Package 'peacesciencer'

October 14, 2022

Type Package

Title Tools and Data for Quantitative Peace Science Research

Version 1.0.0

Depends R (>= 3.5.0)

Maintainer Steve Miller <steven.v.miller@gmail.com>

Description These are useful tools and data sets for the study of quantitative peace science. The goal for this package is to include tools and data sets for doing original research that mimics well what a user would have to previously get from a software package that may not be well-sourced or well-supported. Those software bundles were useful the extent to which they encourage replications of long-standing analyses by starting the data-generating process from scratch. However, a lot of the functionality can be done relatively quickly and more transparently in the R programming language.

License GPL-2

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.1.2

URL https://github.com/svmiller/peacesciencer/

BugReports https://github.com/svmiller/peacesciencer/issues/

Imports magrittr, dplyr, geosphere, tidyr, stringr, rlang, stevemisc (>= 1.3.0), lifecycle

Suggests countrycode, tibble, testthat, knitr, rmarkdown

NeedsCompilation no

Author Steve Miller [aut, cre] (<https://orcid.org/0000-0003-4072-6263>)

Repository CRAN

Date/Publication 2022-03-24 15:10:02 UTC

R topics documented:

`add_archigos` *Add Archigos political leader information to dyad-year and state-year data*

Description

`add_archigos()` allows you to add some information about leaders to dyad-year or state-year data. The function leans on an abbreviated version of the data, which also comes in this package.

Usage

```
add_archigos(data)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed") or state-year data frame

Details

The function leans on attributes of the data that are provided by the `create_dyadyear()` or `create_stateyear()` function. Make sure that function (or data created by that function) appear at the top of the proverbial pipe.

Value

`add_archigos()` takes a dyad-year or state-year data frame and adds a few summary variables based off the leader-level data. These include whether there was a leader transition in the state-year (or first/second state in the dyad-year), whether there was an "irregular" leader transition, the number of leaders in the state-year, the unique leader ID for Jan. 1 of the year, and the unique leader ID for Dec. 31 of the year.

Author(s)

Steven V. Miller

References

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_archigos()

create_stateyears() %>% add_archigos()
```

`add_atop_alliance` *Add Alliance Treaty Obligations and Provisions (ATOP) alliance data to a dyad-year data frame*

Description

`add_atop_alliance()` allows you to add Alliance Treaty Obligations and Provisions (ATOP) data to a (dyad-year, leader-dyad-year) data frame.

Usage

```
add_atop_alliance(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

Data are from version 5.0 of ATOP.

This function will also work with leader-dyad-years, though users should be careful with leader-level applications of alliance data. Alliance data are primarily communicated yearly, making it possible—even likely—that at least one leader-dyad in a given year is credited with an alliance that was not active in the particular leader-dyad. The ATOP alliance data are not communicated with time measurements more granular than the year, at least for dyad-years. The alliance-level data provided by ATOP do have termination dates, but I am unaware how well these start and termination dates coincide with particular members joining after the fact or exiting early. The alliance phase data appear to communicate that "phases" are understood as beginning/ending when the underlying document is amended in such a way that it affects one of their variable codings, but this may or may not be because of a signatory joining after the fact or exiting early. More guidance will be useful going forward, but use these data for leader-level analyses with that in mind.

Value

add_atop_alliance() takes a (dyad-year, leader-dyad-year) data frame and adds information about the alliance pledge in that given dyad-year from the ATOP data. These include whether there was an alliance with a defense pledge, an offense pledge, neutrality pledge, non-aggression pledge, or pledge for consultation in time of crisis.

Author(s)

Steven V. Miller

References

Leeds, Brett Ashley, Jeffrey M. Ritter, Sara McLaughlin Mitchell, and Andrew G. Long. 2002. Alliance Treaty Obligations and Provisions, 1815-1944. *International Interactions* 28: 237-60.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_atop_alliance()
```

---

add_capital_distance *Add capital-to-capital distance to a data frame*

Description

add_capital_distance() allows you to add capital-to-capital distance to a (dyad-year, leader-year, leader-dyad-year, state-year) data frame. The capitals are coded in the cow_capitals and gw_capitals data frames, along with their latitudes and longitudes. The distance variable that emerges capdist is calculated using the "Vincenty" method (i.e. "as the crow flies") and is expressed in kilometers.

Usage

```
add_capital_distance(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

The function leans on attributes of the data that are provided by one of the "create" functions in this package (e.g. create_dyadyear() or create_stateyear()).

Value

add_capital_distance() takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds the capital-to-capital distance between the first state and the second state (in dyad-year data) or the minimum capital-to-capital distance for a given state in a given year. A minor note about this function: cases of capital transition are recorded in the underlying data but, in the conversion to capital-years (and eventual merging into a dyad-year data frame), the Jan. 1 capital is used for calculating distances.

Author(s)

Steven V. Miller

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
cow_ddy %>% add_capital_distance()

create_stateyears() %>% add_capital_distance()
```

---

add_ccode_to_gw *Add Correlates of War state system codes to your data with Gleditsch-Ward state codes.*

Description

add_ccode_to_gw() allows you to match, as well as one can, Correlates of War system membership data with Gleditsch-Ward system data.

Usage

```
add_ccode_to_gw(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

The data-raw directory on the project's Github contains more information about the underlying data that assists in merging in these codes.

The user will invariably need to be careful and ask why they want these data included. The issue here is that both have a different composition and the merging process will not (and cannot) be perfect.

We can note that a case like Gran Colombia is not too difficult to handle (i.e. CoW does not have this entity and none of the splinter states conflict with CoW's coding). However, there is greater weirdness with a case like the unification of West Germany and East Germany. Herein, Correlates of War treats the unification as the reappearance of the original Germany whereas Gleditsch-Ward treat the unification as an incorporation of East Germany into West Germany. The script will *not* create state-year or dyad-year duplicates for the Gleditsch-Ward codes. The size of the original data remain unchanged. However, there will be some year duplicates for various Correlates of War codes (prominently Serbia and Yugoslavia in 2006). Use with care. You can also use the **countrycode** package. Whether you use this function or the **countrycode** package, do *not* do this kind of merging without assessing the output.

Value

`add_ccode_to_gw()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame that already has Gleditsch-Ward state system codes and adds their corollary Correlates of War codes.

Author(s)

Steven V. Miller

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

create_dyadyears(system = "gw") %>% add_ccode_to_gw()

create_stateyears(system = 'gw') %>% add_ccode_to_gw()
```

`add_contiguity` *Add Correlates of War direct contiguity information to a data frame*

Description

`add_contiguity()` allows you to add Correlates of War contiguity data to a dyad-year, leader-year, or leader-dyad-year, or state-year data frame.

Usage

```
add_contiguity(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

The contiguity codes in the dyad-year data range from 0 to 5. 1 = direct land contiguity. 2 = separated by 12 miles of water or fewer (a la Stannis Baratheon). 3 = separated by 24 miles of water or fewer (but more than 12 miles). 4 = separated by 150 miles of water or fewer (but more than 24 miles). 5 = separated by 400 miles of water or fewer (but more than 150 miles).

Importantly, 0 are the dyads that are not contiguous at all in the CoW contiguity data. This is a conscious decision on my part as I do not think of the CoW's contiguity data as exactly ordinal. Cross-reference CoW's contiguity data with the minimum distance data in this exact package to see how some dyads that CoW codes as not contiguous are in fact very close to each other, sometimes even land-contiguous. For example, Zimbabwe and Namibia are separated by only about a few hundred feet of water at that peculiar intersection of the Zambezi River where the borders of Zambia, Botswana, Namibia, and Zimbabwe meet. There is no contiguity record for this in the CoW data. There are other cases where contiguity records are situationally missing (e.g. India-Bangladesh, and Bangladesh-Myanmar in 1971) or other cases where states are much closer than CoW's contiguity data imply (e.g. Pakistan and the Soviet Union were separated by under 30 kilometers of Afghani territory). The researcher is free to recode these 0s to be, say, 6s, but this is why **peacesciencer** does not do this.

For additional clarity, the "master records" produce duplicates for cases when the contiguity relationship changed in a given year. This function returns the *minimum* contiguity relationship observed in that given year. There should be no duplicates in the returned output.

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

Value

`add_contiguity()` takes a data frame and adds information about the contiguity relationship based on the "master records" for the Correlates of War direct contiguity data (v. 3.2). If the data are dyad-year (or leader dyad-year), the function returns the lowest contiguity type observed in the dyad-year (if contiguity is observed at all). If the data are state-year (or leader-year), the data return the total number of land and sea borders calculated from these master records.

Author(s)

Steven V. Miller

References

Stinnett, Douglas M., Jaroslav Tir, Philip Schafer, Paul F. Diehl, and Charles Gochman (2002). "The Correlates of War Project Direct Contiguity Data, Version 3." Conflict Management and Peace Science 19 (2):58-66.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
```

```
cow_ddy %>% add_contiguity()

create_stateyears() %>% add_contiguity()
```

`add_cow_alliance` *Add Correlates of War alliance data to a data frame*

Description

`add_cow_alliance()` allows you to add Correlates of War alliance data to a dyad-year data frame

Usage

```
add_cow_alliance(data)
```

Arguments

`data`	a dyad-year or leader-dyad-year data frame (either "directed" or "non-directed")

Details

Duplicates in the original directed dyad-year alliance data were pre-processed. Check `cow_alliance` in the package's `data-raw` directory on Github for more information.

This function will also work with leader-dyad-years, though users should be careful with leader-level applications of alliance data. Alliance data are primarily communicated yearly, making it possible—even likely—that at least one leader-dyad in a given year is credited with an alliance that was not active in the particular leader-dyad. The Correlates of War's alliance data are not communicated with time measurements more granular than the year. Apply these data to leader-level analyses with that in mind.

Value

`add_cow_alliance()` takes a dyad-year data frame and adds information about the alliance pledge in that given dyad-year. These include whether there was an alliance with a defense pledge, neutrality pledge, non-aggression pledge, or pledge for consultation in time of crisis (entente).

Author(s)

Steven V. Miller

References

Gibler, Douglas M. 2009. *International Military Alliances, 1648-2008*. Congressional Quarterly Press.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_cow_alliance()
```

`add_cow_majors` *Add Correlates of War major power information to a data frame*

Description

`add_cow_majors()` allows you to add Correlates of War major power variables to a dyad-year, leader-year, leader dyad-year, or state-year data frame.

Usage

```
add_cow_majors(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

Value

`add_cow_majors()` takes a data frame and adds information about major power status for the given state or dyad in that year. If the data are dyad-year (or leader dyad-year), the function returns two columns for whether the first state (i.e. `ccode1`) or the second state (i.e. `ccode2`) are major powers in the given year, according to the Correlates of War. 1 = is a major power. 0 = is not a major power. If the data are state-year (or leader-year), the functions returns just one column (`cowmaj`) for whether the state was a major power in a given state-year.

Author(s)

Steven V. Miller

References

Correlates of War Project. 2017. "State System Membership List, v2016." Online, https://correlatesofwar.org/data-sets/state-system-membership

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_cow_majors()
```

`add_cow_mids` *Add Correlates of War (CoW) Militarized Interstate Dispute (MID) data to dyad-year data frame*

Description

`add_cow_mids()` merges in CoW's MID data to a dyad-year data frame. The version of the CoW-MID data in this package is version 5.0.

Usage

```
add_cow_mids(data, keep)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed")
`keep`	an optional parameter, specified as a character vector, passed to the function in a `select(one_of(.))` wrapper. This allows the user to discard unwanted columns from the directed dispute data so that the output does not consume too much space in memory. Note: the Correlates of War system codes (`ccode1`, `ccode2`), the observation year (`year`), the presence or absence of an ongoing MID (`cowmidongoing`), and the presence or absence of a unique MID onset (`cowmidonset`) are *always* returned. It would be foolish and self-defeating to eliminate those observations. The user is free to keep or discard anything else they see fit. If `keep` is not specified in the function, the ensuing output returns everything.

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. This merging process employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame.

The function will also return a message to the user about the case-exclusion rules that went into this process. Users who are interested in implementing their own case-exclusion rules should look up the "whittle" class of functions also provided in this package.

Value

add_cow_mids() takes a dyad-year data frame and adds dyad-year dispute information from the CoW-MID data.

Author(s)

Steven V. Miller

References

Palmer, Glenn, and Roseanne W. McManus and Vito D'Orazio and Michael R. Kenwick and Mikaela Karstens and Chase Bloch and Nick Dietrich and Kayla Kahn and Kellan Ritter and Michael J. Soules. 2021. "The MID5 Dataset, 2011–2014: Procedures, coding rules, and description" *Conflict Management and Peace Science*.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
cow_ddy %>% add_cow_mids()

# keep just the dispute number and Side A/B identifiers
cow_ddy %>% add_cow_mids(keep=c("dispnum","sidea1", "sidea2"))
```

---

add_cow_trade *Add Correlates of War trade data to a data frame*

Description

add_cow_trade() allows you to add Correlates of War trade data to your (dyad-year, leader-year, leader-dyad-year, state-year) data frame

Usage

```
add_cow_trade(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

For the dyad-year (and leader-dyad-year) data, there must be some kind of information loss in order to work within the limited space available to this package. This package loads a truncated version of the data in which the trade values are rounded to three decimal points in order to greatly reduce the disk space for this package. I do not think this to be terribly problematic, though I admit I do not like it. If this is a problem for your research question, you may want to consider not using this function for dyad-year or leader-dyad-year data.

Be mindful that the data are fundamentally state-year or dyad-year and that extensions to leader-level data should be understood as approximations for leaders (leader-dyads) in a given state-year (dyad-year).

Value

`add_cow_trade()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds information about the volume of trade in that given dyad-year or state-year. For the state-year (leader-year) data, these are minimally the sum of all imports and the sum of all exports. For dyad-year (leader-dyad-year) data, this function returns the value of imports in current million USD in the first country from the second country (and vice-versa) along with their "smooth" equivalents.

Author(s)

Steven V. Miller

References

Barbieri, Katherine, Omar M. G. Keshk, and Brian Pollins. 2009. "TRADING DATA: Evaluating our Assumptions and Coding Rules." *Conflict Management and Peace Science*. 26(5): 471-491.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
# The function below works, but depends on running `download_extdata()` beforehand.
# cow_ddy %>% add_cow_trade()

create_stateyears() %>% add_cow_trade()
```

`add_cow_wars` *Add Correlates of War war data to dyad-year or state-year data frame.*

Description

`add_cow_wars()` allows you to Correlates of War data to a dyad-year or state-year data frame

Usage

```
add_cow_wars(data, type, intratype = "all")
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes
`type`	the type of war you want to add. Options include "inter" or "intra".
`intratype`	the types of armed conflicts the user wants to consider, specified as a character vector. Options include "local issues" and "central control". Applicable only if `type` is "intra".

