacity.

Examples

```
# If 30 patients are added to the waiting list each week (demand) and 27
# removed (capacity) this results in a queue load of 1.11 (30/27).
calc_queue_load(30, 27)
```

calc_relief_capacity *Relief Capacity*

Description

Calculates required relief capacity to achieve target queue size in a given period of time as a function of demand, queue size, target queue size and time period. Relief Capacity is required if Queue Size > 2 * Target Queue Size.

Relief Capacity = Current Demand + (Queue Size - Target Queue Size)/Time Steps

WARNING!: make sure units match. I.e. if demand is measured per week then time_to_target should be weeks or if demand is per day then time_to_target is per day

Usage

```
calc_relief_capacity(
  demand,
  queue_size,
  target_queue_size,
  time_to_target = 26
)
```

Arguments

`demand` Numeric value of rate of demand in same units as target wait e.g. if target wait is weeks, then demand in units of patients/week.

`queue_size` Numeric value of current number of patients in queue.

`target_queue_size` Numeric value of desired number of patients in queue.

`time_to_target` Numeric value of desired number of time-steps to reach the target queue size by.

Value

A numeric value of the required rate of capacity to achieve a target queue size in a given period of time.

Examples

```
# If demand is 30 patients per week, the current queue size is 1200 and the
# target is to achieve a queue size of 390 in 26 weeks, then

# Relief Capacity = 30 + (1200 - 390)/26 = 61.15 patients per week.

calc_relief_capacity(30, 1200, 390, 26)
```

calc_target_capacity *Target Capacity*

Description

Applies Kingman/Marchal's Formula :

```
 capacity = demand + (cvd**2 + cvc**2) / waiting_time
```

where cvd = coefficent of variation of time between arrivals cvd = coefficient of variation of sevice times waiting_time = target_wait / factor

Usage

```
calc_target_capacity(
  demand,
  target_wait,
  factor = 4,
  cv_demand = 1,
  cv_capacity = 1
)
```

Arguments

demand	Numeric value of rate of demand in same units as target wait e.g. if target wait is weeks, then demand in units of patients/week.
target_wait	Numeric value of number of weeks that has been set as the target within which the patient should be seen.
factor	the amount we divide the target by in the waiting list e.g. if target is 52 weeks the mean wait should be 13 for a factor of 4
cv_demand	coefficient of variation of time between arrivals
cv_capacity	coefficient of variation between removals due to operations completed

Value

numeric. The capacity required to achieve a target waiting time.

Examples

```
demand <- 4 # weeks
target_wait <- 52 # weeks

# number of operations per week to have mean wait of 52/4
calc_target_capacity(demand, target_wait)

# TODO: Include a couple of standard deviations for errors in the mean demand
```

`calc_target_mean_wait` *Average Waiting Time*

Description

This calculates the target mean wait given the two inputs of target_wait and a numerical value for factor. The average wait is actually the target mean wait and is calculated as follows: target_wait / factor. If we want to have a chance between 1.8%-0.2% of making a waiting time target, then the average patient should have a waiting time between a quarter and a sixth of the target. Therefore: The mean wait should sit somewhere between target_wait/factor=6 < Average Waiting Time < target_wait/factor=4.

Usage

```
calc_target_mean_wait(target_wait, factor = 4)
```

Arguments

`target_wait` Numeric value of the number of weeks that has been set as the target within which the patient should be seen.

`factor` Numeric factor used in average wait calculation - to get a quarter of the target use factor=4 and one sixth of the target use factor = 6 etc. Defaults to 4.

Value

Numeric value of target mean waiting time to achieve a given target wait.

Examples

```
# If the target wait is 52 weeks then the target mean wait with a factor of 4
# would be 13 weeks and with a factor of 6 it would be 8.67 weeks.
calc_target_mean_wait(52, 4)
```

`calc_target_queue_size` *Target Queue Size*

Description

Uses Little's Law to calculate the target queue size to achieve a target waiting time as a function of observed demand, target wait and a variability factor used in the target mean waiting time calculation.

Target Queue Size = Demand * Target Wait / 4.

The average wait should sit somewhere between target_wait/factor=6 < Average Waiting Time < target_wait/factor=4 The factor defaults to 4.

Only applicable when Capacity > Demand.

Usage

```
calc_target_queue_size(demand, target_wait, factor = 4)
```

Arguments

`demand` Numeric value of rate of demand in same units as target wait e.g. if target wait is weeks, then demand in units of patients/week.

`target_wait` Numeric value of number of weeks that has been set as the target within which the patient should be seen.

`factor` Numeric factor used in average wait calculation

- to get a quarter of the target use factor=4
- to get one sixth of the target use factor = 6 etc. Defaults to 4.

Value

Numeric target queue length.

