Package 'pct'

March 5, 2025

Type Package

Title Propensity to Cycle Tool

Version 0.10.0

Description Functions and example data to teach and
increase the reproducibility of the methods and code underlying
the Propensity to Cycle Tool (PCT), a research project and web application
hosted at <https://www.pct.bike/>.
For an academic paper on the methods,
see Lovelace et al (2017) <doi:10.5198/jtlu.2016.862>.

Depends R (>= 3.5.0)

License GPL-3

URL https://itsleeds.github.io/pct/, https://github.com/ITSLeeds/pct

BugReports https://github.com/ITSLeeds/pct/issues

Encoding UTF-8

LazyData true

Imports boot, stplanr (>= 0.2.8), readr, sf, crul

Suggests covr, curl, dplyr, ggplot2, knitr, leaflet, pbapply, remotes,
rmarkdown, tmap, bookdown

VignetteBuilder knitr

RoxygenNote 7.2.3

Language en-GB

NeedsCompilation no

Author Robin Lovelace [aut, cre] (<https://orcid.org/0000-0001-5679-6536>),
Layik Hama [aut] (<https://orcid.org/0000-0003-1912-4890>),
Nathanael Sheehan [ctb] (<https://orcid.org/0000-0002-2779-0976>)

Maintainer Robin Lovelace <rob00x@gmail.com>

Repository CRAN

Date/Publication 2025-03-05 08:40:07 UTC

Contents

`desire_lines_leeds` *Cycle route desire lines for Leeds*

Description

Cycle route desire lines for Leeds

Examples

```
# see data-raw folder for generation code
desire_lines_leeds
```

get_centroids_ew *Download MSOA centroids for England and Wales*

Description

Downloads and processes data on where people live in England and Wales. See geoportal.statistics.gov.uk.

Usage

```
get_centroids_ew()
```

get_desire_lines *Desire lines*

Description

This function generates "desire lines" from census 2011 data. By default gets all desire lines from census in region, but can get the top n.

Usage

```
get_desire_lines(region = NULL, n = NULL, omit_intrazonal = FALSE)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
n	top n number of destinations with most trips in the 2011 census within the region.
omit_intrazonal	should intrazonal OD pairs be omited from result? FALSE by default.

Examples

```
if(curl::has_internet()) {
desire_lines = get_desire_lines("wight")
plot(desire_lines)
intra_zonal = desire_lines$geo_code1 == desire_lines$geo_code2
plot(desire_lines[intra_zonal, ])
}
```

get_od *Get origin destination data from the 2011 Census*

Description

This function downloads a .csv file representing movement between MSOA zones in England and Wales. By default it returns national data, but `region` can be set to subset the output to a specific local authority or region.

Usage

```
get_od(
  region = NULL,
  n = NULL,
  type = "within",
  omit_intrazonal = FALSE,
  base_url = paste0("https://s3-eu-west-1.amazonaws.com/",
    "statistics.digitalresources.jisc.ac.uk", "/dkan/files/FLOW/"),
  filename = "wu03ew_v2",
  u = NULL
)
```

Arguments

`region`	The PCT region or local authority to download data from (e.g. `west-yorkshire` or `Leeds`). See `View(pct_regions_lookup)` for a full list of possible region names.
`n`	top n number of destinations with most trips in the 2011 census within the `region`.
`type`	the type of subsetting: one of `from`, `to` or `within`, specifying how the od dataset should be subset in relation to the `region`.
`omit_intrazonal`	should intrazonal OD pairs be omited from result? `FALSE` by default.
`base_url`	the base url where the OD dataset is stored
`filename`	the name of the file to download, if not the default MSOA level data.
`u`	full url of file to download

Details

OD datasets available include wu03uk_v3 and others listed on the Wicid website.

Examples

```
get_od("wight", n = 3)
get_od()
get_od(filename = "wu03uk_v3")
u = "https://www.nomisweb.co.uk/output/census/2011/wf02ew_oa.zip"
# get_od(u = u)
```

---

get_pct *Generic function to get regional data from the PCT*

Description

This function gets data generated for the Propensity to Cycle Tool project and returns objects in the modern `sf` class.

Usage