Details

Intra-state war data are coerced into true state-year data by first selecting the duplicate state-years on unique onsets, then whichever war was the deadliest. The inter-state war data work functionally the same way.

On intra-state wars: the `primary_state` is used to identify the government principally fighting the domestic non-state actor over central control over local issues. Internationalized civil wars are included in the data, but not for outside actors that intervene on behalf of the government or rebel group.

Extra-state war functionality is not available right now as I try to figure out the demand for its use.

Value

`add_cow_wars()` takes a dyad-year or state-year data frame and returns information about wars from either the inter-state or intra-state war data set from the Correlates of War. The function works for state-year data when the user wants information about extra-state wars or intra-state wars. The function works for dyad-year data when the user wants information about inter-state wars.

Author(s)

Steven V. Miller

References

Dixon, Jeffrey, and Meredith Sarkees. 2016. *A Guide to Intra-State Wars: An Examination of Civil Wars, 1816-2014.* Thousand Oaks, CA: Sage.

Sarkees, Meredith Reid, and Frank Wheldon Wayman. 2010. *Resort to War: A Data Guide to Inter-State, Extra-State, Intra-State, and Non-State Wars, 1816-2007.* Washington DC: CQ Press.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

create_stateyears(system = "cow") %>%
add_cow_wars(type = "intra", intratype = "central control")

create_stateyears(system = "cow") %>%
add_cow_wars(type = "intra", intratype = "local issues")
```

```
cow_ddy %>% add_cow_wars(type = "inter")
```

`add_creg_fractionalization` *Add fractionalization/polarization estimates from CREG to a data frame*

Description

`add_creg_fractionalization()` allows you to add information about the fractionalization/polarization of a state's ethnic and religious groups to your data.

Usage

```
add_creg_fractionalization(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

Please see the information for the underlying data `creg`, and the associated R script in the `data-raw` directory, to see how these data are generated.

The `creg` data have a few duplicates. When standardizing to true CoW codes, the duplicates concern Serbia/Yugoslavia in 1991 and 1992 as well as Russia/the Soviet Union in 1991. When standardizing to true Gleditsch-Ward codes, the duplicates concern Serbia/Yugoslavia in 1991 and Russia/Soviet Union in 1991. In those cases, the function does a group-by arrange for the more fractionalized/polarized estimate under the (reasonable, I think) assumption that these are estimates prior to the dissolution of those states. If this is problematic, feel free to consult the underlying data and merge those in manually.

The underlying data have both Gleditsch-Ward codes and Correlates of War codes. The merge it makes depends on what you declare as the "master" system at the top of the pipe (i.e. in `create_dyadyears()` or `create_stateyears()`). If, for example, you run `create_stateyears(system="cow")` and follow it with `add_gwcode_to_cow()`, the merge will be on the Correlates of War codes and not the Gleditsch-Ward codes. You can see the script mechanics to see how this is achieved.

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

Value

add_creg_fractionalization() takes a dyad-year, leader-year, leader-dyad-year, or state-data frame, whether the primary state identifiers are from the Correlates of War system or the Gleditsch-Ward system, and returns information about the fractionalization and polarization of the state(s) in a given year. The function returns four additional columns when the data are state-year and returns eight additional columns when the data are state-year (or leader-year). The columns returned are the fractionalization of ethnic groups, the polarization of ethnic groups, the fractionalization of religious groups, and the polarization of religious groups. When the data are dyad-year (or leader-dyad-year), the return doubles because it provides information for both states in the dyad.

Author(s)

Steven V. Miller

References

Alesina, Alberto, Arnaud Devleeschauwer, William Easterly, Sergio Kurlat and Romain Wacziarg. 2003. "Fractionalization". *Journal of Economic Growth* 8: 155-194.

Montalvo, Jose G. and Marta Reynal-Querol. 2005. "Ethnic Polarization, Potential Conflict, and Civil Wars" *American Economic Review* 95(3): 796–816.

Nardulli, Peter F., Cara J. Wong, Ajay Singh, Buddy Petyon, and Joseph Bajjalieh. 2012. *The Composition of Religious and Ethnic Groups (CREG) Project*. Cline Center for Democracy.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_creg_fractionalization()

create_stateyears() %>% add_creg_fractionalization()

create_stateyears(system = "gw") %>% add_creg_fractionalization()
```

---

add_democracy *Add democracy information to a data frame*

Description

add_democracy() allows you to add estimates of democracy to your data.

Usage

```
add_democracy(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

A vignette on the package's website talks about how these data are here primarily to encourage you to maximize the number of observations in the analysis to follow. Xavier Marquez' `QuickUDS` estimates have the best coverage. If democracy is ultimately a control variable, or otherwise a variable not of huge concern for the analysis (i.e. the user has no particular stake on the best measurement of democracy or the best conceptualization and operationalization of "democracy"), please use Marquez' estimates instead of Polity or V-dem. If the user is doing an analysis of inter-state conflict, and across the standard post-1816 domain in conflict studies, *definitely* don't use the Polity data because the extent of its missingness is both large and unnecessary. Please read the vignette describing these issues here: http://svmiller.com/peacesciencer/articles/democracy.html

Value

`add_democracy()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds information about the level of democracy for the state or two states in the dyad in a given year. If the data are dyad-year or leader-dyad-year, the function adds six total columns for the first state (i.e. `ccode1` or `gwcode1`) and the second state (i.e. `ccode2` or `gwcode2`) about the level of democracy measured by the Varieties of Democracy project (`v2x_polyarchy`), the Polity project (`polity2`), and Xavier Marquez' `QuickUDS` extensions/estimates. If the data are state-year or leader-year, the function returns three additional columns to the original data that contain that same information for a given state in a given year.

Author(s)

Steven V. Miller

References

Coppedge, Michael, John Gerring, Carl Henrik Knutsen, Staffan I. Lindberg, Jan Teorell, David Altman, Michael Bernhard, M. Steven Fish, Adam Glynn, Allen Hicken, Anna Luhrmann, Kyle L. Marquardt, Kelly McMann, Pamela Paxton, Daniel Pemstein, Brigitte Seim, Rachel Sigman, Svend-Erik Skaaning, Jeffrey Staton, Agnes Cornell, Lisa Gastaldi, Haakon Gjerlow, Valeriya Mechkova, Johannes von Romer, Aksel Sundtrom, Eitan Tzelgov, Luca Uberti, Yi-ting Wang, Tore Wig, and Daniel Ziblatt. 2020. "V-Dem Codebook v10" Varieties of Democracy (V-Dem) Project.

Marshall, Monty G., Ted Robert Gurr, and Keith Jaggers. 2017. "Polity IV Project: Political Regime Characteristics and Transitions, 1800-2017." Center for Systemic Peace.

Marquez, Xavier, "A Quick Method for Extending the Unified Democracy Scores" (March 23, 2016). doi: 10.2139/ssrn.2753830

Pemstein, Daniel, Stephen Meserve, and James Melton. 2010. "Democratic Compromise: A Latent Variable Analysis of Ten Measures of Regime Type." *Political Analysis* 18(4): 426-449.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_democracy()

create_stateyears(system="gw") %>% add_democracy()
create_stateyears(system="cow") %>% add_democracy()
```

`add_fpsim` *Add dyadic foreign policy similarity measures to your data*

Description

`add_fpsim()` allows you to add a variety of dyadic foreign policy similarity measures to your (dyad-year, leader-dyad-year) data frame

Usage

```
add_fpsim(data, keep)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes
`keep`	an optional parameter, specified as a character vector, about what dyadic foreign policy similarity measure(s) the user wants returned from this function. If `keep` is not specified, the function returns all 14 dyadic foreign policy similarity measures calculated by Haege (2011). Otherwise, the function subsets the underlying data to just what the user wants and merges in that.

Details

For the dyad-year (and leader-dyad-year) data, there must be some kind of information loss in order to reduce the disk space data like these command. In this case, all calculations are rounded to three decimal spots. I do not think this to be terribly problematic, though I admit I do not like it. If this is a problem for your research question (though I can't imagine it would be), you may want to consider not using this function for dyad-year or leader-dyad-year data.

Be mindful that the data are fundamentally dyad-year and that extensions to leader-level data should be understood as approximations for leaders-dyads in a given dyad-year.

The data this function uses are directed dyad-year and the merge is a left-join, making this function agnostic about whether your dyad-year (or leader-dyad-year) data are directed or non-directed.

Haege's (2011) article reads at first glance as agnostic about which of these particular measures you should consider a "preferred" or "default" measure of dyadic foreign policy similarity. Indeed,

the 2011 publication in *Political Analysis* mostly drives the point home that *S* has important limitations and the multiple variants Haege calculates are not substitutable. This means a user interested in measuring dyadic foreign policy similarity might have to cycle through all of them to assess their varying effects whereas a user interested in this as just a control variable for the model can (probably) get by with picking just one and not belaboring the measure any further.

Suggested Defaults:

An evaluation of the data, the article, and an email exchange with the author leads to the following points the user should consider. What follows is a rationale for why users should think of kappa as a default measure for dyadic foreign policy similarity, though why the "valued" equivalent for the alliance data is an inadvisable default. The example at the end of the document offers the operational "nudge" for what the user should want from this function.

- The choice of measure will in part depend on the temporal domain. If the user has just a post-WWII sample, the UN voting measures offer better coverage. We're all partial to the alliance data, though, because of its 19th century coverage.
- Haege implores the use of chance-corrected measures, like Cohen's (1960) kappa or Scott's (1955) pi. Of the two, Haege suggests kappa over pi. The rationale is the user would need to build in a very strong assumption that the baseline propensity of forming a tie in the dyad is the same for both members of the dyad to make Scott's (1955) pi as appropriate an estimate as Cohen's (1960) kappa even as both have the important chance correction.
- The choice of squared versus absolute distances is arbitrary. Users probably do not think about the differences, or know about the differences. *S* was usually calculated with absolute differences in software packages, though this was never usually belabored to the user. Comparability with *S* might be an argument in favor of absolute distance as a default, but keep in mind that squared distances are much more commonly used in most other types of distance and association metrics.
- The choice of binary or valued is also a design choice for the user to consider on the full merits, though the practice of valuing alliance ties on a quantitative scale builds in strong assumptions about the scale of alliance strength as presented in something like the Correlates of War or ATOP typology. *S* has traditionally done this by default, which is another reason its application in a lot of quantitative peace science research is suspect.

Value

`add_fpsim()` takes a (dyad-year, leader-dyad-year) data frame and adds information about the dyadic foreign policy similarity, based on several measures calculated and offered by Frank Haege.

Author(s)

Steven V. Miller

References

The Main Source of the Data:

For any use of these data whatsoever (except for Tau-b), please cite Haege (2011). Data are version 2.0.

- Haege, Frank M. 2011. "Choice or Circumstance? Adjusting Measures of Foreign Policy Similarity for Chance Agreement." *Political Analysis* 19(3): 287-305.

Tau-b is calculated by me and not Haege, and no additional citation (beyond citing the package) is necessary.

Citations for the Particular Similarity Measure You Choose:

Additional citations depend on what particular measure of similarity you're using, whether Kendall's (1938) Tau-b, Signorino and Ritter's (1999) *S*, Cohen's (1960) kappa and Scott's (1955) pi. Haege (2011) is part of a chorus arguing against the use of *S*, though *S* measures are included in these data if you elect to ignore the chorus and use this measure. Likewise, Tau-b is in here, though it is not a good measure of dyadic foreign policy similarity for reasons that Signorino and Ritter (1999) mention. Haege (2011) argues for a chance-corrected measure of dyadic foreign policy similarity, either Cohen's (1960) kappa or Scott's (1955) pi.

- Cohen, Jacob. 1960. "A Coefficient of Agreement for Nominal Scales." *Educational and Psychological Measurement* 20(1): 37-46.
- Kendall, M.G. 1938. "A New Measure of Rank Correlation." *Biometrika* 30(1/2): 81–93.
- Scott, William A. 1955. "Reliability of Content Analysis: The Case of Nominal Scale Coding." *Public Opinion Quarterly* 19(3): 321–5.
- Signorino, Curtis S. and Jeffrey M. Ritter. "Tau-b or Not Tau-B: Measuring the Similarity of Foreign Policy Positions." 43(1): 115–44.

Citations for the Underlying Data Informing the Similarity Measure:

Haege (2011) also suggests you cite the underlying data informing the similarity measure, whether it is UN voting or alliances. In his case, he recommended a Voeten citation from 2013 and the alliance data proper. In the case of the alliances, I know Gibler's (2009) book is recommended even if the alliance data have since been updated (and reflected in this measure). In the UN voting data, my understanding is the 2017 paper in *Journal of Conflict Resolution* is also the preferred citation.

- Bailey, Michael A., Anton Strezhnev, and Erik Voeten. 2017. "Estimating the Dynamic State Preferences from United Nations Voting Data." *Journal of Conflict Resolution* 61(2): 430–456.
- Gibler, Douglas M. 2009. *International Military Alliances, 1648-2008*. Washington DC: CQ Press.

Examples

```
## Not run:
# just call `library(tidyverse)` at the top of the your script.
library(magrittr)
# The function below works, but depends on
# running `download_extdata()` beforehand.
cow_ddy %>% add_fpsim()

# Select just the two kappa measures that are suggested defaults.
# `kappaba`: kappa for binary alliance data if you have pre-WWII data.
# `kappavv`: kappa for UN voting data if you just post-WWII data.
cow_ddy %>% add_fpsim(keep=c("kappaba", "kappavv"))


## End(Not run)
```

add_gml_mids *Add Gibler-Miller-Little (GML) Militarized Interstate Dispute (MID) data to a data frame*

Description

`add_gml_mids()` merges in GML's MID data to a (dyad-year, leader-year, leader-dyad-year, state-year) data frame. The current version of the GML MID data is 2.2.1.