Examples

```
# If demand is 30 patients per week and the target wait is 52 weeks, then the
# Target queue size = 30 * 52/4 = 390 patients.

calc_target_queue_size(30, 52, 4)
```

calc_waiting_list_pressure *Calculate Waiting List Pressure*

Description

For a waiting list with target waiting time, the pressure on the waiting list is twice the mean delay divided by the waiting list target. The pressure of any given waiting list should be less than 1. If the pressure is greater than 1 then the waiting list is most likely going to miss its target. The waiting list pressure is calculated as follows: pressure = 2 * mean_wait / target_wait.

Usage

```
calc_waiting_list_pressure(mean_wait, target_wait)
```

Arguments

`mean_wait` Numeric value of target mean waiting time to achieve a given target wait.

`target_wait` Numeric value of the number of weeks that has been set as the target within which the patient should be seen.

Value

Numeric value of wait_pressure which is the waiting list pressure.

Examples

```
calc_waiting_list_pressure(63, 52)
```

`create_bulk_synthetic_data` *Create Bulk Synthetic Data*

Description

Creates a series of waiting lists, one for each row in the dataframe parameter and joins them together into one dataframe with relevant creation criteria

Usage

```
create_bulk_synthetic_data(bulk_data)
```

Arguments

`bulk_data`	A dataframe object, each row being a waiting list with parameters to generate the synthetic data. A sample data.frame is available as demo-data

Value

Dataframe of waiting lists for each specified site and specialty, opcs etc

Examples

```
create_bulk_synthetic_data(demo_df)
```

`create_waiting_list` *Create Waiting List*

Description

Creates a waiting list using the parameters specified

Usage

```
create_waiting_list(
  n,
  mean_arrival_rate,
  mean_wait,
  start_date = Sys.Date(),
  limit_removals = TRUE,
  sd = 0,
  rott = 0,
  ...
)
```

Arguments

`n` Numeric value of rate of demand in same units as target wait

- e.g. if target wait is weeks, then demand in units of patients/week.

`mean_arrival_rate` Numeric value of mean daily arrival rate.

`mean_wait` Numeric value of mean wait time for treatment/on waiting list.

`start_date` Character value of date from which to start generated waiting list.

`limit_removals` Defaults to TRUE

`sd` Numeric value, standard deviation. Defaults to 0.

`rott` Numeric value, proportion of referrals to be randomly flagged as ROTT. Defaults to 0.

`...` Container for the list

Value

A tibble of a random generated list of patients with addition_date, removal_date, wait_length and rott status for each patient

Examples

```
create_waiting_list(366,50,21,”2024-01-01”,10,0.1)
```

demo_df *A Demo ’data.frame’ Object*

Description

A pre-created data.frame ready to be used to test the create_bulk_synthetic_data and create_waiting_list functions. Each row of the data.frame represents an individual waiting list for which the site, specialty, OPCS code(s) and respective mean wait, arrival rate, start_date, sd and rott can be specified. It allows the user to see an example of the structure of the data.frame required by the create_bulk_synthetic_data function to create your synthetic dataset.

Usage

```
demo_df
```

Format

A dataframe with 5 rows and 9 columns:

hospital_site Character. Hospital site code of the waiting list.
main_spec_code Numeric. Main specialty code of the waiting list.
opcs4_code Character. OPCS4 code(s) of the procedure(s) on the waiting list.
n Numeric. Number of days for which to create synthetic waiting list data.
mean_arrival_rate Numeric. Mean number of arrivals per day.
mean_wait Numeric. Mean wait time for treatment/on waiting list.
start_date Character. Date from which to start generated waiting list in format yyyy-mm-dd.
sd Numeric. Standard deviation.
rott Numeric. Proportion of referrals to be randomly flagged as ROTT

`wl_insert` *Insert new referrals into the waiting list*

Description

adds new referrals (removal date is set as NA)

Usage

```
wl_insert(waiting_list, additions, referral_index = 1)
```

Arguments

`waiting_list` dataframe. A df of referral dates and removals
`additions` character vector. A list of referral dates to add to the waiting list
`referral_index` integer. The column number in the waiting_list which contains the referral dates

Value

dataframe. A df of the updated waiting list

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
additions <- c.Date("2024-01-03", "2024-01-05", "2024-01-18")
longer_waiting_list <- wl_insert(waiting_list, additions)

# TODO: What if more columns
# Check column types
```

wl_join *Join two waiting list*

Description

Take two waiting list and sorting in date order

Usage

```
wl_join(wl_1, wl_2, referral_index = 1)
```

Arguments

`wl_1`	a waiting list: dataframe consisting addition and removal dates
`wl_2`	a waiting list: dataframe consisting addition and removal dates
`referral_index`	the column index where referrals are listed

Value

updated_list a new waiting list

Examples

```
# referrals <- c.Date("2024-01-01","2024-01-04","2024-01-10","2024-01-16")
# removals <- c.Date("2024-01-08",NA,NA,NA)
# wl_1 <- data.frame("referral" = referrals ,"removal" = removals )

