Package ‘CSLSscenarios’

April 2, 2021

Type Package

Title CSLS MODFLOW Scenario Analysis

Version 1.0.0

Author Carolyn Voter

Maintainer Carolyn Voter <cvoter@wisc.edu>

Description Analyze MODFLOW scenarios for average seasonal water levels and significant (ecological) impact.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

Imports CSLSdata,
dplyr,
knitr,
lubridate,
magrittr,
raster,
reshape2,
rlang,
stats,
stringr,
NISTunits

Suggests DT,
extrafont,
ggplot2,
patchwork,
rmarkdown,
shiny

VignetteBuilder knitr

RoxygenNote 7.1.1

Roxygen list(markdown = TRUE)

R topics documented:

`calculate_bathymetry` *Calculate bathymetric-related metrics*

Description

Given a data frame with the metrics for a single set of simulations, calculate the associated volume (m3), area (m2), maximum depth (m), and mean depth (m) for each exceedance_level (m). Uses CSLSdata::bathymetry and CSLSdata::lake_raster, only performs calculations on lakes which have information in all three data sources.

Usage

```
calculate_bathymetry(df)
```

Arguments

`df`	a data frame with a "lake", "metric", "variable", and "value" columns where one of the metrics is "exceedance_level".

Value

bathymetry, a data frame with "lake", "metric", "variable", and "value" where the metrics are "volume", "area", "max_depth", and "mean_depth" and the variables are all the exceedance level probabilities inputted in df.

`calculate_docks` *Calculates number of summers with good/bad conditions for docks*

Description

Given a data frame with columns for "lake", "level", "month", and "year" as well as a data frame with additional information about the average dock length ("length_ft") and minimum desired depth at the end of the dock ("min_depth_ft") for each "lake", calculates the number of years in a time series where summer levels (July-Sept) dropped below the desired minimum depth, assuming docks were installed in June with the minimum desired depth at the end of the dock.

Usage

```
calculate_docks(
  df,
  info = data.frame(lake = c("Pleasant", "Long"), length_ft = c(39, 23), min_depth_ft =
    c(2, 1))
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`info`	a data frame with the average dock length (length_ft) for lakes and minimum desired depth (min_depth_ft) at the end of the dock.

Details

Assumes the average horizontal lake profile for lakes is available within CSLSdata::bathymetry.

Value

docks, a data frame with the following columns:

`lake`	the lake name, e.g. "Pleasant" or "Long"
`metric`	name of the hydrologic metric, in this case, "dock"
`variable`	"num_no_move" (number of years dock does not need to be moved after a successful install), "percent_no_move" (percent of years dock does not need to be moved after a successful install), "num_install" (number of years a dock is successfully installed), "percent_install" (percent of years a dock is successfully installed), "percent_good_year" (percent of years a dock is both successfully installed and does not need to be moved)
`value`	value of the metric

calculate_durations *Calculate durations*

Description

Given a data frame with a "lake", "date", and "level" columns as well as a vector of desired exceedance probabilities, calculates consecutive months at/above (for probabilities <= 50%) or at/below (for probabilities >= 50%) the exceedance levels.

Usage

```
calculate_durations(
  df,
  probs = c(10, 25, 50, 75, 90),
  departures = FALSE,
  exceeds = NULL
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`probs`	a vector with all exceedance probabilities to calculate. Defaults to c(10, 25, 75, 90).
`departures`	logical defaults to FALSE. If TRUE, calculates durations 1ft above median and 1ft below median
`exceeds`	defaults to NULL. If provided, should be a data frame with first column "lake" and subsequent columns corresponding to exceedance probabilities (e.g., "10", "25", "50", "75", "90") and associated lake levels.

Value

durations, a data frame with the following columns:

`lake`	name of lake
`variable`	exceedance probability, e.g., "10", "25", "75", "90", "a50" (for above 50%), "b50" (for below 50%)
`value`	one count of number of months levels were consecutively above or below the given exceedance probability

calculate_exceedances *Calculate exceedance levels*

Description

Given a data frame with a "lake" and "level" column, calculates levels associated with desired exceedance probabilities.