Usage

```
add_gml_mids(data, keep, init = "sidea-all-joiners")
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes
`keep`	an optional parameter, specified as a character vector, applicable to just the dyad-year data, and passed to the function in a `select(one_of(.))` wrapper. This allows the user to discard unwanted columns from the directed dispute data so that the output does not consume too much space in memory. Note: the Correlates of War system codes (`ccode1`, `ccode2`), the observation year (`year`), the presence or absence of an ongoing MID (`gmlmidongoing`), and the presence or absence of a unique MID onset (`gmlmidonset`) are *always* returned. It would be foolish and self-defeating to eliminate those observations. The user is free to keep or discard anything else they see fit. If `keep` is not specified in the function, the ensuing output returns everything.
`init`	how should initiators be coded? Applicable only to state-year, leader-dyad-year, and leader-year data. This parameter accepts one of three possible values (`"sidea-orig"`, `"sidea-with-joiners"`, `"sidea-all-joiners"`). `"sidea-orig"` = a state initiates a MID (which appears as a summary return in the output) if the state was on Side A at the onset of the dispute. `"sidea-with-joiners"` = a state initiates a MID (which appears as a summary return in the output) if the state was on Side A at the onset of the dispute or if the state joined the MID on Side A. `"sidea-all-joiners"` = a state initiates a MID (which appears as a summary return in the output) if the state was on Side A at the onset of the dispute or if it joined at any point thereafter. See details section for more discussion. The default is `"sidea-all-joiners"`.

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. This merging process employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame.

The function will also return a message to the user about the case-exclusion rules that went into this process. Users who are interested in implementing their own case-exclusion rules should look up the "whittle" class of functions also provided in this package.

Determining "initiation" for state-year summaries of inter-state disputes is possible since there is an implied directionality of "initiation." In about half of all cases, this is straightforward. You can use the participant summaries and determine that if the dispute was bilateral and the dispute did not escalate beyond an attack, the state on Side A initiated the dispute. For multilateral MIDs, these conditions still hold at least for originators. However, there is *considerable* difficulty for cases where 1) participant-level summaries suggested actions at the level of clash or higher, 2) the participant was a joiner and not an originator. The effort required to flesh this out is enormous, and perhaps forthcoming in a future update.

`add_gml_mids()` allows you to make one of three judgment calls here (see the arguments section of the documentation). If it were my call to make, I would say you should probably use the option `"sidea-all-joiners"`. My review of the MID data with Doug Gibler suggests most states that join a dispute are not roped into a conflict (i.e. targeted by some other state) after the first incident. They routinely initiate their entry into the conflict, which is what this concept of "initiation" is supposed to capture in the literature. There are no doubt cases where some third state is brought into the dispute by the actions of some other state even as the original MID coding rules place a high barrier on coding that type of dispute entry. However, the time required to individually assess whether a state initiated their entry into a MID under something other than the simplest of cases (e.g. bilateral cases where the highest participant action fell short of a clash) would be too time-consuming. It would require an audit of almost half of all participant-level summaries in the data. In a forthcoming publication, Gibler and Miller offer excellent coverage here with a new data set on militarized events. However, this would include only confrontations after World War II.

Value

`add_gml_mids()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds dispute information from the GML MID data. If the data are dyad-year, the return is a laundry list of information about onsets, ongoing conflicts, and assorted participant- and dispute-level summaries. If the data are leader-dyad-year, these are carefully matched to leaders as well. If the data are state-year or leader-year, the function returns information about ongoing disputes (and onsets) and whether there were any ongoing disputes (and onsets) the state (or leader) initiated.

Author(s)

Steven V. Miller

References

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. “An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001.” International Studies Quarterly 60(4): 719-730.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
```

```
cow_ddy %>% add_gml_mids()

# keep just the dispute number and Side A/B identifiers
cow_ddy %>% add_gml_mids(keep=c("dispnum","sidea1", "sidea2"))
```

`add_gwcode_to_cow` *Add Gleditsch-Ward state system codes to your data with Correlates of War state codes.*

Description

`add_gwcode_to_cow()` allows you to match, as well as one can, Gleditsch-Ward system membership data with Correlates of War state system membership data.

Usage

```
add_gwcode_to_cow(data)
```

Arguments

`data`	a data frame with appropriate **peacesciencer** attributes

Details

The `data-raw` directory on the project's Github contains more information about the underlying data that assists in merging in these codes.

The user will invariably need to be careful and ask why they want these data included. The issue here is that both have a different composition and the merging process will not (and cannot) be perfect. We can note that a case like Serbia/Yugoslavia is not too difficult to handle (since "Serbia" never overlaps with "Yugoslavia" in the Gleditsch-Ward data and Correlates of War understands Serbia as the predecessor state, dominant state, and successor state to Yugoslavia). However, there is greater weirdness with a case like Yemen/Yemen Arab Republic. The script will *not* create state-year or dyad-year duplicates for the Correlates of War codes. The size of the original data remain unchanged. However, there will be some year duplicates for various Gleditsch-Ward codes (e.g. Yemen, again). Use with care. You can also use the **countrycode** package. Whether you use this function or the **countrycode** package, do *not* do this kind of merging without assessing the output.

Value

`add_gwcode_to_cow()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame that already has Correlates of War state system codes and adds their corollary Gleditsch-Ward codes.

Author(s)

Steven V. Miller

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_gwcode_to_cow()

create_stateyears() %>% add_gwcode_to_cow()
```

`add_igos` *Add Correlates of War international governmental organizations (IGOs) data to dyad-year or state-year data.*

Description

`add_igos()` allows you to add information from the Correlates of War International Governmental Organizations data to dyad-year or state-year data, matching on Correlates of War system codes.

Usage

```
add_igos(data)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed") or a state-year data frame.

Details

The function leans on attributes of the data that are provided by the `create_dyadyear()` or `create_stateyear()` function. Make sure that function (or data created by that function) appear at the top of the proverbial pipe.

Value

`add_igos()` takes a dyad-year data frame or state-year data frame and adds information available from the Correlates of War International Governmental Organizations data. If the data are dyad-year, the function returns the original data with just one additional column for the total number of mutual IGOs for which both members of the dyad are full members. If the data are state-year, the function returns the original data with four additional columns. These are the number of IGOs for which the state is a full member, the number of IGOs for which the state is an associate member, the number of IGOs for which the state is an observer, and the number of IGOs for which the state is involved in any way (i.e. the sum of the other three columns).

Author(s)

Steven V. Miller

References

Pevehouse, Jon C.W., Timothy Nordstron, Roseanne W McManus, and Anne Spencer Jamison. 2020. "Tracking Organizations in the World: The Correlates of War IGO Version 3.0 datasets." *Journal of Peace Research* 57(3): 492-503.

Wallace, Michael, and J. David Singer. 1970. "International Governmental Organization in the Global System, 1815-1964." *International Organization* 24: 239-87.

Examples

```
# just call library(tidyverse) at the top of the pipe
library(magrittr)

cow_ddy %>% add_igos()

create_stateyears() %>% add_igos()
```

`add_lead` *Add (Select) Leader Experience and Attribute Descriptions (LEAD) Data to Leader-Year or Leader-Dyad-Year Data*

Description

`add_lead()` allows you to add some data recorded in the LEAD data to your leader-year or leader-dyad-year data.

Usage

```
add_lead(data, keep)
```

Arguments

`data`	a leader-year or leader-dyad-year data frame
`keep`	an optional parameter, specified as a character vector, about what leader attributes the user wants to return from this function. If `keep` is not specified, everything from the `LEAD` data in this package is returned. Otherwise, the function subsets the `LEAD` data to just what the user wants.

Value

`add_lead()` takes a leader-year or leader-dyad-year data frame and adds some data recorded in the LEAD data to it. For leader-dyad-year data, suffices of "1" and "2" are added to the data to indicate attributes of the first leader (`obsid1`) or the second leader (`obsid2`), respectively.

Author(s)

Steven V. Miller

References

Ellis, Carli Mortenson, Michael C. Horowitz, and Allan C. Stam. 2015. "Introducing the LEAD Data Set." *International Interactions* 41(4): 718–741.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

create_leaderyears() %>% add_lead()

create_leaderyears() %>% add_lead(keep = c("yrsexper"))
```

`add_lwuf` *Add Estimates of Leader Willingness to Use Force to Leader-Year Data*

Description

`add_lwuf()` allows you to add estimates of leader willingness to use force to leader-year data or leader-dyad-year data.

Usage

```
add_lwuf(data, keep)
```

Arguments

`data`	a leader-year or leader dyad-year data frame as generated in **peacesciencer**
`keep`	an optional argument, specified as a character vector, of the variables from the `lwuf` data frame the user wants in their data. See the `lwuf` data and its documentation for more. If the argument is unspecified, the function will return all measures of leader willingness to use force as generated by Carter and Smith.

Details

See `lwuf` for more information, but I'll copy-paste it here too.

The letter published by Carter and Smith (2020) contains more information as to what these thetas refer. The "M1" theta is a variation of the standard Rasch model from the boilerplate information in the LEAD data. The authors consider this to be "theoretically relevant" or "risk-related" as these all refer to conflict or risk-taking. The "M2" theta expands on "M1" by including political orientation and psychological characteristics. "M3" and "M4" expand on "M1" and "M2" by considering all 36 variables in the LEAD data.

The authors construct and include all these measures, though their analyses suggest "M2" is the best-performing measure. You should probably consider using `theta2_mean` as your default estimate of leader willingness to use force in leader-year analyses.

Value

`add_lwuf()` takes a leader-year or leader-dyad-year data frame and adds estimates of leader willingness to use force, as generated by Carter and Smith (2020).

Author(s)

Steven V. Miller

References

Carter, Jeff and Charles E. Smith, Jr. 2020. "A Framework for Measuring Leaders' Willingness to Use Force." *American Political Science Review* 114(4): 1352–1358.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

create_leaderyears() %>% add_lwuf()
```

`add_minimum_distance` *Add minimum distance data to your data frame*

Description

`add_minimum_distance()` allows you to add the minimum distance (in kilometers) to a (dyad-year, leader-year, leader-dyad-year, state-year) data frame. These estimates are recorded in the `cow_mindist` and `gw_mindist` data that come with this package. The data are current as of the end of 2019.

Usage

```
add_minimum_distance(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

The function leans on attributes of the data that are provided by one of the "create" functions in this package (e.g. `create_dyadyear()` or `create_stateyear()`).

Value

`add_minimum_distance()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds the minimum distance between the first state and the second state (in dyad-year data) or the minimum minimum (sic) distance for a given state in a given year.

Author(s)

Steven V. Miller

References

Schvitz, Guy, Luc Girardin, Seraina Ruegger, Nils B. Weidmann, Lars-Erik Cederman, and Kristian Skrede Gleditsch. 2022. "Mapping The International System, 1886-2017: The `CShapes` 2.0 Dataset." *Journal of Conflict Resolution*. 66(1): 144-161.

Weidmann, Nils B. and Kristian Skrede Gleditsch. 2010. "Mapping and Measuring Country Shapes: The `cshapes` Package." *The R Journal* 2(1): 18-24.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
cow_ddy %>% add_minimum_distance()

create_dyadyears(system = "gw") %>% add_minimum_distance()

create_stateyears(system ="gw") %>% add_minimum_distance()
```

add_nmc *Add Correlates of War National Military Capabilities Data*

Description

add_nmc() allows you to add the Correlates of War National Material Capabilities data to your data.

Usage

```
add_nmc(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

Value

add_nmc() takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds information about the national material capabilities for the state or two states in the dyad in a given year. If the data are dyad-year (or leader-dyad-year), the function adds 12 total columns for the first state (i.e. ccode1) and the second state (i.e. ccode2) for all estimates of national military capabilities provided by the Correlates of War project. If the data are state-year (or leader-year), the function returns six additional columns to the original data that contain that same information for a given state in a given year.

Author(s)

Steven V. Miller

References

Singer, J. David, Stuart Bremer, and John Stuckey. (1972). "Capability Distribution, Uncertainty, and Major Power War, 1820-1965." in Bruce Russett (ed) *Peace, War, and Numbers*, Beverly Hills: Sage, 19-48.

Singer, J. David. 1987. "Reconstructing the Correlates of War Dataset on Material Capabilities of States, 1816-1985." *International Interactions* 14(1): 115-32.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_nmc()

create_stateyears() %>% add_nmc()
```

`add_peace_years` *Add Peace Years to Your Conflict Data*

Description

[Superseded]

`add_peace_years()` calculates peace years for your ongoing conflicts. The function works for both dyad-year and state-year data generated in **peacesciencer**. As of the forthcoming v. 0.7.0, `add_peace_years()` will be deprecated for the more generic and versatile `add_spells()`. Users are free to continue with the function, though I recommend it only for more balanced panels (like state-year or dyad-year), and less for imbalanced panels (like leader-years, or leader-dyad-years). As the change in name implies, `add_spells()` will have greater flexibility with both cross-sectional units and time.

Usage

```
add_peace_years(data, pad = FALSE)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed") or state-year data frame
`pad`	an optional parameter, defaults to FALSE. If TRUE, the peace-year calculations fill in cases where panels are unbalanced/have gaps. Think of a state like Germany disappearing for 45 years as illustrative of this.

Details

The function internally uses `sbtscs()` from **stevemisc**. In the interest of full disclosure, `sbtscs()` leans heavily on `btscs()` from **DAMisc**. I optimized some code for performance.

Importantly, the underlying function (`sbtscs()` in **stevemisc**, by way of `btscs()` in **DAMisc**) has important performance issues if you're trying to run it when your event data are sandwiched by observations without any event data. Here's what I mean. Assume you got the full Gleditsch-Ward state-year data from 1816 to 2020 and then added the UCDP armed conflict data to it. If you want the peace-years for this, the function will fail because every year from 1816 to 1945 (along with 2020, as of writing) have no event data. You can force the function to "not fail" by setting `pad = TRUE` as

an argument, but it's not clear this is advisable for this reason. Assume you wanted event data in UCDP for just the extrasystemic onsets. The data start in 1946 and, in 1946, the United Kingdom, Netherlands, and France had extrasystemic conflicts. For *all* years before 1946, the events are imputed as 1 for those countries that had 1s in the first year of observation and everyone else is NA and implicitly assumed to be a zero. For those NAs, the function runs a sequence resulting in some wonky spells in 1946 that are not implied by (the absence of) the data. In fact, none of those are implied by the absence of data before 1946.

The function works just fine if you truncate your temporal domain to reflect the nature of your event data. Basically, if you want to use this function more generally, filter your dyad-year or state-year data to make sure there are no years without any event data recorded (e.g. why would you have a CoW-MID analyses of dyad-years with observations before 1816?). This is less a problem when years with all-NAs succeed (and do not precede) the event data. For example, the UCDP conflict data run from 1946 to 2019 (as of writing). Having 2020 observations in there won't compromise the function output when `pad = TRUE` is included as an argument.

Finally, `add_peace_years()` will only calculate the peace years and will leave the temporal dependence adjustment to the taste of the researcher. Importantly, I do not recommend manually creating splines or square/cube terms because it creates more problems in adjusting for temporal dependence in model predictions. In a regression formula in R, you can specify the Carter and Signorino (2010) approach as `... + gmlmidspell + I(gmlmidspell^2) + I(gmlmidspell^3)` (assuming you ran `add_peace_years()` on a dyad-year data frame including the Gibler-Miller-Little conflict data). The Beck et al. cubic splines approach is `... + splines::bs(gmlmidspell, 4)`. This function includes the spell and three splines (hence the 4 in the command). Either approach makes for easier model predictions, given R's functionality.

Value

`add_peace_years()` takes a dyad-year or state-year data frame and adds peace years for ongoing conflicts. Dyadic conflict data supported include the Correlates of War (CoW) Militarized Interstate Dispute (MID) data set and the Gibler-Miller-Little (GML) corrections to CoW-MID. State-level conflict data supported in this function include the UCDP armed conflict data and the CoW intrastate war data.

Author(s)

Steven V. Miller

References

Armstrong, Dave. 2016. "**DAMisc**: Dave Armstrong's Miscellaneous Functions." *R package version 1.4-3*.

Beck, Nathaniel, Jonathan N. Katz, and Richard Tucker. 1998. "Taking Time Seriously: Time-Series-Cross-Section Analysis with a Binary Dependent Variable." *American Journal of Political Science* 42(4): 1260–1288.

Carter, David B. and Curtis S. Signorino. 2010. "Back to the Future: Modeling Time Dependence in Binary Data." *Political Analysis* 18(3): 271–292.

Miller, Steven V. 2017. "Quickly Create Peace Years for BTSCS Models with `sbtscs` in `stevemisc`." http://svmiller.com/blog/2017/06/quickly-create-peace-years-for-btscs-models-with-stevemisc/

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
cow_ddy %>%
add_gml_mids(keep = NULL) %>%
add_cow_mids(keep = NULL) %>%
add_contiguity() %>%
add_cow_majors() %>%
filter_prd()  %>%
add_peace_years()
```

---

add_rugged_terrain *Add rugged terrain information to a data frame*

Description

add_rugged_terrain() allows you to add information, however crude, about the "ruggedness" of a state's terrain to your (dyad-year, leader-year, leader-dyad-year, state-year) data.