```
get_pct(
  base_url = "https://github.com/npct/pct-outputs-regional-notR/raw/master",
  purpose = "commute",
  geography = "lsoa",
  region = NULL,
  layer = NULL,
  extension = ".geojson",
  national = FALSE
)
```

Arguments

`base_url`	Where the data is stored.
`purpose`	Trip purpose (typically `school` or `commute`)
`geography`	Geographic resolution of outputs, `msoa` or `lsoa` (the default)
`region`	The PCT region or local authority to download data from (e.g. `west-yorkshire` or `Leeds`). See `View(pct_regions_lookup)` for a full list of possible region names.
`layer`	The PCT layer of interest, `z`, `c`, `l`, `rf`, `rq` or `rnet` for zones, centroids, desire lines, routes (fast or quiet) and route networks, respectively
`extension`	The type of file to download (only `.geojson` supported at present)
`national`	Download nationwide data? `FALSE` by default

Examples

```
## Not run:
rf = get_pct(region = "isle-of-wight", layer = "rf")
names(rf)[1:20]
vars_to_plot = 10:13
plot(rf[vars_to_plot])
z = get_pct(region = "isle-of-wight", layer = "z")
rf = get_pct(region = "west-yorkshire", layer = "rf")
z_all = get_pct(layer = "z", national = TRUE)

## End(Not run)
```

---

get_pct_centroids *Get centroid results from the PCT*

Description

Wrapper around [get_pct()] that gets centroid data from the PCT.

Usage

```
get_pct_centroids(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
purpose	Trip purpose (typically school or commute)
geography	Geographic resolution of outputs, msoa or lsoa (the default)
extension	The type of file to download (only .geojson supported at present)

Examples

```
## Not run:
# don't test to reduce build times
c = get_pct_centroids("isle-of-wight")
plot(c)

## End(Not run)
```

get_pct_lines *Get desire lines results from the PCT*

Description

Wrapper around [get_pct()] that gets l (lines) data from the PCT.

Usage

```
get_pct_lines(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
purpose	Trip purpose (typically school or commute)
geography	Geographic resolution of outputs, msoa or lsoa (the default)
extension	The type of file to download (only .geojson supported at present)

Examples

```
## Not run:
# don't test to reduce build times
l = get_pct_lines("isle-of-wight")
plot(l)

## End(Not run)
```

get_pct_rnet *Get route network results from the PCT*

Description

Wrapper around [get_pct()] that gets route road network data from the PCT.

Usage

```
get_pct_rnet(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
purpose	Trip purpose (typically school or commute)
geography	Geographic resolution of outputs, msoa or lsoa (the default)
extension	The type of file to download (only .geojson supported at present)

Examples

```
## Not run:
# don't test to reduce build times
rnet =  get_pct_rnet("isle-of-wight")
plot(rnet)

## End(Not run)
```

---

get_pct_routes_fast *Get fast road network results from the PCT*

Description

Wrapper around [get_pct()] that gets rf data from the PCT.

Usage

```
get_pct_routes_fast(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
purpose	Trip purpose (typically school or commute)
geography	Geographic resolution of outputs, msoa or lsoa (the default)
extension	The type of file to download (only .geojson supported at present)

Examples

```
## Not run:
# don't test to reduce build times
rf = get_pct_routes_fast("isle-of-wight")
plot(rf)

## End(Not run)
```

---

get_pct_routes_quiet *Get quiet road network results from the PCT*

Description

Wrapper around [get_pct()] that gets rq data from the PCT.

Usage

```
get_pct_routes_quiet(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

region	The PCT region or local authority to download data from (e.g. west-yorkshire or Leeds). See View(pct_regions_lookup) for a full list of possible region names.
purpose	Trip purpose (typically school or commute)
geography	Geographic resolution of outputs, msoa or lsoa (the default)
extension	The type of file to download (only .geojson supported at present)

Examples

```
## Not run:
# don't test to reduce build times
rq =  get_pct_routes_quiet("isle-of-wight")
plot(rq)

## End(Not run)
```

`get_pct_zones` *Get zone results from the PCT*

Description

Wrapper around `[get_pct()]` that gets zone data from the PCT.