Usage

```
calculate_exceedances(
  df,
  probs = c(10, 25, 50, 75, 90),
  departures = NULL,
  melted = TRUE
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`probs`	a vector with all exceedance probabilities to calculate. Defaults to c(10, 25, 50, 75, 90)
`departures`	optional vector with departures from median to evaluate probability of occurrence. Defaults to NULL to trigger exceedance probability calculations instead.
`melted`	logical defaults to true to indicate should melt data frame to just 3 columns (lake, variable, value). Otherwise, keeps as data frame with one column per exceedance probability.

Value

exceeds, a data frame with the names of all lakes and corresponding exceedance levels

`calculate_lake` *Calculate frequency of lake and flood conditions*

Description

Given a data frame with columns for "lake" and "level" as well as additional information about the elevation at which lake becomes a lake and suitable for paddleboating and the elevation at which the lake floods back floating leaf plants, evaluate the frequency of lake conditions, good paddleboating conditions during warm months (April thorugh September), and flooded conditions at all lakes with lake information.

Usage

```
calculate_lake(
  df,
  lake_info = data.frame(lake = c("Long", "Plainfield"), min_elev_m = c(333.8985,
    333.1118), max_elev_m = c(335.536, 335.0016))
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`lake_info`	a data frame with the "min_elev_m" and "max_elev_m" at which each "lake" becomes a lake/good for paddleboating and floods back vegetation.

Value

a data frame with the number of months and percent of time with lake, good paddleboating, and flooded vegetation conditions at each lake included in "lake_info".

`calculate_metrics` *Calculate hydrologic metrics evaluated in the CSLS*

Description

This function calculates hydrologic metrics describing the magnitude, frequency, duration, rate of change, and timing of lake levels.

Usage

```
calculate_metrics(
  df_month,
  dts = c(1, 3, 12),
  metrics = c("median_level", "cv_level", "exceedance_level", "volume", "area",
    "mean_depth", "max_depth", "centrarchid_substrate", "vegetation_area",
    "exceedance_range", "depart_median", "median_dur", "cv_dur", "num_dur", "num_2yr",
    "median_rise_rate", "cv_rise_rate", "median_fall_rate", "cv_fall_rate", "fast_rise",
    "fast_fall", "good_spawning", "move_dock", "turtle_bay", "stratification", "is_lake",
    "paddlesports", "motorboat", "season_compare", "fast_rise_decade",
    "fast_fall_decade", "num_dur_decade", "num_2yr_decade",      "move_dock_decade"),
  dur_exceeds = NULL
)
```

Arguments

`df_month` data frame to use. Must include columns for "date" (must be POSIXct), "lake" (must be factor), and one with the lake levels (named in "col_name" argument).

`dts` vector of dt values (months) to calculate metrics for, defaults to 1 (monthly), 3 (seasonal), and 12 (annual).

`metrics` a list of which metrics to use. Defaults to all of them c("median_level", "cv_level", "exceedance_level", "exceedance_range", "median_dur", "cv_dur", "median_rise_rate", "cv_rise_rate", "median_fall_rate", "cv_fall_rate").

`dur_exceeds` optional data frame with columns for "lake" and exceedance probabilities (e.g., "10", "25", "50", "75", "90) and associated lake levels. Used to calculate durations above/below these levels. Defaults to NULL to calculate durations on exceedance proabablities on provided time series.

Value

summary, a data frame with the following columns:

`lake` name of lake, character

`metric` name of metric, corresponds with values in inputted "metrics" argument, character

`variable` identifier for different types of metric. For example, the metric "median_rise_rate" will have variables of "1", "3", and "12" associated with it indicating the median rate over 1 month, 3 months, and 12 months, character

`value` calculated value of the metric, numeric

`series` type of time series used (monthly, seasonal, annual)

`calculate_motorboat` *Calculate frequency of legal motorboat conditions*

Description

Given a data frame with columns for "lake" and "level" as well as additional information about the elevation at which lakes are suitable for motorboating (lake area > 62.5 acres), evaluate the frequency of legal motorboating conditions at all lakes with motorboat information.