Usage

```
add_rugged_terrain(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

Please see the information for the underlying data rugged, and the associated R script in the data-raw directory, to see how these data are generated. Importantly, these data are time-agnostic and move *slowly*. We're talking about geography here. Both data sets benchmark around 1999-2000 and it's a leap of faith to use these data for comparisons across the entirety of the Correlates of War or Gleditsch-Ward system membership. Every use of data of these types have been either cross-sectional snapshots or for making state-to-state comparisons after World War II (think of your prominent civil war studies here). Be mindful about what you expect to get from these data.

The underlying data have both Gleditsch-Ward codes and Correlates of War codes. The merge it makes depends on what you declare as the "master" system at the top of the pipe (e.g.. in create_dyadyears() or create_stateyears()). If, for example, you run create_stateyears(system="cow") and follow it with add_gwcode_to_cow(), the merge will be on the Correlates of War codes and not the Gleditsch-Ward codes. You can see the script mechanics to see how this is achieved.

Value

add_rugged_terrain() takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame, whether the primary state identifiers are from the Correlates of War system or the Gleditsch-Ward system, and returns information about the "ruggedness" of the state's terrain. The two indicators returned are the "terrain ruggedness index" calculated by Nunn and Puga (2012) and a logarithmic transformation of how mountainous the state is (as calculated by Gibler and Miller, 2014). The dyad-year (leader-dyad-year) data get four additional columns (i.e. both indicators for both states in the dyad) whereas the state-year data get just the two additional columns.

Author(s)

Steven V. Miller

References

Fearon, James D., and David Laitin, "Ethnicity, Insurgency, and Civil War" *American Political Science Review* 97: 75–90.

Gibler, Douglas M. and Steven V. Miller. 2014. "External Territorial Threat, State Capacity, and Civil War." *Journal of Peace Research* 51(5): 634-646.

Nunn, Nathan and Diego Puga. 2012. "Ruggedness: The Blessing of Bad Geography in Africa." *Review of Economics and Statistics*. 94(1): 20-36.

Riley, Shawn J., Stephen D. DeGloria, and Robert Elliot. 1999. "A Terrain Ruggedness Index That Quantifies Topographic Heterogeneity," *Intermountain Journal of Sciences* 5: 23–27.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

cow_ddy %>% add_rugged_terrain()

create_stateyears() %>% add_rugged_terrain()

create_stateyears(system = "gw") %>% add_rugged_terrain()
```

---

add_sdp_gdp *Add (Surplus and Gross) Domestic Product Data*

Description

add_sdp_gdp() allows you to add estimated GDP and "surplus" domestic product data from a 2020 analysis published in *International Studies Quarterly* by Anders, Fariss, and Markowitz.

Usage

```
add_sdp_gdp(data)
```

Arguments

data	a data frame with appropriate **peacesciencer** attributes

Details

The function leans on attributes of the data that are provided by one of the "create" functions. Make sure a recognized function (or data created by that function) appear at the top of the proverbial pipe. Users will also want to note that the underlying function access two different data sets. It appears that the results published in the *International Studies Quarterly* used Correlates of War classification, but a follow-up repository on Github uses Gleditsch-Ward classification. The extent to which these estimates are generated by simulation, it does mean the estimates will be slightly different across both data sets even for common observations (e.g. the United States in 1816).

Because these are large nominal numbers, the estimates have been log-transformed. Users can always exponentiate these if they choose. Researchers can use these data to construct reasonable estimates of surplus GDP per capita, but must exponentiate the underlying variables before doing this.

Be mindful that the data are fundamentally state-year and that extensions to leader-level data should be understood as approximations for leaders in a given state-year.

Value

`add_sdp_gdp()` takes a (dyad-year, leader-year, leader-dyad-year, state-year) data frame and adds information about the estimated gross domestic product (in 2011 USD) for that year, the estimated population in that year, the GDP per capita in that year, and what Anders, Fariss and Markowitz term the "surplus domestic product" in that year. If the data are dyad-year (leader-dyad-year), the function adds eight total columns for the first state (i.e. *ccode1*) and the second state (i.e. *ccode2*) for all these estimates. If the data are state-year (or leader-year), the function returns four additional columns to the original data that contain that same information for a given state in a given year.

Author(s)

Steven V. Miller

References

Anders, Therese, Christopher J. Fariss, and Jonathan N. Markowitz. 2020. "Bread Before Guns or Butter: Introducing Surplus Domestic Product (SDP)" *International Studies Quarterly* 64(2): 392–405.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
```

```
cow_ddy %>% add_sdp_gdp()

create_stateyears() %>% add_sdp_gdp()

create_stateyears(system = "gw") %>% add_sdp_gdp()
```

`add_spells` *Add "Spells" to Data*

Description

`add_spells()` calculates "spells" in your state-year, leader-year, or dyad-year data. The application here is mostly concerned with things like "peace spells" between conflicts in a given cross-sectional unit (e.g. a state or dyad).

Usage

```
add_spells(data, conflict_event_type = "ongoing", ongo = FALSE)
```

Arguments

`data`	an applicable data frame (e.g. leader-year, dyad-year, state-year, as created in **peacesciencer**)
`conflict_event_type`	type of event for which spells should be calculated, either "ongoing" or "onset". Default is "ongoing". If "ongoing", the spells are calculated on the presence of an ongoing event. If "onset", spells are calculated on the onset of a conflict event with successive zeros (if observed) calculated as "peace". See Details section for more.
`ongo`	If TRUE, successive 1s are considered ongoing events and treated as NA after the first 1. If FALSE, successive 1s are all treated as failures. Defaults to FALSE.

Details

The function internally uses `ps_spells()` from **stevemisc**. In the interest of full disclosure, `ps_spells()` leans heavily on `add_duration()` from **spduration**. I optimized some code for performance.

Thinking of an application like peace-years, `add_spells()` will only calculate the peace years and will leave the temporal dependence adjustment to the taste of the researcher. Importantly, I do not recommend manually creating splines or square/cube terms because it creates more problems in adjusting for temporal dependence in model predictions. In a regression formula in R, you can specify the Carter and Signorino (2010) approach as `... + gmlmidspell + I(gmlmidspell^2) + I(gmlmidspell^3)` (assuming you ran `add_spells()` on a dyad-year data frame including the Gibler-Miller-Little conflict data). The Beck et al. cubic splines approach is `... + splines::bs(gmlmidspell, 4)`. This function includes the spell and three splines (hence the 4 in the command). Either approach makes for easier model predictions, given R's functionality.

Thinking of our dyadic analyses of conflict, I've always understood that something like "peace-years" should be calculated on the ongoing event and not the onset of the event. Think of something like the Iran-Iraq War (MID#2115) as illustrative here. The MID (which became a war) started in 1980 and ended in 1988. There are no other bilateral incidents between Iran-Iraq independent of the war, per Correlates of War coding rules. If peace years are calculated at the "onset" of the event, it would list peace-years between the two countries from 1981 to 1988. I've never understood that to make sense, but still I've seen others insist this is the correct way to do it. `add_peace_years()` would force the calculation on the ongoing event, which I still maintain is correct. `add_spells()` will allow you to calculate on onsets, even if ongoing events are the default.

The underlying function for `add_spells()` will stop without a return if there are NAs bracketing observed events. The surest way this will happen is if you're doing something like a dyad-year analysis of inter-state conflicts from 1816 to 2010, but `create_dyadyears()` created observations from 2011 to 2020 for you as well. Remove those before using this function and confine the temporal domain to just those time-units (e.g. years) for which there is observed event data. See what I do in the example below.

Value

`add_spells()` takes a dyad-year, leader-year, or state-year data frame and adds spells for ongoing conflicts. Dyadic conflict data supported include the Correlates of War (CoW) Militarized Interstate Dispute (MID) data set and the Gibler-Miller-Little (GML) corrections to CoW-MID. State-level conflict data supported in this function include the UCDP armed conflict data and the CoW intra-state war data. Leader-year conflict data supported include the GML MID data.

Author(s)

Steven V. Miller

References

Beger, Andreas, Daina Chiba, Daniel W. Hill, Jr, Nils W. Metternich, Shahryar Minhas and Michael D. Ward. 2018. "**spduration**: Split-Population and Duration (Cure) Regression." *R package version 0.17.1*.

Beck, Nathaniel, Jonathan N. Katz, and Richard Tucker. 1998. "Taking Time Seriously: Time-Series-Cross-Section Analysis with a Binary Dependent Variable." *American Journal of Political Science* 42(4): 1260–1288.

Carter, David B. and Curtis S. Signorino. 2010. "Back to the Future: Modeling Time Dependence in Binary Data." *Political Analysis* 18(3): 271–292.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

aaa <- subset(cow_ddy, year <= 2010)

aaa %>%
```

```
add_gml_mids(keep = NULL) %>%
add_cow_mids(keep = NULL) %>%
add_contiguity() %>%
add_cow_majors() %>%
filter_prd()  %>%
add_spells()
```

`add_strategic_rivalries` *Add Thompson and Dreyer's (2012) strategic rivalry data to dyad-year data frame*

Description

`add_strategic_rivalries()` merges in Thompson and Dreyer's (2012) strategic rivalry data to a dyad-year data frame. The right-bound, as of right now, are bound at 2010.

Usage

```
add_strategic_rivalries(data, across_types = 1)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed")
`across_types`	optional, relevant for state-year, takes a value of 1, 2, or 3 to look for whether one of three types fits criteria for ideological, interventionary, positional, spatial rivalry. Defaults to 1.

Details

`add_strategic_rivalries()` will include some other information derived from the rivalry data that the user may not want (e.g. start year of the rivalry). Feel free to select those out after the fact. Function includes an on-the-fly adjustment for Austria for rivalry #79. In this case, the Austria-Serbia rivalry continues for two years after Austria-Hungary (`ccode: 300`) became Austria (`ccode: 305`).

The `across_types` argument is optional and observed for only state-year calls. It defaults to 1. At the default, the function looks into the rivalry data (in `td_rivalries`) and focuses on the `type1` column. If, say, a state has an ongoing rivalry and it is primarily spatial, it codes that as a spatial rivalry. Assume you input `across_types = 2`, the function then looks across both the `type1` and `type2` columns to see if there is a spatial component to the rivalry as either its primary or secondary dimension. If so, it codes that as a 1. `across_types` must be 1, 2, or 3.

Value

add_strategic_rivalries() takes a dyad-year data frame and adds information about ongoing strategic rivalries. It will also include a simple dummy variable for whether there was an ongoing rivalry in the year or not. For state-year data, it returns the count of ongoing strategic rivalries for the state in the year meeting a certain criteria (i.e. whether the state has an interventionary, ideological, positional, or spatial rivalry in an ongoing year, and how many).

Author(s)

Steven V. Miller

References

Miller, Steven V. 2019. "Create and Extend Strategic (International) Rivalry Data in R". URL: http://svmiller.com/blog/2019/10/create-extend-strategic-rivalry-data-r/

Thompson, William R. and David Dreyer. 2012. *Handbook of International Rivalries*. CQ Press.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
cow_ddy %>% add_strategic_rivalries()

# across_types defaults to 1
create_stateyears() %>% add_strategic_rivalries()
```

---

add_ucdp_acd *Add UCDP Armed Conflict Data to state-year data frame*

Description

add_ucdp_acd() allows you to add UCDP Armed Conflict data to a state-year data frame

Usage

```
add_ucdp_acd(data, type, issue, only_wars = FALSE)
```

Arguments

data	state-year data frame
type	the types of armed conflicts the user wants to consider, specified as a character vector. Options include "extrasystemic", "interstate", "intrastate", and "II". "II" is convenience shorthand for "internationalized intrastate". If you want just one (say: "intrastate"), then the type you want in quotes is sufficient. If you want multiple, wrap it in a vector with c().

issue	do you want to subset the data to just different armed conflicts over different types of issues? If so, specify those here as you would with the type argument. Options include "territory", "government", and "both".
only_wars	subsets the conflict data to just those with intensity levels of "war" (i.e. >1,000 deaths). Defaults to FALSE.

Details

Right now, only state-year data are supported. Function is in true pilot mode.

Value

add_ucdp_acd() takes a state-year data frame and returns state-year information from the UCDP Armed Conflict data set (v. 20.1). The variables returned are whether there is an ongoing armed conflict in that year, whether there was an armed conflict episode onset that year, what was the maximum intensity observed that year (if an armed conflict was observed), and a character vector of the associated conflict IDs that year.

Author(s)

Steven V. Miller

References

Gleditsch, Nils Petter; Peter Wallensteen, Mikael Eriksson, Margareta Sollenberg & Havard Strand (2002) Armed Conflict 1946–2001: A New Dataset. *Journal of Peace Research* 39(5): 615–637.