Usage

```
get_pct_zones(
  region = NULL,
  purpose = "commute",
  geography = "lsoa",
  extension = ".geojson"
)
```

Arguments

`region`	The PCT region or local authority to download data from (e.g. `west-yorkshire` or `Leeds`). See `View(pct_regions_lookup)` for a full list of possible region names.
`purpose`	Trip purpose (typically `school` or `commute`)
`geography`	Geographic resolution of outputs, `msoa` or `lsoa` (the default)
`extension`	The type of file to download (only `.geojson` supported at present)

Examples

```
## Not run:
# don't test to reduce build times
z = get_pct_zones("isle-of-wight")
plot(z)

## End(Not run)
```

leeds_uber_sample *Top 15 min mean journy times within Leeds from Uber*

Description

Data downloaded 4th March 2019. According to Uber, the dataset is from: 1/1/2018 - 1/31/2018 (Every day, Daily Average)

Examples

```
# see data-raw folder for generation code
leeds_uber_sample
```

model_pcycle_pct_2020 *Model cycling levels as a function of explanatory variables*

Description

Model cycling levels as a function of explanatory variables

Usage

```
model_pcycle_pct_2020(pcycle, distance, gradient, weights)
```

Arguments

pcycle	The proportion of trips by bike, e.g. 0.1, meaning 10%
distance	Vector distance numeric values of routes in km (switches to km if more than 100).
gradient	Vector gradient numeric values of routes.
weights	The weights used in the model, typically the total number of people per OD pair

Examples

```
# l = get_pct_lines(region = "isle-of-wight")
# l = get_pct_lines(region = "cambridgeshire")
l = wight_lines_pct
pcycle = l$bicycle / l$all
pcycle_dutch = l$dutch_slc / l$all
m1 = model_pcycle_pct_2020(
  pcycle,
  distance = l$rf_dist_km,
  gradient = l$rf_avslope_perc - 0.78,
  weights = l$all
  )
m2 = model_pcycle_pct_2020(
```

```
  pcycle_dutch, distance = l$rf_dist_km,
  gradient = l$rf_avslope_perc - 0.78,
  weights = l$all
)
m3 = model_pcycle_pct_2020(
  pcycle_dutch, distance = l$rf_dist_km,
  gradient = l$rf_avslope_perc - 0.78,
  weights = rep(1, nrow(l))
)
m1
plot(l$rf_dist_km, pcycle, cex = l$all / 100, ylim = c(0, 0.5))
points(l$rf_dist_km, m1$fitted.values, col = "red")
points(l$rf_dist_km, m2$fitted.values, col = "blue")
points(l$rf_dist_km, pcycle_dutch, col = "green")
cor(l$dutch_slc, m2$fitted.values * l$all)^2 # 95% captured
# identical means:
mean(l$dutch_slc)
mean(m2$fitted.values * l$all)
pct_coefficients_2020 = c(
  alpha = -4.018 + 2.550,
  d1 = -0.6369 -0.08036,
  d2 = 1.988,
  d3 = 0.008775,
  h1 = -0.2555,
  i1 = 0.02006,
  i2 = -0.1234
)
pct_coefficients_2020
m2$coef
plot(pct_coefficients_2020, m2$coeff)
cor(pct_coefficients_2020, m2$coeff)^2
cor(pct_coefficients_2020, m3$coeff)^2 # explains 95%+ variability in params
```

`mode_names` *Mode names in the Census*

Description

And conversion into R-friendly versions

Examples

```
mode_names
```

od_leeds *Example OD data for Leeds*

Description

od_leeds contains the 100 most travelled work desire lines in Leeds, according to the 2011 Census.

Examples

```
# see data-raw folder for generation code
od_leeds
```

pct_regions *PCT regions from www.pct.bike*

Description

See data-raw folder for generation code

Examples

```
pct_regions
```

pct_regions_lookup *Lookup table matching PCT regions to local authorities*

Description

For matching pct_regions object with local authority names in England and Wales.