Usage

```
calculate_motorboat(
  df,
  boat_info = data.frame(lake = "Pleasant", elev_m = 292.6422)
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`boat_info`	a data frame with the "elev_m" at which each "lake" switches from legal to not legal for motorboating.

Value

a data frame with the number of months and percent of time with mixed conditions at each lake included in "boat_info".

`calculate_paddlesports` *Calculate frequency of lake and flood conditions*

Description

Given a data frame with columns for "lake" and "level" as well as additional information about the elevation at which lake becomes a lake and suitable for paddleboating and the elevation at which the lake floods back floating leaf plants, evaluate the frequency of lake conditions, good paddleboating conditions during warm months (April thorugh September), and flooded conditions at all lakes with lake information.

Usage

```
calculate_paddlesports(
  df,
  lake_info = data.frame(lake = "Long", elev_m = 334.29)
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`lake_info`	a data frame with the "min_elev_m" and "max_elev_m" at which each "lake" becomes a lake/good for paddleboating and floods back vegetation.

Value

a data frame with the number of months and percent of time with lake, good paddleboating, and flooded vegetation conditions at each lake included in "lake_info".

`calculate_plant_area` *Calculate area of each vegetation class corresponding to lake elevations*

Description

Given a metrics data frame with values for exceedance_level, use the information in CSLSdata::bathymetry to calculate the area of each plant community at each exceedance level.

Usage

```
calculate_plant_area(df)
```

Arguments

`df`	a data frame with a "lake", "metric", "variable", and "value" columns where one of the metrics is "exceedance_level".

Value

plant_areas, a data frame with "lake", "metric", "variable", and "value" where the metrics are plant community types and the variables are all the exceedance level probabilities inputted in df, and the values are the corresponding plant areas in m^2.

`calculate_rates` *Calculate rates of change*

Description

Given a data frame with a "lake", "date", and "level" columns as well as a vector of desired lag (e.g., 1 month, 3 months, 12 months), calculates the change in lake levels over each time period.

Usage

```
calculate_rates(df, months = c(1, 3, 12))
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`months`	a vector with all lag times (in months) to consider, defaults to c(1, 3, 12).

Value

rates, a data frame with the following columns:

`lake`	name of lake
`variable`	number of months lagged (e.g., 1, 3, 12)
`value`	change in lake levels over given time period

`calculate_spawning` *Calculate good spawning years*

Description

Given a data frame with a "lake", "level", "year", and "month" columns, calculates whether a year is good (TRUE) or bad (FALSE) for pike spawning.

Usage

```
calculate_spawning(df, growing_months = c(4, 5, 6, 7, 8, 9, 10))
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`growing_months`	months associated with growing season. Defaults to c(5, 6, 7, 8, 9, 10).

Value

good_spawning, a data frame with the lakes, years, and whether spring levels were sufficiently high, summer levels were sufficiently steady, and overall the year was good for pike spawning.

`calculate_stratification` *Calculate frequency of mixed (unstratified) conditions*

Description

Given a data frame with columns for "lake" and "level" as well as additional information about the elevation at which lakes stratify, evaluate the frequency of mixed conditions at all lakes with stratification information.

Usage

```
calculate_stratification(
  df,
  strat_info = data.frame(lake = "Pleasant", elev_m = 297.67)
)
```

Arguments

`df`	a data frame with a "lake" and a "level" column
`strat_info`	a data frame with the "elev_m" at which each "lake" switches from stratified to mixed.

Value

a data frame with the number of months and percent of time with mixed conditions at each lake included in "strat_info".

`calculate_substrate` *Calculate centrarchid substrate area at given Pleasant Lake elevation*

Description

Given a data frame with the exceedance level metrics for a single set of simulations at Pleasant Lake, calculate the associated substrate area for centrarchid fish. Uses CSLSdata::fish_substrate, but converts areas from acres to m^2 to be consistent with other area metrics.

Usage

```
calculate_substrate(df)
```

Arguments

`df`	a data frame with a "lake", "metric", "variable", and "value" columns where one of the metrics is "exceedance_level".

Value

substrate, a data frame with "lake", "metric", "variable", and "value" where the metric is "centrarchid_substrate", the variables are all the exceedance level probabilities inputted in df, and the value is the mean substrate hardness at generalized centrarchid spawning depth.