Pettersson, Therese; Stina Hogbladh & Magnus Oberg (2019). Organized violence, 1989-2018 and peace agreements. *Journal of Peace Research* 56(4): 589-603.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
library(dplyr)

create_stateyears(system = "gw") %>%
filter(between(year, 1946, 2019)) %>%
add_ucdp_acd()

create_stateyears(system = "gw") %>%
filter(between(year, 1946, 2019)) %>%
add_ucdp_acd(type = "intrastate", issue = "government")
```

`add_ucdp_onsets` *Add UCDP onsets to state-year data*

Description

`add_ucdp_onsets()` allows you to add information about conflict episode onsets from the UCDP data program to state-year data.

Usage

```
add_ucdp_onsets(data)
```

Arguments

`data`	a state-year data frame

Details

The function leans on attributes of the data that are provided by the `create_dyadyear()` or `create_stateyear()` function. Make sure that function (or data created by that function) appear at the top of the proverbial pipe. The underlying data are version 19.1. Importantly, the UCDP yearly onset data are nominally state-year, but technically state-dyad-episode-year for cases of onsets. For example, there are four France-1946 observations because of four new conflict episodes with Cambodia, Laos, Thailand, and Vietnam. There are two Panama-1989 episodes, one for the invasion by the United States and another for a failed coup attempt. That means the are duplicates in the original data that I process into summaries. The user will probably want to consider some kind of recoding here.

Value

`add_ucdp_onsets()` takes a state-year data frame and adds a few summary variables based off armed conflict onsets data provided by UCDP. The variables returned are the sum of new conflict dyads (should they exist) in a given state-year, and the sum of new onset episodes (or new conflicts) that are separated by one, two, three, five, or 10 years since the last conflict episode.

Author(s)

Steven V. Miller

References

Gleditsch, Nils Petter; Peter Wallensteen, Mikael Eriksson, Margareta Sollenberg & Havard Strand (2002) Armed Conflict 1946–2001: A New Dataset. *Journal of Peace Research* 39(5): 615–637.

Pettersson, Therese; Stina Hogbladh & Magnus Oberg (2019). Organized violence, 1989-2018 and peace agreements. *Journal of Peace Research* 56(4): 589-603.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
library(dplyr)

create_stateyears(system="gw") %>% add_ucdp_onsets()

create_stateyears() %>%
  add_gwcode_to_cow() %>% add_ucdp_onsets()

# Recall, these are summaries. You'll need to post-process to what you want.

create_stateyears(system="gw") %>%
  add_ucdp_onsets() %>%
  mutate(onset = ifelse(sumonset1 > 0, 1, 0))
```

`archigos` *Archigos: A (Subset of a) Dataset on Political Leaders*

Description

These are leader-level data drawn from the Archigos data. Space considerations mean I offer here just a few columns based on these data. Data are version 4.1.

Usage

```
archigos
```

Format

A data frame with 3409 observations on the following 11 variables.

`gwcode` a numeric vector for the Gleditsch-Ward state code

`obsid` a character vector for observation ID

`leadid` the unique leader identifier

`leader` the leader name

`yrborn` the year the leader was born

`gender` a categorical variable for leader gender ("M" for men, "W" for women)

`startdate` a date for the leader start date

`enddate` a date for the leader end date

`entry` a character vector for the leader's entry type

`exit` a character vector for the leader's exit type

`exitcode` a character vector for more information about the leader's exit type

Details

Space considerations mean I can only offer a few columns from the overall data. Archigos data are rich with information. Consult the raw data available on Hein Goeman's website for more.

To best conform with data requirements on CRAN, a few leader names were renamed if they included irregular characters (e.g. umlauts or accents). These leaders, in these particular applications, hav been renamed to "(Juan Orlando) Hernandez" (`HON-2014`), "(Antonio) Saca Gonzalez" (`SAL-2004`), "Julian Trujillo Largacha" (`COL-1878`), "Cesar Gaviria Trujillo" (`COL-1990`), "Gabriel Garcia Moreno" (`ECU-1869`), "Marcos A. Morinigo" (`PAR-1894-1`), "Higinio Morinigo" (`PAR-1940`), "Sebastian Pinera" (`CHL-2010`), "Sauli Niinisto" (`FIN-2012`), "Louis Gerhard De Geer" (`SWD-1876`), "Stefan Lofven" (`SWD-2014`), "Lars Lokke Rasmussen" (`DEN-2009`, `DEN-2015`), and "Fernando de Araujo" (`ETM-2008-1`). None of these names contain these special characters in the data here.

For clarity's sake, I renamed the `ccode` column in the raw data to be `gwcode`. This is because it may deceive the user peeking into the data that these are not Correlates of War state codes, but Gleditsch-Ward state codes.

References

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

`atop_alliance` *Alliance Treaty Obligations and Provisions (ATOP) Project Data (v. 5.0)*

Description

These are directed dyad-year-level data for alliance obligations and provisions from the ATOP project

Usage

```
atop_alliance
```

Format

A data frame with 272,046 observations on the following eight variables.

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

atop_defense a numeric vector that equals 1 if there was an alliance observed with a defense pledge

atop_offense a numeric vector that equals 1 if there was an alliance observed with a offense pledge

atop_neutral a numeric vector that equals 1 if there was an alliance observed with a neutrality pledge

atop_nonagg a numeric vector that equals 1 if there was an alliance observed with a non-aggression pledge

atop_consul a numeric vector that equals 1 if there was an alliance observed with a consultation pledge

Details

The data-raw directory on the project's Github shows how the data were processed.

References

Leeds, Brett Ashley, Jeffrey M. Ritter, Sara McLaughlin Mitchell, and Andrew G. Long. 2002. Alliance Treaty Obligations and Provisions, 1815-1944. *International Interactions* 28: 237-60.

---

ccode_democracy *Democracy data for all Correlates of War states*

Description

These are democracy data for all Correlates of War state system members.

Usage

```
ccode_democracy
```

Format

A data frame with 16536 observations on the following 5 variables.

ccode the Correlates of War system code

year a numeric vector for the year

v2x_polyarchy the Varieties of Democracy "polyarchy" estimate

polity2 the the polity2 score from the Polity project

xm_qudsest an extension of the Unified Democracy Scores (UDS) estimates, made possibly by the QuickUDS package from Xavier Marquez.

Details

Missing data connote data that are unavailable for various reasons. Either there is no democracy data to code or, in the case of the Polity project, the state system member is outright not evaluated for the variable.

The Polity data are from 2017. The Varieties of Democracy data are version 10. Xavier Marquez' `QuickUDS` estimates (i.e. extensions of Pemstein et al. (2010)) come from a package Marquez makes available on his Github (`https://github.com/xmarquez/QuickUDS`).

References

Coppedge, Michael, John Gerring, Carl Henrik Knutsen, Staffan I. Lindberg, Jan Teorell, David Altman, Michael Bernhard, M. Steven Fish, Adam Glynn, Allen Hicken, Anna Luhrmann, Kyle L. Marquardt, Kelly McMann, Pamela Paxton, Daniel Pemstein, Brigitte Seim, Rachel Sigman, Svend-Erik Skaaning, Jeffrey Staton, Agnes Cornell, Lisa Gastaldi, Haakon Gjerlow, Valeriya Mechkova, Johannes von Romer, Aksel Sundtrom, Eitan Tzelgov, Luca Uberti, Yi-ting Wang, Tore Wig, and Daniel Ziblatt. 2020. "V-Dem Codebook v10" Varieties of Democracy (V-Dem) Project.

Marshall, Monty G., Ted Robert Gurr, and Keith Jaggers. 2017. "Polity IV Project: Political Regime Characteristics and Transitions, 1800-2017." Center for Systemic Peace.

Marquez, Xavier, "A Quick Method for Extending the Unified Democracy Scores" (March 23, 2016). doi: 10.2139/ssrn.2753830

Pemstein, Daniel, Stephen Meserve, and James Melton. 2010. "Democratic Compromise: A Latent Variable Analysis of Ten Measures of Regime Type." *Political Analysis* 18(4): 426-449.

`cow_alliance` *Correlates of War directed dyad-year alliance data*

Description

These are version 4.1 of the Correlates of War directed dyad-year alliance data.

Usage

```
cow_alliance
```

Format

A data frame with 120784 observations on the following 7 variables.

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

`cow_defense` a numeric vector that equals 1 if the alliance included a defense pledge

`cow_neutral` a numeric vector that equals 1 if the alliance included a neutrality pledge

`cow_nonagg` a numeric vector that equals 1 if the alliance included a non-aggression pledge

`cow_entente` a numeric vector that equals 1 if the alliance included a pledge to consult if a crisis occurred

Details

The directed dyad-year alliance data are for alliance initiations, not straight dyad-years, "per se." This suggests the presence of duplicate directed dyad-years. For computing ease, given the intended use, I take care of these duplicate dyad-years behind the scenes. Consider the case of the U.S. and Canada in 1958. Therein, there were apparently two separate alliance initiations that included defense pledges. My behind-the-scenes cleaning process groups by `ccode1`, `ccode2`, and `year` and summarizes those alliance pledge variables. I then replace any value greater than 1 with 1. This indicates the presence or absence of a defense pledge in a given directed dyad-year.

References

Gibler, Douglas M. 2009. *International Military Alliances, 1648-2008*. Congressional Quarterly Press.

`cow_capitals` *A complete list of capitals and capital transitions for Correlates of War state system members*

Description

This is a complete list of capitals and capital transitions for Correlates of War state system members. I use it internally for calculating capital-to-capital distances in the `add_capital_distances()` function.

Usage

```
cow_capitals
```

Format

A data frame with 252 observations on the following 7 variables.

`ccode` a numeric vector for the Correlates of War state code

`statenme` a character vector for the state

`capital` a character vector for the name of the capital

`styear` a character vector for the start year. See details section for more information.

`endyear` a character vector for the end year. See details section for more information.

`lat` a numeric vector of the latitude coordinates for the capital

`lng` a numeric vector of the longitude coordinates for the capital

Details

For convenience, the start year for most states is 1816. Samoa, for example, was not a state in 1816. However, the functions that use the `cow_capitals` data will not create observations for states that did not exist at a given point in time.

The data should be current as of the end of 2020.

Cases where a start year is not 1816 indicate a capital transition. For example, Brazil's capital moved from Rio de Janeiro to Brasilia (a planned capital) in 1960. Only 25 states in the data experienced a capital transition. The most recent was Burundi in 2018. Indonesia, as of writing, is planning on a capital transition, but this has not been completed yet.

Kazakhstan renamed its capital for the state leader in 2019. These data retain the name of Astana. This will be changed in the event the software I use records this change.

The capitals data are not without some peculiarities. Prominently, Portugal transferred the Portuguese court from Lisbon to Rio de Janeiro from 1808 to 1821. *This is recorded in the data.* A knowledge of the inter-state conflict data will note there was no war or dispute between, say, Portugal and Spain (or Portugal and any other country) at any point during this time, but it does create some weirdness that would suggest a massive distance between two countries, like Portugal and Spain, that are otherwise land-contiguous.

On Spain: the republican government moved the capital at the start of the civil war (in 1936) to Valencia. However, it abandoned this capital by 1937. I elect to not record this capital transition.

The data also do some (I think) reasonable back-dating of capitals to coincide with states in transition without necessarily formal capitals by the first appearance in the state system membership data. These concern Lithuania, Kazakhstan, and the Philippines. Kaunas is the initial post-independence capital of Lithuania. Almaty is the initial post-independence capital of Kazakhstan. Quezon City is the initial post-independence capital of the Philippines. This concerns, at the most, one or two years for each of these three countries.

`cow_contdir` *Correlates of War Direct Contiguity Data (v. 3.2)*

Description

These contain an abbreviated version of the "master records" for the Correlates of War direct contiguity data. Data contain a few cosmetic changes to assist with some functions downstream from it.

Usage

```
cow_contdir
```

Format

A data frame with 2025840 observations on the following 4 variables.

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`conttype` a numeric vector for the contiguity relationship

`begin` the year-month when this contiguity relationship begins (YYYYMM)

`end` the year-month when this contiguity relationship ends (YYYYMM)

Details

The "master record" provided by the Correlates of War is "non-directed." I make these data "directed" for convenience.

For clarity, the contiguity codes range from 1 to 5. 1 = direct land contiguity. 2 = separated by 12 miles of water or fewer (a la Stannis Baratheon). 3 = separated by 24 miles of water or fewer (but more than 12 miles). 4 = separated by 150 miles of water or fewer (but more than 24 miles). 5 = separated by 400 miles of water or fewer (but more than 150 miles). Cases of separation by more than 400 miles of water are not included in the master record (but are easily discerned based on complete dyad-year data).

References

Stinnett, Douglas M., Jaroslav Tir, Philip Schafer, Paul F. Diehl, and Charles Gochman (2002). "The Correlates of War Project Direct Contiguity Data, Version 3." Conflict Management and Peace Science 19 (2):58-66.

---

cow_ddy *A directed dyad-year data frame of Correlates of War state system members*

Description

This is a complete directed dyad-year data frame of Correlates of War state system members. I offer it here as a shortcut for various other functions when I am working on new additions and don't want to invest time in waiting for `create_dyadyears()` to run.

Usage

```
cow_ddy
```

Format

A data frame with 2063670 observations on the following 3 variables.

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

Details

Data are a quick generation from the `create_dyadyears()` function in this package.

cow_gw_years *Correlates of War and Gleditsch-Ward states, by year*

Description

This is a complete (I believe) data set on Correlates of War states and Gleditsch-Ward states, a byproduct of a `full_join()` between `gw_states` and `cow_states` that leans largely on the state abbreviation variable.

Usage

```
cow_gw_years
```

Format

A data frame with 16936 observations on the following 6 variables.

`gwcode` a Gleditsch-Ward state code

`stateabb` the state abbreviation, which was the greatest source of agreement between both data sets

`gw_statename` the state name as it appears in the Gleditsch-Ward data

`ccode` a Correlates of War state code

`cow_statename` the state name as it appears in the Correlates of War data

`year` a numeric vector for the year

Details

The `data-raw` directory on the project's Github contains more information about how these data were created. I'm going to use it for internal stuff. The workflow is going to treat the Correlates of War state system membership codes as more of the "master" codes, for which the user can add Gleditsch-Ward identifiers as they see fit. Data are extended to 2020, assuming no changes to state system membership for either data set.

cow_igo_ndy *Correlates of War Non-Directed Dyad-Year International Governmental Organizations (IGOs) Data*

Description

This is a non-directed dyad-year version of the Correlates of War IGOs data. I use it internally for merging IGOs data into dyad-year data.

Usage

```
cow_igo_ndy
```

Format

A data frame with 917695 observations on the following 4 variables.