# referrals <- c.Date("2024-01-04","2024-01-05","2024-01-16","2024-01-25")
# removals <- c.Date("2024-01-09",NA,"2024-01-19",NA)
# wl_2 <- data.frame("referral" = referrals ,"removal" = removals )
# wl_join(wl_1,wl_2)
```

wl_queue_size *Queue size calculator*

Description

Calculates queue sizes from a waiting list

Usage

```
wl_queue_size(waiting_list, start_date = NULL, end_date = NULL)
```

Arguments

`waiting_list`	dataframe consisting addition and removal dates
`start_date`	start of calculation period
`end_date`	end of calculation period

Value

a list of dates and queue sizes

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
wl_queue_size(waiting_list)
```

wl_referral_stats *Calculate some stats about referrals*

Description

Calculate some stats about referrals

Usage

```
wl_referral_stats(waiting_list, start_date = NULL, end_date = NULL)
```

Arguments

`waiting_list`	dataframe. A df of referral dates and removals
`start_date`	date. The start date to calculate from
`end_date`	date. The end date to calculate to

Value

dataframe. A df containing number of referrals, mean demand, and the coefficient of variation of referrals

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
referral_stats <- wl_referral_stats(waiting_list)

# TODO : referral <- arrival
# debug and test
# simplify notation
# add detail to params above
# arrival mean and variance
```

wl_removal_stats *Calculate some stats about removals*

Description

Calculate some stats about removals

Usage

```
wl_removal_stats(waiting_list, start_date = NULL, end_date = NULL)
```

Arguments

waiting_list	dataframe. A df of referral dates and removals
start_date	date. The start date to calculate from
end_date	date. The end date to calculate to

Value

dataframe. A df containing number of removals, mean capacity, and the coefficient of variation of removals

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
removal_stats <- wl_removal_stats(waiting_list)

# TODO : referral <- arrival
# debug and test
# simplify notation
# add detail to params above
# arrival mean and variance
```

wl_schedule *A simple operation scheduler*

Description

Takes a list of dates and schedules them to a waiting list, by adding a removal date to the dataframe. This is done in referral date order, I.e. earlier referrals are scheduled first (FIFO).

Usage

```
wl_schedule(waiting_list, schedule, referral_index = 1, removal_index = 2)
```

Arguments

`waiting_list`	dataframe. A df of referral dates and removals
`schedule`	vector of dates. The dates to schedule open referrals into (ie. dates of unbooked future capacity)
`referral_index`	integer. The column number in the waiting_list which contains the referral dates
`removal_index`	integer. The column number in the waiting_list which contains the removal dates

Value

dataframe. A df of the updated waiting list with removal dates added according to the schedule

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
schedule <- c.Date("2024-01-03", "2024-01-05", "2024-01-18")
updated_waiting_list <- wl_schedule(waiting_list, schedule)

# TODO ALLOW:
# schedule to be a dataframe or vector
```

wl_simulator *Simple simulator to create a waiting list*

Description

Creates a simulated waiting list comprising referral dates, and removal dates

Usage

```
wl_simulator(
  start_date,
  end_date,
  demand,
  capacity,
  waiting_list = NULL,
  referral_index = 1
)
```

Arguments

`start_date` date. The start date for the simulation

`end_date` date. The end date for the simulation

`demand` numeric. Weekly demand (ie. typical referrals per week)

`capacity` numeric. Weekly capacity (ie. typical removals per week)

`waiting_list` integer. The number of patients on the waiting list

`referral_index` integer. The column number in the waiting_list which contains the referral dates

Value

dataframe. A df of simulated referrals and removals

Examples

```
over_capacity_simulation <-
  wl_simulator("2024-01-01", "2024-03-31", 100, 110)
under_capacity_simulation <-
  wl_simulator("2024-01-01", "2024-03-31", 100, 90)

# TODO
# error messages (e.g. start_date > end_date)
```

`wl_stats` *Calculate some stats about the waiting list*

Description

A summary of all the key stats associated with a waiting list

Usage

```
wl_stats(waiting_list, target_wait = 4, start_date = NULL, end_date = NULL)
```

Arguments

`waiting_list` dataframe. A df of referral dates and removals

`target_wait` numeric. The required waiting time

`start_date` date. The start date to calculate from

`end_date` date. The end date to calculate to

Value

dataframe. A df of important waiting list statistics

Examples

```
referrals <- c.Date("2024-01-01", "2024-01-04", "2024-01-10", "2024-01-16")
removals <- c.Date("2024-01-08", NA, NA, NA)
waiting_list <- data.frame("referral" = referrals, "removal" = removals)
waiting_list_stats <- wl_stats(waiting_list)

# TO DO!!
# Check target_capacity weekly daily
# Error if dates are in the wrong order
# Calculate the number of missed operations
# Start date and end date calculations not working well
#
# MAKE CONSISTENT NOTATION
# default start and end date if empty
# make units of output weekly operations not daily
# mean removal too big
# Z score in capacity calculations
# mean demand and mean capacity not interrarraival and departures plesae.
```

Index