`calculate_substrate_hardness` *Calculate centrarchid substrate area at given Pleasant Lake elevation*

Description

Given a data frame with the exceedance level metrics for a single set of simulations at Pleasant Lake, calculate the associated substrate area for centrarchid fish. Uses CSLSdata::fish_substrate, but converts areas from acres to m^2 to be consistent with other area metrics.

Usage

```
calculate_substrate_hardness(df)
```

Arguments

`df` a data frame with a "lake", "metric", "variable", and "value" columns where one of the metrics is "exceedance_level".

Value

substrate, a data frame with "lake", "metric", "variable", and "value" where the metric is "centrarchid_substrate", the variables are all the exceedance level probabilities inputted in df, and the value is the mean substrate hardness at generalized centrarchid spawning depth.

`calculate_turtle` *Calculate frequency of connection to Turtle Bay*

Description

Given a data frame with columns for "lake", "level", "month", and "year" as well as additional information about the elevation of Turtle Bay ("turtle_elev_m") and minimum required clearance for good connection ("turtle_clearance_ft"), calculates frequency of connection at all times and during the growing season (March through August).

Usage

```
calculate_turtle(df, turtle_elev_m = 298.7, turtle_clearance_ft = 1)
```

Arguments

`df` a data frame with a "lake" and a "level" column

`turtle_elev_m` elevation of turtle bay inlet by eye examination of bathymetry raster (m).

`turtle_clearance_ft`
minimum clearnace needed for good connection to turtle bay.

Value

a data frame with the number of months and percent of time with connection to Turtle Bay during all months and during warm season (March through August) months.

`check_stratification` *Calculate frequency of mixed (unstratified) conditions*

Description

Given a data frame with columns for "lake" and "level" as well as additional information about the elevation at which lakes stratify, evaluate the frequency of mixed conditions at all lakes with stratification information.

Usage

```
check_stratification(
  this_rule,
  this_hydro,
  metric_uncertainty,
  strat_info = data.frame(lake = "Pleasant", elev_m = 297.67)
)
```

Arguments

`this_rule` data frame with information on "percent", "difference", "significant_if" (i.e., "higher" or "lower") for a single indicator.

`this_hydro` data frame with the hydrologic metrics (solute_budget) to evaluate.

`metric_uncertainty`
data frame with lake, metric, variable, and allowable "difference" due to uncertainty in the metric. Currently evaluated as the standard deviation in the "no irrigation" scenarios of the metric.

`strat_info` a data frame with the "elev_m" at which each "lake" switches from stratified to mixed.

Value

this_hydro, a data frame noting thresholds and impact.

`compare_scenarios` *Compare scenarios*

Description

This function compares two time series and evaluates the second series for ecologically significant differences from the first.

Usage

```
compare_scenarios(
  df1,
  df2,
  metric_uncertainty,
  rules = CSLSscenarios::ecological_rules,
  bathymetry = CSLSdata::bathymetry
)
```

Arguments

`df1` data frame with baseline hydrologic metrics

`df2` data frame with hydrologic metrics of scenario being evaluated for significant impact.

`metric_uncertainty`
data frame with lake, metric, variable, and allowable "difference" due to uncertainty in the metric. Currently evaluated as the standard deviation in the "no irrigation" scenarios of the metric.

`rules` data frame with ecological rules for ecological indicators related to hydrologic metrics.

`bathymetry` data frame with bathymetric relationships with parameters like lake area, lake volume, plant area, etc.

Value

comparison, a data frame with the following columns:

`lake` name of lake

`hydrology` type of hydrologic metric (e.g., magnitude)

`metric` hydrologic metric (e.g., exceedance_level)

`variable` type of hydrologic metric (e.g. "90" for 90th percentile exceedance level)

`category` category of ecological indicator (e.g., plants, fish)

`indicator` ecological indicator (e.g., volume_habitat)

`impacted` logical, TRUE if this ecological indicator is impacted from baseline under this scenario

`significant_if` notes whether scenario is significant if scenario values are "higher" or "lower" than threshold values

`value1` base value of hydrologic metric

`threshold` threshold value of hydrologic metric

`value2` scenario value of hydrologic metric

`threshold_diff` difference between threshold and base value of hydrologic metric