`ccode1` the Correlates of War state system code for the first state

`ccode2` the Correlates of War state system code for the second state

`year` the year

`dyadigos` the sum of mutual IGOs for which each state appears as a full member in a given year

Details

The `data-raw` directory on the project's Github contains additional information about how these data were generated from the otherwise enormous dyad-year IGOs data provided by the Correlates of War project. Given the size of that data, and the size limitations of R packages for CRAN, the data I provide here can only be simpler summaries. If you want specifics, you'll need to consult the underlying raw data provided on the Correlates of War project.

References

Pevehouse, Jon C.W., Timothy Nordstrom, Roseanne W McManus, Anne Spencer Jamison, 2020. "Tracking Organizations in the World: The Correlates of War IGO Version 3.0 datasets", Journal of Peace Research 57(3): 492-503.

Wallace, Michael, and J. David Singer. 1970. "International Governmental Organization in the Global System, 1815-1964." International Organization 24: 239-87.

`cow_igo_sy` *Correlates of War State-Year International Governmental Organizations (IGOs) Data*

Description

This is a state-year version of the Correlates of War IGOs data. I use it internally for merging IGOs data into state-year data.

Usage

```
cow_igo_sy
```

Format

A data frame with 1557 observations on the following 5 variables.

`ccode` the Correlates of War state system code for the state

`year` the year

`sum_igo_full` the sum of IGOs for which the state is a full member in a given year

`sum_igo_associate` the sum of IGOs for which the state is just an associate member in a given year

`sum_igo_observer` the sum of IGOs for which the state is just an observer in a given year

`sum_igo_anytype` the sum of IGOs for which the state is a member of any kind in a given year.

Details

The `data-raw` directory on the project's Github contains additional information about how these data were generated from the otherwise enormous dyad-year IGOs data provided by the Correlates of War project. Given the size of that data, and the size limitations of R packages for CRAN, the data I provide here can only be simpler summaries. If you want specifics, you'll need to consult the underlying raw data provided on the Correlates of War project.

References

Pevehouse, Jon C.W., Timothy Nordstrom, Roseanne W McManus, Anne Spencer Jamison. 2020. "Tracking Organizations in the World: The Correlates of War IGO Version 3.0 datasets", Journal of Peace Research 57(3): 492-503.

Wallace, Michael, and J. David Singer. 1970. "International Governmental Organization in the Global System, 1815-1964." International Organization 24: 239-87.

`cow_majors` *Correlates of War Major Powers Data (1816-2016)*

Description

These are the Correlates of War major powers data.

Usage

```
cow_majors
```

Format

A data frame with 14 observations on the following 8 variables.

`ccode` a numeric vector for the Correlates of War country code

`styear` the start year as a major power

`stmonth` the start month as a major power

`stday` the start day as a major power

`endyear` the end year as a major power

`endmonth` the end month as a major power

`endday` the end day as a major power

`version` a version identifier

Details

Data are provided "as-is" with no additional re-cleaning before inclusion into this data set (beyond eliminating the state abbreviation).

References

Correlates of War Project. 2017. "State System Membership List, v2016." Online, https://correlatesofwar.org/data-sets/state-system-membership

---

`cow_mid_ddydisps` *Directed Dyadic Dispute-Year Data with No Duplicate Dyad-Years (CoW-MID, v. 5.0)*

Description

These are directed dyadic dispute year data derived from the Correlates of War (CoW) Militarized Interstate Dispute (MID) project. Data are from version 5.0. These were whittled to where there is no duplicate dyad-years. Its primary aim here is merging into a dyad-year data frame.

Usage

```
cow_mid_ddydisps
```

Format

A data frame with 10234 observations on the following 25 variables.

`dispnum` a numeric vector for the CoW-MID dispute number

`ccode1` a numeric vector for the focal state in the dyad

`ccode2` a numeric vector for the target state in the dyad

`year` a numeric vector for the dispute-year

`cowmidongoing` a numeric vector for whether there was a dispute ongoing in that year

`cowmidonset` a numeric vector for whether it was the onset of a new dispute (or new participant-entry into a recurring dispute)

`sidea1` is `ccode1` on side A of the dispute?

`sidea2` is `ccode2` on side A of the dispute?

`fatality1` a numeric vector for the overall fatality level of `ccode1` in the dispute

`fatality2` a numeric vector for the overall fatality level of `ccode2` in the dispute

`fatalpre1` a numeric vector for the known fatalities (with precision) for `ccode1` in the dispute

`fatalpre2` a numeric vector for the known fatalities (with precision) for `ccode2` in the dispute

`hiact1` a numeric vector for the highest action of `ccode1` in the dispute

`hiact2` a numeric vector for the highest action of `ccode2` in the dispute

`hostlev1` a numeric vector for the hostility level of `ccode1` in the dispute

`hostlev2` a numeric vector for the hostility level of `ccode2` in the dispute

`orig1` is `ccode1` an originator of the dispute?

`orig2` is `ccode2` an originator of the dispute?

`fatality` a numeric vector for the fatality level of the dispute

`hostlev` a numeric vector for the hostility level of the MID

`mindur` a numeric vector for the minimum duration of the MID

`maxdur` a numeric vector for the maximum duration of the MID

`recip` a numeric vector for whether a MID was reciprocated

`stmon` a numeric vector for the start month of the MID

Details

The process of creating these is described at one of the references below. Importantly, these data are somewhat "naive." That is: they won't tell you, for example, that Brazil and Japan never directly fought each other during World War II. Instead, it will tell you that there were two years of overlap for the two on different sides of the conflict and that the highest action for both was a war. The data are thus similar to what the `EUGene` program would create for users back in the day. Use these data with that limitation in mind.

References

Miller, Steven V. 2021. "How to (Meticulously) Convert Participant-Level Dispute Data to Dyadic Dispute-Year Data in R." URL: http://svmiller.com/blog/2021/05/convert-cow-mid-data-to-dispute-year/

Palmer, Glenn, and Roseanne W. McManus and Vito D'Orazio and Michael R. Kenwick and Mikaela Karstens and Chase Bloch and Nick Dietrich and Kayla Kahn and Kellan Ritter and Michael J. Soules. 2021. "The MID5 Dataset, 2011–2014: Procedures, coding rules, and description" *Conflict Management and Peace Science*.

`cow_mid_dirdisps` *Directed Dyadic Dispute-Year Data (CoW-MID, v. 5.0)*

Description

These are directed dyadic dispute year data derived from the Correlates of War (CoW) Militarized Interstate Dispute (MID) project. Data are from version 5.0.

Usage

```
cow_mid_dirdisps
```

Format

A data frame with 11390 observations on the following 18 variables.

`dispnum` a numeric vector for the CoW-MID dispute number

`ccode1` a numeric vector for the focal state in the dyad

`ccode2` a numeric vector for the target state in the dyad

`year` a numeric vector for the dispute-year

`dispongoing` a numeric vector for whether there was a dispute ongoing in that year

`disponset` a numeric vector for whether it was the onset of a new dispute (or new participant-entry into a recurring dispute)

`sidea1` is `ccode1` on side A of the dispute?

`sidea2` is `ccode2` on side A of the dispute?

`fatality1` a numeric vector for the overall fatality level of `ccode1` in the dispute

`fatality2` a numeric vector for the overall fatality level of `ccode2` in the dispute

`fatalpre1` a numeric vector for the known fatalities (with precision) for `ccode1` in the dispute

`fatalpre2` a numeric vector for the known fatalities (with precision) for `ccode2` in the dispute

`hiact1` a numeric vector for the highest action of `ccode1` in the dispute

`hiact2` a numeric vector for the highest action of `ccode2` in the dispute

`hostlev1` a numeric vector for the hostility level of `ccode1` in the dispute

`hostlev2` a numeric vector for the hostility level of `ccode2` in the dispute

`orig1` is `ccode1` an originator of the dispute?

`orig2` is `ccode2` an originator of the dispute?

Details

The process of creating these is described at one of the references below. Importantly, these data are somewhat "naive." That is: they won't tell you, for example, that Brazil and Japan never directly fought each other during World War II. Instead, it will tell you that there were two years of overlap for the two on different sides of the conflict and that the highest action for both was a war. The data are thus similar to what the `EUGene` program would create for users back in the day. Use these data with that limitation in mind.

References

Miller, Steven V. 2021. "How to (Meticulously) Convert Participant-Level Dispute Data to Dyadic Dispute-Year Data in R." URL: http://svmiller.com/blog/2021/05/convert-cow-mid-data-to-dispute-year/

Palmer, Glenn, and Roseanne W. McManus and Vito D'Orazio and Michael R. Kenwick and Mikaela Karstens and Chase Bloch and Nick Dietrich and Kayla Kahn and Kellan Ritter and Michael J. Soules. 2021. "The MID5 Dataset, 2011–2014: Procedures, coding rules, and description" *Conflict Management and Peace Science*.

`cow_mid_disps` *Abbreviated CoW-MID Dispute-level Data (v. 5.0)*

Description

This is an abbreviated version of the dispute-level CoW-MID data.

Usage

```
cow_mid_disps
```

Format

A data frame with 2436 observations on the following 7 variables.

`dispnum` a numeric vector for the CoW-MID dispute number

`outcome` a numeric vector for the outcome of the MID

`styear` a numeric vector for the start year of the MID

`stmon` a numeric vector for the start month of the MID

`settle` a numeric vector for the how dispute was settled

`fatality` a numeric vector for the fatality level of the dispute

`mindur` a numeric vector for the minimum duration of the MID

`maxdur` a numeric vector for the maximum duration of the MID

`hiact` a numeric vector for the highest action of the MID

`hostlev` a numeric vector for the hostility level of the MID

`recip` a numeric vector for whether a MID was reciprocated

Details

These data are purposely light on information; they're not intended to be used for dispute-level analyses, per se. They're intended to augment the directed dyadic dispute-year data by adding in variables that serve as exclusion rules to whittle the data from dyadic dispute-year to just dyad-year data.

References

Palmer, Glenn, and Roseanne W. McManus and Vito D'Orazio and Michael R. Kenwick and Mikaela Karstens and Chase Bloch and Nick Dietrich and Kayla Kahn and Kellan Ritter and Michael J. Soules. 2021. "The MID5 Dataset, 2011–2014: Procedures, coding rules, and description" *Conflict Management and Peace Science*.

---

cow_mindist — *The Minimum Distance Between States in the Correlates of War System, 1886-2019*

Description

These are non-directed dyad-year data for the minimum distance between states in the Correlates of War state system from 1886 to 2019. The data are generated from the **cshapes** package.

Usage

```
cow_mindist
```

Format

A data frame with 817053 observations on the following 4 variables.

`ccode1` the Correlates of War state system code for the first state

`ccode2` the Correlates of War state system code for the second state

`year` the year

`mindist` the minimum distance between states on Jan. 1 of the year, in kilometers

Details

The data are generated from the **cshapes** package. The package authors purport that the data are generated to be compatible with Correlates of War system codes, but a review I did several years ago for an unrelated project (published in 2017 in *Conflict Management & Peace Science*, which you should cite for all your articles if you're reading this) suggested the output does not seem to perfectly meet that billing. These included oddball cases like Zanzibar, United Arab Republic, Comoros, East Germany, and a few others. Those appear to be fixed in this version.

Data are automatically generated (by default) as directed dyad-years. I elect to make them non-directed for space considerations. Making non-directed dyad-year data into directed dyad-year data isn't too difficult in R. It just looks weird to see the code that does it.

Previous versions of these data were for the minimum distance as of Dec. 31 of the referent year. These are now Jan. 1. Most of the data I prove elsewhere in this package are to be understood as the data as they were at the *start* of the year. This is how I process, for example, the `capitals` data as they get merged in the `add_capital_distance()` function. However, the script that generates these data are set at Jan. 1 of the year and not Dec. 31. Right now, the **cshapes** does not appear to work on my system and I do not know why. Fortunately, the package authors made these data available.

References

Schvitz, Guy, Luc Girardin, Seraina Ruegger, Nils B. Weidmann, Lars-Erik Cederman, and Kristian Skrede Gleditsch. 2022. "Mapping The International System, 1886-2017: The `CShapes` 2.0 Dataset." *Journal of Conflict Resolution*. 66(1): 144-161.

Weidmann, Nils B. and Kristian Skrede Gleditsch. 2010. "Mapping and Measuring Country Shapes: The `cshapes` Package." *The R Journal* 2(1): 18-24

`cow_nmc` *Correlates of War National Military Capabilities Data*

Description

These are version 6.0 of the Correlates of War National Military Capabilities data. Data omit the state abbreviation and version identifier for consideration.

Usage

```
cow_nmc
```

Format

A data frame with 15171 observations on the following 9 variables.

`ccode` a numeric vector for the Correlates of War country code

`year` the year

`milex` an estimate of military expenditures (in thousands). See details section for more.

`milper` an estimate of the size of military personnel (in thousands) for the state

`irst` an estimate of iron and steel production (in thousands of tons)

`pec` an estimate of primary energy consumption (thousands of coal-ton equivalents)

`tpop` an estimate of the total population size of the state (in thousands)

`upop` an estimate of the urban population size of the state (in thousands). See details section for more.

`cinc` The Composite Index of National Capability ("CINC") score. See details section for more.

Details

The user will want to be a little careful with how some of these data are used, beyond the typical caveat about how difficult it is to pin-point how many thousands of coal-tons a state like Baden was producing in the 19th century.

First, military expenditures are denominated in British pounds sterling for observations between 1816 and 1913. The observations from 1914 and beyond are denominated in current United States dollars. This is according to the manual.

Second, urban population size is an estimate based on, well, an estimate of the size of the population living in an area with 100,000 or more people.

Third, the Composite Index of National Capability score is calculated as each state's world share of each of the six composite indicators also included in the data in a given year. It theoretically is bound between 0 and 1. A state with a 1 is 100% responsible for 1) all of the military expenditures in the world, 2) is the only state with a military, 3) does all the iron and steel production, 4) all the world's primary energy consumption, and 5) is the only state in the world with a population and an urban population. Incidentally, the maximum scores observed in the data belong to the United States in 1945.

References

Singer, J. David, Stuart Bremer, and John Stuckey. (1972). "Capability Distribution, Uncertainty, and Major Power War, 1820-1965." in Bruce Russett (ed) Peace, War, and Numbers, Beverly Hills: Sage, 19-48.

Singer, J. David. 1987. "Reconstructing the Correlates of War Dataset on Material Capabilities of States, 1816-1985" International Interactions, 14: 115-32.

cow_sdp_gdp *(Surplus and Gross) Domestic Product for Correlates of War States*

Description

These are state-year level data for surplus and gross domestic product for Correlates of War state system members. Data also include population estimates for per capita standardization.