Examples

```
names(pct_regions_lookup)
head(pct_regions_lookup)
```

rnet_leeds *Route network for Leeds*

Description

Route network for Leeds

Examples

```
# see data-raw folder for generation code
rnet_leeds
```

routes_fast_leeds *Fastest cycle routes for the desire_lines_leeds*

Description

Fastest cycle routes for the desire_lines_leeds

Examples

```
# see data-raw folder for generation code
routes_fast_leeds
```

santiago_lines *Desire lines in central Santiago*

Description

See https://github.com/pedalea/pctSantiago folder for generation code

Examples

```
# u = "https://github.com/pedalea/pctSantiago/releases/download/0.0.1/od_agg_zone_sub.Rds"
# download.file(u, destfile = "od_agg_zone_sub.Rds")
# desire_lines = readRDS("od_agg_zone_sub.Rds")
santiago_zones
```

santiago_od *OD data in central Santiago*

Description

See https://github.com/pedalea/pctSantiago folder for generation code

Examples

```
# u = "https://github.com/pedalea/pctSantiago/releases/download/0.0.1/santiago_od.Rds"
# download.file(u, destfile = "santiago_od.Rds", mode = "wb")
# santiago_od = readRDS("santiago_od.Rds")
santiago_od
```

santiago_routes_cs *200 cycle routes in central Santiago, Chile*

Description

This data was obtained using code shown in the International application of the PCT methods vignette.

Examples

```
library(sf)
names(santiago_routes_cs)
head(santiago_routes_cs)
plot(santiago_routes_cs)
```

santiago_zones *Zones in central Santiago*

Description

See https://github.com/pedalea/pctSantiago folder for generation code

Examples

```
# u = "https://github.com/pedalea/pctSantiago/releases/download/0.0.1/z_centre.Rds"
# download.file(u, destfile = "z_centre.Rds", mode = "wb")
# santiago_zones = readRDS("z_centre.Rds")
santiago_zones
```

uptake_pct_godutch *Calculate cycling uptake for UK 'Go Dutch' scenario*

Description

This function implements the uptake model described in the original Propensity to Cycle Tool paper (Lovelace et al. 2017): https://doi.org/10.5198/jtlu.2016.862

Usage

```
uptake_pct_godutch(
  distance,
  gradient,
  alpha = -3.959 + 2.523,
  d1 = -0.5963 - 0.07626,
  d2 = 1.866,
  d3 = 0.00805,
  h1 = -0.271,
  i1 = 0.009394,
  i2 = -0.05135,
  verbose = FALSE
)
```

Arguments

distance	Vector distance numeric values of routes in km (switches to km if more than 100).
gradient	Vector gradient numeric values of routes.
alpha	The intercept
d1	Distance term 1
d2	Distance term 2
d3	Distance term 3
h1	Hilliness term 1
i1	Distance-hilliness interaction term 1
i2	Distance-hilliness interaction term 2
verbose	Print messages? FALSE by default.

Details

See uptake_pct_govtarget().

Examples

```
# https://www.jtlu.org/index.php/jtlu/article/download/862/1381/4359
# Equation 1B:
distance = 15
gradient = 2
logit = -3.959 + 2.523 +
  ((-0.5963 - 0.07626) * distance) +
  (1.866 * sqrt(distance)) +
  (0.008050 * distance^2) +
  (-0.2710 * gradient) +
  (0.009394 * distance * gradient) +
  (-0.05135 * sqrt(distance) * gradient)
logit
# Result: -3.144098
```

```
pcycle = exp(logit) / (1 + exp(logit))
# Result: 0.04132445
boot::inv.logit(logit)
uptake_pct_godutch(distance, gradient,
  alpha = -3.959 + 2.523, d1 = -0.5963 - 0.07626,
  d2 = 1.866, d3 = 0.008050, h1 = -0.2710, i1 = 0.009394, i2 = -0.05135
)
# these are the default values
uptake_pct_godutch(distance, gradient)
l = routes_fast_leeds
pcycle_scenario = uptake_pct_godutch(l$length, l$av_incline)
plot(l$length, pcycle_scenario)
```