`diff` difference between scenario and base value of hydrologic metric

`bathy_significant_if`
notes whether scenario is significant if scenario child/bathymetric values are "higher" or "lower" than child/bathymetric threshold values

`bathy1` base value of child/bathymetric metric (if applicable)

`bathy_threshold`
threshold value of child/bathymetric metric (if applicable)

`bathy2` scenario value of child/bathymetric metric (if applicable)

`bathy_threshold_diff`
difference between bathy_threshold and bathy1

`bathy_diff` difference between bathy2 and bathy1

`ecological_rules` *Dataset: Ecological rules for significant impact*

Description

Dataset: Ecological rules for significant impact

Usage

```
data(ecological_rules)
```

Format

A data frame with the following columns.

hydrology type of hydrologic metric (magnitude, frequency, duration, rate of change, or timing)

category category of ecological indicator (fish, plants, chemistry, human_use)

indicator ecological indicator (e.g., "volume_habitat", "good_spawning")

metric name of hydrologic metric related to this ecological indicator

variable identifier for different types of metric. For example, the metric "exceedance_level" will have variables for each exceedance level (10, 25, 50, 75, 90)

bathy_metric if applicable, name of child metric that needs to be calculated from hydrologic metric before evaluating significance

percent percent difference from base value allowed before change is considered a "significant impact"

difference additive difference from base value allowed before change is considered a "significant impact"

significant_if whether sigificance occurs at "lower" values or "higher" values compared to base value

Pleasant whether this indicator should be evaluated for Pleasant Lake

Long whether this indicator should be evaluated for Long Lake

Plainfield whether this indicator should be evaluated for Plainfield Lake

---

`evaluate_impact_rules` *Evaluate ecological impact rules to apply to metric values*

Description

Given information about whether impact is determined by a multiplicative factor (i.e., percent difference), additive difference (i.e., "difference") and whether significance occurs at values higher or lower than the base value, synthesizes this into a list for use in calculations of thresholds.

Usage

```
evaluate_impact_rules(this_rule, metric_uncertainty)
```

Arguments

`this_rule` data frame with information on "percent", "difference", "significant_if" (i.e., "higher" or "lower") for a single indicator.

`metric_uncertainty` data frame with lake, metric, variable, and allowable "difference" due to uncertainty in the metric. Currently evaluated as the standard deviation in the "no irrigation" scenarios of the metric.

Value

impact, a list with values for "factor", "difference" and "significant_if"

`evaluate_solute_median` *Evaluate median solute concentration*

Description

Given metrics related to solute concentration, evaluate whether new median concentration is lower thant the 90% concentration or higher than the 10% concentration.

Usage

```
evaluate_solute_median(this_hydro, this_rule, metric_uncertainty)
```

Arguments

`this_hydro` data frame with the hydrologic metrics (solute_budget) to evaluate.

`this_rule` data frame with information on "percent", "difference", "significant_if" (i.e., "higher" or "lower") for a single indicator.

`metric_uncertainty` data frame with lake, metric, variable, and allowable "difference" due to uncertainty in the metric. Currently evaluated as the standard deviation in the "no irrigation" scenarios of the metric.

Value

this_hydro, a data frame noting thresholds and impact.

`extrapolate_bathymetry` *Extrapolate relationship between elevation and bathymetric metrics*

Description

Assumes ecological rule is a percent change (increase or decrease) from baseline value (e.g., 10% decrease), not a numerical difference (e.g., 2 fewer times).

Usage

```
extrapolate_bathymetry(
  this_hydro,
  this_rule,
  bathymetry,
  bathy_metric,
  metric_uncertainty
)
```

Arguments

`this_hydro` data frame with the hydrologic metrics (exceedance levels) to evaluate.

`this_rule` data frame with the ecological rules for given parameter to evaluate.

`bathymetry` data frame with relationships between lake elevation and other parameters (e.g., lake area, lake volume, plant area, substrate area)

`bathy_metric` name of column in bathymetry to evaluate

`metric_uncertainty` data frame with lake, metric, variable, and allowable "difference" due to uncertainty in the metric. Currently evaluated as the standard deviation in the "no irrigation" scenarios of the metric.

Details

Because so many bathymetric relationships are non-monotonic, had to come up with a hacky way of finding correct elevation match for given threshold child bathy parameter. From check on monotonic child parameters, should yield accurate threshold elevations to within 1 cm.