Usage

```
cow_sdp_gdp
```

Format

A data frame with 27753 observations on the following five variables.

`ccode` a numeric vector for the Correlates of War state code

`year` a numeric vector for the year

`wbgdp2011est` a numeric vector for the estimated natural log of GDP in 2011 USD (log-transformed)

`wbpopest` a numeric vector for the estimated population size (log-transformed)

`sdpest` a numeric vector for the estimated surplus domestic product (log-transformed)

`wbgdppc2011est` a numeric vector for the estimated GDP per capita (log-transformed)

Details

These were extracted from the actual replication files from *International Studies Quarterly*. Because these data are ultimately being simulated, a user can expect some slight differences between the Correlates of War version of these data (which Anders et al. published) and the Gleditsch-Ward version of these data (which appear to be the one the authors will more vigorously support going forward).

Space considerations compel me to round these data to three decimal points. These "economic" data are routinely the biggest in the package, and it's because of the decimal points. The justification for this is these data are estimated/simulated anyways and the information loss is at the 1/1000th decimal point. This procedure basically cuts the size of the data to be less than 25% of its original size. The original simulations are available for remote download if you'd like. Type `?download_extdata()` for more information.

References

Anders, Therese, Christopher J. Fariss, and Jonathan N. Markowitz. 2020. "Bread Before Guns or Butter: Introducing Surplus Domestic Product (SDP)" *International Studies Quarterly* 64(2): 392–405.

cow_states *Correlates of War State System Membership Data (1816-2016)*

Description

These are the Correlates of War state system membership data.

Usage

```
cow_states
```

Format

A data frame with 243 observations on the following 10 variables.

`stateabb` a character vector for the state abbreviation

`ccode` a numeric vector for the Correlates of War country code

`statenme` a character vector for the state name

`styear` the start year in the system

`stmonth` the start month in the system

`stday` the start day in the system

`endyear` the end year in the system

`endmonth` the end month in the system

`endday` the end day in the system

`version` a version identifier

Details

Data are provided "as-is" with no additional re-cleaning before inclusion into this data set.

References

Correlates of War Project. 2017. "State System Membership List, v2016." Online, https://correlatesofwar.org/data-sets/state-system-membership

cow_trade_sy *Correlates of War National Trade Data Set (v. 4.0)*

Description

These are state-year-level data for national trade from the Correlates of War project.

Usage

```
cow_trade_sy
```

Format

A data frame with 14410 observations on the following four variables.

`ccode` the Correlates of War state system code

`year` the year

`imports` total imports of the state in current million USD

`exports` total exports of the state in current million USD

Details

The `data-raw` directory on the project's Github shows how the data were processed.

References

Barbieri, Katherine and Omar M.G. Keshk. 2016. Correlates of War Project Trade Data Set Codebook, Version 4.0. Online: https://correlatesofwar.org

Barbieri, Katherine, Omar M.G. Keshk, and Brian Pollins. 2009. "TRADING DATA: Evaluating Our Assumptions and Coding Rules." *Conflict Management and Peace Science*, 26(5): 471-491.

cow_war_inter *Correlates of War Inter-State War Data (v. 4.0)*

Description

These are a modified version of the inter-state war data from the Correlates of War project. Data are version 4.0. The temporal domain is 1816-2007. Data are functionally directed dyadic war-year.

Usage

```
cow_war_inter
```

Format

A data frame with 1932 observations on the following 15 variables.

`warnum` the Correlates of War war number

`ccode1` the Correlates of War state code for side1

`ccode2` the Correlates of War state code for side2

`year` a numeric vector for the year

`cowinteronset` a dummy variable for whether this is an inter-state war onset (i.e. either the year in `StartYear1` or `StartYear2` in the raw data)

`cowinterongoing` a numeric constant of 1

`sidea1` a numeric vector for the side in the war for `ccode1`, either 1 or 2

`sidea2` a numeric vector for the side in the war for `ccode2`, either 1 or 2

`initiator1` a dummy variable that equals 1 if `ccode1` initiated the war

`initiator2` a dummy variable that equals 1 if `ccode2` initiated the war

`outcome1` the outcome for `ccode1` as numeric vector. Outcomes are 1 (winner), 2 (loser), 3 (compromise/tied), 4 (transformed into another type of war), 5 (ongoing at end of 2007, which is not observed in these data), 6 (stalemate), 7 (conflict continues below severity of war), and 8 (changed sides)

`outcome2` the outcome for `ccode2` as numeric vector. Outcomes are 1 (winner), 2 (loser), 3 (compromise/tied), 4 (transformed into another type of war), 5 (ongoing at end of 2007, which is not observed in these data), 6 (stalemate), 7 (conflict continues below severity of war), and 8 (changed sides)

`batdeath1` the estimated deaths for `ccode1` (-9 = unknown)

`batdeath2` the estimated deaths for `ccode2` (-9 = unknown)

`resume` a dummy variable that equals 1 if this is a conflict resumption episode

Details

See `data-raw` directory for how these data were generated. These data are here if you want it, but I caution against using them as gospel. There are a few problems here. One: -9s proliferate the data for battle deaths on either side, which is unhelpful. There are 10 cases where the sum of battle deaths is exactly 1,000 or 1,001. This is suspicious. The "side" variables are not well-explained—in fact they're not explained at all in the codebook— and this can lead a user astray if they want to interpret them analogous to the `sidea` variables in the Correlates of War Militarized Interstate Dispute data. You probably want to use the initiator variables for this. Further, the war data routinely betray the MID data and the two do not speak well to each other. The language Sarkees and Wayman (2010) use in their book talk about how MIDs "precede" a war or are "associated" with a war, which forgets the war data are supposed to be a subset of the MID data. In one case (Gulf War), they get the associated dispute number wrong and, in one prominent case (War of Bosnian Independence), they argue no MID exists at all (it's actually MID#3557).

References

Sarkees, Meredith Reid, and Frank Wheldon Wayman. 2010. Resort to War: A Data Guide to Inter-State, Extra-State, Intra-State, and Non-State Wars, 1816-2007. Washington DC: CQ Press.

cow_war_intra *Correlates of War Intra-State War Data (v. 4.1)*

Description

These are a modified version of the intra-state war data from the Correlates of War project. Data are version 4.1. The temporal domain is 1816-2007.

Usage

```
cow_war_intra
```

Format

A data frame with 1361 observations on the following 17 variables.

`warnum` the Correlates of War war number

`warname` the Correlates of War war name

`wartype` a character vector for the type of war, either "local issues" or "central control"

`year` a numeric vector for the year

`cowintraonset` a dummy variable for whether this is a civil war onset (i.e. either the year in `StartYear1` or `StartYear2` in the raw data)

`cowintraongoing` a numeric constant of 1

`resume_combat` a dummy variable for whether this is a resumption of a conflict (i.e. `StartYear2` is not -8)

`primary_state` a dummy variable for whether the state is the primary state having the civil war

`ccodea` the Correlates of War state code for the participant on Side A. -8 = not applicable (participant is not a state)

`sidea` the name of the participant on Side A. -8 = not applicable (no additional party on this side)

`ccodeb` the Correlates of War state code for the participant on Side B. -8 = not applicable (participant is not a state)

`sideb` the name of the participant on Side B. -8 = not applicable (no additional party on this side)

`intnl` a dummy variable for if this is an internationalized civil war

`outcome` an unordered-categorical variable for the outcome of the civil war. Values include 1 (Side A wins), 2 (Side B wins), 3 (Compromise), 4 (war transformed into another type of war), 5 (war is ongoing at the end of 2007), 6 (stalemate), 7 (conflict continues below severity of war)

`sideadeaths` the estimated deaths for the Side A participant (-9 = unknown, -8 = not applicable)

`sidebdeaths` the estimated deaths for the Side B participant (-9 = unknown, -8 = not applicable)

`ongo2007` a dummy variable for if this war is ongoing as of the end of 2007

Details

See data-raw directory for how these data were generated. In the Guinnea-Bissau Civil War (1998, 1999), the "Mane Junta" have the accented-e scrubbed to coincide with CRAN's character requirements.

References

Dixon, Jeffrey, and Meredith Sarkees. 2016. A Guide to Intra-State Wars: An Examination of Civil Wars, 1816-2014. Thousand Oaks, CA: Sage.

Sarkees, Meredith Reid, and Frank Wheldon Wayman. 2010. Resort to War: A Data Guide to Inter-State, Extra-State, Intra-State, and Non-State Wars, 1816-2007. Washington DC: CQ Press.

---

create_dyadyears *Create dyad-years from state system membership data*

Description

create_dyadyears() allows you to dyad-year data from either the Correlates of War (CoW) state system membership data or the Gleditsch-Ward (gw) system membership data. The function leans on internal data provided in the package.

Usage

```
create_dyadyears(system = "cow", mry = TRUE, directed = TRUE, subset_years)
```

Arguments

system	a character specifying whether the user wants Correlates of War state-years ("cow") or Gleditsch-Ward ("gw") state-years. Correlates of War is the default.
mry	optional, defaults to TRUE. If TRUE, the function extends the script beyond the most recent system membership updates to include observation to the most recently concluded calendar year. For example, the Gleditsch-Ward data extend to the end of 2017. When mry == TRUE, the function returns more recent years (e.g. 2018, 2019) under the assumption that states alive at the end of 2017 are still alive today. Use with some care.
directed	optional, defaults to TRUE. If TRUE, the function returns so-called "directed" dyad-year data. In directed dyad-year data, France-Germany (220-255) and Germany-France (255-220) are observationally different. If FALSE, the function returns non-directed data. In non-directed data, France-Germany and Germany-France in the same year are the same observation. The standard here is to drop cases where the country code for the second observation is less than the country code for the first observation.
subset_years	and optional character vector for subsetting the years returned to just some temporal domain of interest to the user. For example, c(1816:1820) would subset the data to just all dyad-years in 1816, 1817, 1818, 1819, and 1820. Be advised that it's easiest to subset the data after the full universe of dyad-year data have

been created. This means you could, if you choose, effectively overwrite `mry = TRUE` with this argument since the `mry` argument is applied at the expansion of the state system data, which occurs at the start of the function.

Value

`create_dyadyears()` takes state system membership data provided by either Correlates of War or Gleditsch-Ward and returns a dyad-year data frame.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2019. "Create Country-Year and (Non)-Directed Dyad-Year Data With Just a Few Lines in R" http://svmiller.com/blog/2019/01/create-country-year-dyad-year-from-country-data/

Examples

```
# CoW is default, will include years beyond 2016 (most recent CoW update)
create_dyadyears()

# Gleditsch-Ward, include most recent years
create_dyadyears(system="gw")

# Gleditsch-Ward, don't include most recent years
create_dyadyears(system="gw", mry=FALSE)

# Gleditsch-Ward, don't include most recent years, directed = FALSE
create_dyadyears(system="gw", mry=FALSE, directed = FALSE)
```

---

create_leaderdays *Create leader-days from leader data*

Description

`create_leaderdays()` allows you to generate leader-day data from leader-level data provided in **peacesciencer**.

Usage

```
create_leaderdays(system = "archigos", standardize = "none")
```

Arguments

system	a leader system with which to create leader-days. Right now, only "archigos" is supported.
standardize	a character vector of length one: "cow", "gw", or "none". If "cow", the function standardizes the leader-days to just those that overlap with state system membership in the Correlates of War state system (see: cow_states). If "gw", the function standardizes the leader-days to just those that overlap with the state system dates of the Gleditsch-Ward date (see: gw_states). If "none", the function returns all leader-days as presented in Archigos (which is nominally denominated in Gleditsch-Ward state system codes, if not necessarily Gleditsch-Ward state system dates). Default is "none".

Details

create_leaderdays(), as of writing, only supports the Archigos data set of leaders. I envision this function being mostly for internal uses. Basically, create_leaderyears() effectively starts by first running a version of create_leaderdays(). So, why not have this function too?

Value

create_leaderdays() takes leader-level data available in **peacesciencer** and returns a leader-day-level data frame.

Author(s)

Steven V. Miller

References

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

Examples

```
create_leaderdays()

create_leaderdays(standardize = "gw")
```

`create_leaderdyadyears` *Create leader-dyad-years from the Archigos data*

Description

`create_leaderdyadyears()` allows you to created leader dyad-year data from the Archigos data first introduced and described by Goemans et al. (2009).

Usage

```
create_leaderdyadyears(directed = TRUE, system = "gw")
```

Arguments

`directed`	optional, defaults to TRUE. If TRUE, the function returns so-called "directed" leader dyad-year data. If FALSE, the function returns non-directed data where the state codes for the second leader are all greater than the state codes for the second leader.
`system`	a character specifying whether the user wants Correlates of War state-years ("cow") or Gleditsch-Ward ("gw") state-years. Gleditsch-Ward is the default.

Details

This is a *complete* and *universal* leader dyad-year data frame for all possible dyadic leader pairings from 1870 to 2015. This has several implications. First: these data are enormous. The output is over 2 million rows long! Second: the time required to create these data from scratch would take too long for a normal function call. This amounts to an unholy combination of data that are too large for CRAN's disk space restrictions (5 MB) and too time-consuming to do from scratch every time. Thus, the data are pre-generated and stored remotely. Check `download_extdata()` for more information.

Value

`create_leaderdyadyears()` takes remote data available for separate download and returns a complete leader dyad-year data frame for all leaders, and all possible dyads, from 1870 to 2015.

Author(s)

Steven V. Miller

References

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

Examples

```
## Not run:
# download_extdata()
# ^ make sure you've run this first.
# default is directed
create_leaderdyadyears()

# non-directed
create_leaderdyadyears(directed = FALSE)

## End(Not run)
```

`create_leaderyears` *Create leader-years from leader data*

Description

`create_leaderyears()` allows you to generate leader-year data from leader-level data provided in **peacesciencer**

Usage

```
create_leaderyears(system = "archigos", standardize = "none", subset_years)
```

Arguments

`system`	a leader system with which to create leader-years. Right now, only "archigos" is supported.
`standardize`	a character vector of length one: "cow", "gw", or "none". If "cow", the function standardizes the leader-years to just those that overlap with state system membership in the Correlates of War state system (see: `cow_states`). If "gw", the function standardizes the leader-years to just those that overlap with the state system dates of the Gleditsch-Ward date (see: `gw_states`). If "none", the function returns all leader-years as presented in Archigos (which is nominally denominated in Gleditsch-Ward state system codes, if not necessarily Gleditsch-Ward state system dates). Default is "none".
`subset_years`	and optional character vector for subsetting the years returned to just some temporal domain of interest to the user. For example, `c(2000:2005)` would subset the data to just all leader-years in 2000, 2001, 2002, 2003, 2004, and 2005 Be advised that it's easiest to subset the data after the full universe of leader-year data have been created. It is also agnostic about what was supplied to the `standardize` argument.