`uptake_pct_govtarget` *Calculate cycling uptake for UK 'Government Target' scenario*

Description

Uptake model that takes distance and hilliness and returns a percentage of trips that could be made by cycling along a desire line under scenarios of change. Source: appendix of pct paper, hosted at: www.jtlu.org which states that: "To estimate cycling potential,the Propensity to Cycle Tool (PCT) was designed to use the best available geographically disaggregated data sources on travel patterns."

Usage

```
uptake_pct_govtarget(
  distance,
  gradient,
  alpha = -3.959,
  d1 = -0.5963,
  d2 = 1.866,
  d3 = 0.00805,
  h1 = -0.271,
  i1 = 0.009394,
  i2 = -0.05135,
  verbose = FALSE
)

uptake_pct_govtarget_2020(
  distance,
  gradient,
  alpha = -4.018,
  d1 = -0.6369,
  d2 = 1.988,
  d3 = 0.008775,
  h1 = -0.2555,
  h2 = -0.78,
```

```
  i1 = 0.02006,
  i2 = -0.1234,
  verbose = FALSE
)

uptake_pct_godutch_2020(
  distance,
  gradient,
  alpha = -4.018 + 2.55,
  d1 = -0.6369 - 0.08036,
  d2 = 1.988,
  d3 = 0.008775,
  h1 = -0.2555,
  h2 = -0.78,
  i1 = 0.02006,
  i2 = -0.1234,
  verbose = FALSE
)

uptake_pct_ebike_2020(
  distance,
  gradient,
  alpha = -4.018 + 2.55,
  d1 = -0.6369 - 0.08036 + 0.05509,
  d2 = 1.988,
  d3 = 0.008775 - 0.000295,
  h1 = -0.2555 + 0.1812,
  h2 = -0.78,
  i1 = 0.02006,
  i2 = -0.1234,
  verbose = FALSE
)

uptake_pct_govtarget_school2(
  distance,
  gradient,
  alpha = -7.178,
  d1 = -1.87,
  d2 = 5.961,
  h1 = -0.529,
  h2 = -0.63,
  verbose = FALSE
)

uptake_pct_godutch_school2(
  distance,
  gradient,
  alpha = -7.178 + 3.574,
```

```
  d1 = -1.87 + 0.3438,
  d2 = 5.961,
  h1 = -0.529,
  h2 = -0.63,
  verbose = FALSE
)
```

Arguments

distance	Vector distance numeric values of routes in km (switches to km if more than 100).
gradient	Vector gradient numeric values of routes.
alpha	The intercept
d1	Distance term 1
d2	Distance term 2
d3	Distance term 3
h1	Hilliness term 1
i1	Distance-hilliness interaction term 1
i2	Distance-hilliness interaction term 2
verbose	Print messages? FALSE by default.
h2	Hilliness term 2

Details

The functional form of the cycling uptake model used in the PCT is as follows: (Source: npct.github.io)

```
logit (pcycle) = -3.959 +   # alpha
  (-0.5963 * distance) +    # d1
  (1.866 * distancesqrt) +  # d2
  (0.008050 * distancesq) + # d3
  (-0.2710 * gradient) +    # h1
  (0.009394 * distance * gradient) +  # i1
  (-0.05135 * distancesqrt *gradient) # i2

pcycle = exp ([logit (pcycle)]) / (1 + (exp([logit(pcycle)])
```

`uptake_pct_govtarget_2020()` and `uptake_pct_godutch_2020()` approximate the uptake models used in the updated 2020 release of the PCT results.

If the `distance` parameter is greater than 100, it is assumed that it is in m. If for some reason you want to model cycling uptake associated with trips with distances of less than 100 m, convert the distances to km first.