Value

impact_evaluation, a data frame with the following columns:

`lake` name of lake, character

`metric` hydrologic metric, i.e. "exceedance_level"

`variable` hydrologic metric variable, i.e. 10, 25, 50, 75, or 90

`value1` baseline value for hydrologic metric

`threshold` threshold value for hydrologic metric for impact

`value2` scenario value for hydrologic metric

`impacted` logical, indicates whether lake is impacted relative to this ecological indicator under this scenario (TRUE) or not (FALSE)

`MODFLOW_comparison` *Dataset: comparison of MODFLOW scenarios*

Description

Comparison of MODFLOW scenarios with determinations of impact for variaous ecological indicators.

Usage

```
data(MODFLOW_comparison)
```

Format

A data frame with the following columns.

lake name of lake, e.g., Pleasant, Long, Plainfield

hydrology type of hydrologic metric (e.g., "magnitude")

metric name of hydrologic metrics, e.g. median_level, cv_rise_rate

variable name of variation on metrics, e.g. 10 for 10% exceedance level

category type of ecological indicator (e.g., "fish", "plants")

indicator ecological indicator

impacted logical indicated if ecological indicator is impacted by change in hydrology, TRUE or FALSE

significant_if whether metric is impacted from base if it is "lower" or "higher" than the threshold

value1 value of hydrologic metric for base scenario

threshold threshold value of hydrologic metric

value2 value of hydrologic metric for comparison scenario

compare1 only used for solute_budget - concentration (low or high) used for comparison for value2 median concentration

threshold_diff threshold - value1

diff value2 - value1

bathy_significant_if whether bathymetry metric is impacted from base if it is "lower" or "higher" than the bathymetry threshold

bathy1 value of bathymetry metric for base scenario

bathy_threshold threshold value of bathymetry metric

bathy2 value of bathymetry metric for comparison scenario

bathy_threshold_diff bathy_threshold - bathy1

bathy_diff bathy2 - bathy1

scenario MODFLOW scenario, e.g., "cal", "no_irr", "cur_irr", "wells_off"

sim id of MODFLOW simulation

sim_type "base", "conservative", or "permissive"

`MODFLOW_metrics` *Dataset: hydrologic metrics calculated from MODFLOW scenarios*

Description

Hydrologic metrics calculated using all simulation results from monte carlo MODFLOW scenarios

Usage

```
data(MODFLOW_metrics)
```

Format

A data frame with the following columns.

lake name of lake, e.g., Pleasant, Long, Plainfield

metric name of hydrologic metrics, e.g. median_level, cv_rise_rate

variable name of variation on metrics, e.g. 10 for 10% exceedance level

value value of hydrologic metric

series value based on monthly, seasonal, or annual time series

scenario MODFLOW scenario (e.g., "irr" or "no_irr")

sim id of MODFLOW simulation

`select_bounds` *Select upper/lower bounds for comparisons*

Description

Selects a "conservative" and "permissive" MODFLOW simulation for evaluation of impacts.

Usage

```
select_bounds(df, base_scenario = "no_irr", compare_scenario = "cur_irr")
```

Arguments

`df`	a data frame with MODFLOW metrics
`base_scenario`	name of base scenario, defaults to "no_irr"
`compare_scenario`	name of scenario to evaluate for impacts relative to the base scenario, defaults to "irr".

Value

use_sims, a data frame with the following columns:

`sim`	simulations numbers to use
`sim_type`	notes whether this simulation represents a "conservative" or "permissive" determination of impact

`well_rank_dist` *Dataset: Distance of wells from centroild of Long/Pleasant lakes*

Description

Dataset: Distance of wells from centroild of Long/Pleasant lakes

Usage

```
data(well_rank_dist)
```

Format

A data frame with the following columns.

lake name of lake

sim well rank aka simulation number. smaller = closer to lake.

dist_m distance from well to centroid of Long Lake (for Plainfield & Long) or Pleasant Lake (for Pleasant), (m)

dist_mi distance from well to centroid of Long Lake (for Plainfield & Long) or Pleasant Lake (for Pleasant), (miles)

Index