Examples

```
distance = 15
gradient = 2
logit_pcycle = -3.959 + # alpha
  (-0.5963 * distance) + # d1
  (1.866 * sqrt(distance)) + # d2
  (0.008050 * distance^2) + # d3
  (-0.2710 * gradient) + # h1
  (0.009394 * distance * gradient) + # i1
  (-0.05135 * sqrt(distance) * gradient) # i2
boot::inv.logit(logit_pcycle)
uptake_pct_govtarget(15, 2)
l = routes_fast_leeds
pcycle_scenario = uptake_pct_govtarget(l$length, l$av_incline)
pcycle_scenario_2020 = uptake_pct_govtarget_2020(l$length, l$av_incline)
plot(l$length, pcycle_scenario, ylim = c(0, 0.2))
points(l$length, pcycle_scenario_2020, col = "blue")

# compare with published PCT data:
## Not run:
l_pct_2020 = get_pct_lines(region = "isle-of-wight")
# test for another region:
# l_pct_2020 = get_pct_lines(region = "west-yorkshire")
l_pct_2020$rf_avslope_perc[1:5]
l_pct_2020$rf_dist_km[1:5]
govtarget_slc = uptake_pct_govtarget(
  distance = l_pct_2020$rf_dist_km,
  gradient = l_pct_2020$rf_avslope_perc
) * l_pct_2020$all + l_pct_2020$bicycle
govtarget_slc_2020 = uptake_pct_govtarget_2020(
  distance = l_pct_2020$rf_dist_km,
  gradient = l_pct_2020$rf_avslope_perc
) * l_pct_2020$all + l_pct_2020$bicycle
mean(l_pct_2020$govtarget_slc)
mean(govtarget_slc)
mean(govtarget_slc_2020)
godutch_slc = uptake_pct_godutch(
  distance = l_pct_2020$rf_dist_km,
  gradient = l_pct_2020$rf_avslope_perc
) * l_pct_2020$all + l_pct_2020$bicycle
godutch_slc_2020 = uptake_pct_godutch_2020(
  distance = l_pct_2020$rf_dist_km,
  gradient = l_pct_2020$rf_avslope_perc
) * l_pct_2020$all + l_pct_2020$bicycle
mean(l_pct_2020$dutch_slc)
mean(godutch_slc)
mean(godutch_slc_2020)

## End(Not run)
# Take an origin destination (OD) pair between an LSOA centroid and a
# secondary school. In this OD pair, 30 secondary school children travel, of
# whom 3 currently cycle. The fastest route distance is 3.51 km and the
```

```
# gradient is 1.11%. The
# gradient as centred on Dutch hilliness levels is 1.11 - 0.63 = 0.48%.
# The observed number of cyclists is 2. ... Modelled baseline= 30 * .0558 = 1.8.
uptake_pct_govtarget_school2(3.51, 1.11)
# pcycle = exp ([logit (pcycle)])/(1 + (exp([logit(pcycle)]))).
# pcycle = exp(1.953)/(1 + exp(1.953)) = .8758, or 87.58%.
uptake_pct_godutch_school2(3.51, 1.11)
```

`wight_lines_30` *Desire lines from the PCT for the Isle of Wight*

Description

This data was obtained using code shown in the introductory pct package vignette.

Examples

```
names(wight_lines_30)
plot(wight_lines_30)
```

`wight_od` *Official origin-destination data for the Isle of Wight*

Description

This data was obtained using code shown in the introductory pct package vignette.

Examples

```
names(wight_od)
head(wight_od)
```

`wight_routes_30` *Cycle route data for the Isle of Wight*

Description

This data was obtained using code shown in the introductory pct package vignette.

Examples

```
library(sf)
names(wight_routes_30)
head(wight_routes_30)
plot(wight_routes_30)
```

wight_zones *Zones and centroid data from the PCT for the Isle of Wight*

Description

This data was obtained using code shown in the introductory pct package vignette.

Examples

```
library(sf)
names(wight_lines_30)
plot(wight_lines_30)
```

zones_leeds *Zone data for Leeds*

Description

Zones in Leeds

Examples

```
# see data-raw folder for generation code
zones_leeds
```

Index