Details

`create_leaderyears()`, as of writing, only supports the Archigos data set of leaders.

Many leader ages are known with precision. Many are not recorded in the Archigos data. Knowing well that years are aggregates of days, the leader age variable that gets returned in this output should be treated as an approximation of the leader's age.

Be mindful that leader tenure is calculated *before* any standardization argument. Archigos has some leader entries that precede the state system entry for the state, or otherwise do not coincide with state system dates. For example, Lynden Pindling was in his seventh year as leader of The Bahamas (in various titles) before independence in 1973 (in which he became prime minister). Leader tenure is not tethered to state system dates in situations like this (only the dates recorded in the Archigos data).

The leader tenure variable returned here does have the odd effect of potentially misstating leader tenure, or at least making it seem unusual. For example, Jimmy Carter (`USA-1877`) was president in 1977 (year 1), 1978 (year 2), 1979 (year 3), 1980 (year 4), and exited in January 1981 (year 5). Again: years are aggregates of days and it's not evident how else this information should be perfectly communicated with that in mind. Users with some R skills can extract the underlying information from the `archigos` data and, perhaps, calculate something like the maximum leader tenure (in days) on either Dec. 31 of the referent year, or leader exit before Dec. 31 that year, or something to that effect. No matter, I think this to at least be a defensible variable to present to the user with those limitations in mind. If the user is interested in leader tenure in a leader-year analysis, this variable should be fine. If the user is interested in something like the effect of a fifth year on some kind of leader behavior, they will want to figure out something else.

Value

`create_leaderyears()` takes leader-level data available in **peacesciencer** and returns a leader-year-level data frame. This minimal output contains the observation ID from Archigos, the year, the state code for the leader (i.e. either Correlates of War or Gleditsch-Ward, depending on the `standardize` argument), the leader's name in Archigos (if it may help the reader to have that), an approximation of the leader's age, and the year in office for the leader (as a running count, starting at 1).

Author(s)

Steven V. Miller

References

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

Examples

```
# standardize = 'none' is default
create_leaderyears()

create_leaderyears(standardize = 'gw')
```

`create_statedays` *Create state-days from state system membership data*

Description

`create_statedays()` allows you to create state-day data from either the Correlates of War (`CoW`) state system membership data or the Gleditsch-Ward (`gw`) system membership data. The function leans on internal data provided in the package.

Usage

```
create_statedays(system = "cow", mry = TRUE)
```

Arguments

`system`	a character specifying whether the user wants Correlates of War state-years ("cow") or Gleditsch-Ward ("gw") state-years. Correlates of War is the default.
`mry`	optional, defaults to TRUE. If TRUE, the function extends the script beyond the most recent system membership updates to include observation to the most recently concluded calendar year. For example, the Gleditsch-Ward data extend to the end of 2017. When `mry == TRUE`, the function returns more recent years (e.g. 2018, 2019) under the assumption that states alive at the end of 2017 are still alive today. Use with some care.

Value

`create_statedays()` takes state system membership data provided by either Correlates of War or Gleditsch-Ward and returns a simple state-day data frame.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2019. "Create Country-Year and (Non)-Directed Dyad-Year Data With Just a Few Lines in R" http://svmiller.com/blog/2019/01/create-country-year-dyad-year-from-country-data/

Examples

```
# CoW is default, will include years beyond 2016 (most recent CoW update)
create_statedays()

# Gleditsch-Ward, include most recent years
create_statedays(system="gw")

# Gleditsch-Ward, don't include most recent years
create_statedays(system="gw", mry=FALSE)
```

---

create_stateyears *Create state-years from state system membership data*

Description

`create_stateyears()` allows you to generate state-year data from either the Correlates of War (`CoW`) state system membership data or the Gleditsch-Ward (`gw`) system membership data. The function leans on internal data provided in the package.

Usage

```
create_stateyears(system = "cow", mry = TRUE, subset_years)
```

Arguments

`system`	a character specifying whether the user wants Correlates of War state-years ("cow") or Gleditsch-Ward ("gw") state-years. Correlates of War is the default.
`mry`	optional, defaults to TRUE. If TRUE, the function extends the script beyond the most recent system membership updates to include observation to the most recently concluded calendar year. For example, the Gleditsch-Ward data extend to the end of 2017. When `mry == TRUE`, the function returns more recent years (e.g. 2018, 2019) under the assumption that states alive at the end of 2017 are still alive today. Use with some care.
`subset_years`	and optional character vector for subsetting the years returned to just some temporal domain of interest to the user. For example, `c(1816:1820)` would subset the data to just all state-years in 1816, 1817, 1818, 1819, and 1820. Be advised that it's easiest to subset the data after the full universe of state-year data have been created. This means you could, if you choose, effectively overwrite `mry = TRUE` with this argument since the `mry` argument is applied at the expansion of the state system data into state-year data.

Value

`create_stateyears()` takes state system membership data provided by either Correlates of War or Gleditsch-Ward and returns a simple state-year data frame.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2019. "Create Country-Year and (Non)-Directed Dyad-Year Data With Just a Few Lines in R" http://svmiller.com/blog/2019/01/create-country-year-dyad-year-from-country-data/

Examples

```
# CoW is default, will include years beyond 2016 (most recent CoW update)
create_stateyears()

# Gleditsch-Ward, include most recent years
create_stateyears(system="gw")

# Gleditsch-Ward, don't include most recent years
create_stateyears(system="gw", mry=FALSE)
```

---

creg *Composition of Religious and Ethnic Groups (CREG) Fractionalization/Polarization Estimates*

Description

This is a data set with state-year estimates for ethnic and religious fractionalization/polarization, by way of the Composition of Religious and Ethnic Groups (CREG) project at the University of Illinois. I-L-L.

Usage

```
creg
```

Format

A data frame with 11523 observations on the following 9 variables.

`ccode` a Correlates of War state code

`gwcode` a Gleditsch-Ward state code

`creg_ccode` a numeric code for the state, mostly patterned off Correlates of War codes but with important differences. See details section for more.

`year` the year

`ethfrac` an estimate of the ethnic fractionalization index. See details for more.

`ethpol` an estimate of the ethnic polarization index. See details for more.

`relfrac` an estimate of the religious fractionalization index. See details for more.

`relpol` an estimate of the religious polarization index. See details for more.

Details

The `data-raw` directory on the project's Github contains more information about how these data were created. Pay careful attention to how I assigned CoW/G-W codes. The underlying data are version 1.02.

The state codes provided by the CREG project are mostly Correlates of War codes, but with some differences. Summarizing these differences: the state code for Serbia from 1992 to 2013 is actually the Gleditsch-Ward code (340). Russia after the dissolution of the Soviet Union (1991-onward) is 393 and not 365. The Soviet Union has the 365 code. Yugoslavia has the 345 code. The code for Yemen (678) is effectively the Gleditsch-Ward code because it spans the entire post-World War II temporal domain. Likewise, the code for post-unification Germany is the Gleditsch-Ward code (260) as well. The codebook actually says it's 265 (which would be East Germany's code), but this is assuredly a typo based on the data.

The codebook cautions there are insufficient data for ethnic group estimates for Cameroon, France, India, Kosovo, Montenegro, Mozambique, and Papua New Guinea. The French case is particularly disappointing but the missing data there are a function of both France's constitution and modelling issues for CREG (per the codebook). There are insufficient data to make religious group estimates for China, North Korea, and the short-lived Republic of Vietnam.

The fractionalization estimates are the familiar Herfindahl-Hirschman concentration index. The polarization formula comes by way of Montalvo and Reynal-Querol (2000), though this book does not appear to be published beyond its placement online. I recommend Montalvo and Reynal-Querol (2005) instead. You can cite Alesina (2003) for the fractionalization measure if you'd like.

In the most literal sense of "1", the group proportions may not sum to exactly 1 because of rounding in the data. There were only two problem cases in these data worth mentioning. First, in both data sets, there would be the occasional duplicates of group names by state-year (for example: Afghanistan in 1951 in the ethnic group data and the United States in 1948 in the religious group data). In those cases, the script I make available in the `data-raw` directory just select distinct values and that effectively fixes the problem of duplicates, where they do appear. Finally, Costa Rica had a curious problem for most years in the religious group data. All Costa Rica years have group data for Protestants, Roman Catholics, and "others." Up until 1964 or so, the "others" are zero. Afterward, there is some small proportion of "others". However, the sum of Protestants, Roman Catholics, and "others" exceeds 1 (pretty clearly) and the difference between the sum and 1 is entirely the "others." So, I drop the "others" for all years. I don't think that's terribly problematic, but it's worth saying that's what I did.

References

Alesina, Alberto, Arnaud Devleeschauwer, William Easterly, Sergio Kurlat and Romain Wacziarg. 2003. "Fractionalization". *Journal of Economic Growth* 8: 155-194.

Montalvo, Jose G. and Marta Reynal-Querol. 2005. "Ethnic Polarization, Potential Conflict, and Civil Wars" *American Economic Review* 95(3): 796–816.

Nardulli, Peter F., Cara J. Wong, Ajay Singh, Buddy Petyon, and Joseph Bajjalieh. 2012. *The Composition of Religious and Ethnic Groups (CREG) Project*. Cline Center for Democracy.

`declare_attributes` *Declare* **peacesciencer**-*specific attributes to data*

Description

`declare_attributes()` allows the user to declare **peacesciencer**-specific attributes to data they bring from outside the package. This allows the user to use package functions as shortcuts, where appropriate.

Usage

```
declare_attributes(data, data_type, system, conflict_type)
```

Arguments

`data`	a data frame for which you want **peacesciencer**-specific attributes
`data_type`	optional, but a character vector of length 1 coinciding with the type of data the user believes the data frame is. Options include: 'dyad_year', 'leader_day', 'leader_year', 'leader_dyad_year', 'state_day', or 'state_year'.
`system`	optional, but a character vector of length 1 coinciding with the state system of the data. If specified at all, must be 'cow' or 'gw'.
`conflict_type`	optional, and applicable to just conflict data and the "whittle" class functions in **peacesciencer**. If specified, must be a character vector of length 1 that is either 'cow' or 'gml'.

Details

The function's documentation will include what attributes are available to be declared. No doubt, the list of potential attributes will grow in time, but the attributes that can be declared are limited to just what I've built into the package to this point. Users cannot declare more than one attribute of a given type (i.e. a user cannot declare the system to be both Correlates of War and Gleditsch-Ward).

The idea here is, basically, to allow the user to use functions in **peacesciencer** for data they have created or have acquired from elsewhere. However, this functions provides *no* assurances about quality control in the various merges built elsewhere into this package. This package aggressively tests functions for data generated in-house. If your outside data have merges, the various "add" functions may not perfectly perform. There is no real way I can control for this since the data are coming from outside the package and not through one of the "create" functions. In your particular case, that may not be much of a problem. However, it's the user's responsibility to do their own quality control in this situation.

Value

`declare_attributes()` takes a data frame and adds **peacesciencer**-specific attributes to the data frame. This will allow the user to take advantage of many of the functions in this package without starting the process with one of the "create" functions. If nothing is declared in the function, no attribute is added and the function just returns the original data without any change.

Author(s)

Steven V. Miller

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

data.frame(ccode = 2, year = c(1816:1830)) -> usa_years

usa_years %>% declare_attributes(data_type = 'state_year', system = 'cow')
```

---

download_extdata *Download Some Extra Data for Peace Science Research*

Description

`download_extdata()` leverages R's `inst` directory flexibility to allow you to download some extra data and store it in the package.

Usage

```
download_extdata(overwrite = FALSE)
```

Arguments

`overwrite`	logical, defaults to FALSE. If FALSE, the function checks to see if you've already downloaded the data and, if you already have, it does nothing. If TRUE, the function redownloads the data.

Value

`download_extdata()` downloads some extra data stored on my website (http://svmiller.com) and sticks them in the *extdata* directory in the package.

A Description of Various Data Sets This Will Download

Running `download_extdata()` returns the following data that will be stored in the package's `extdata` directory.

Correlates of War Dyadic Trade Data Set (v. 4.0):

These are directed dyad-year-level data for dyadic trade from the Correlates of War project. The trade values presented here have been rounded to three decimal points to conserve space. The data downloaded by this function are about 4.1 megabytes in size.

COLUMN	DESCRIPTION
`ccode1`	a numeric vector for the Correlates of War state code for the first state
`ccode2`	a numeric vector for the Correlates of War state code for the second state
`year`	the year
`flow1`	imports of `ccode1` from `ccode2`, in current million USD
`flow2`	imports of `ccode2` from `ccode1`, in current million USD
`smoothflow1`	smoothed `flow1` values
`smoothflow2`	smoothed `flow2` values

Directed Leader Dyad-Year Data, 1870-2015 (CoW States):

These are all directed leader dyad-year data from 1870-2015. Data come from the Archigos data (version 4.1). The data are standardized to just those observations where both leaders and states appear in the CoW state system data. The data downloaded by this function are about 2 megabytes in size.

COLUMN	DESCRIPTION
`year`	the year
`obsid1`	the unique Archigos (v. 4.1) observation ID for the first leader
`obsid2`	the unique Archigos (v. 4.1) observation ID for the second leader
`ccode1`	a numeric vector for the Correlates of War state code for the first state
`ccode2`	a numeric vector for the Correlates of War state code for the second state
`gender1`	the gender of `obsid1` ("M" or "F")
`gender2`	the gender of `obsid2` ("M" or "F")
`leaderage1`	the approximate age (i.e. `year - yrborn`) for `obsid1` in the year
`leaderage2`	the approximate age (i.e. `year - yrborn`) for `obsid2` in the year
`yrinoffice1`	a running count for the tenure of `obsid1`, starting at 1.
`yrinoffice2`	a running count for the tenure of `obsid2`, starting at 1.

Directed Leader Dyad-Year Data, 1870-2015 (Gleditsch-Ward States):

These are all directed leader dyad-year data from 1870-2015. Data come from the Archigos data (version 4.1). The data represent every possible dyadic leader-pairing in the Archigos data (which is denominated in the Gleditsch-Ward system), but standardizes leader dyad-years to Gleditsch-Ward state system dates. The data downloaded by this function are about 2.2 megabytes in size.

COLUMN	DESCRIPTION
`year`	the year
`obsid1`	the unique Archigos (v. 4.1) observation ID for the first leader
`obsid2`	the unique Archigos (v. 4.1) observation ID for the second leader

gwcode1	a numeric vector for the Gleditsch-Ward state code for the first state
gwcode2	a numeric vector for the Gleditsch-Ward state code for the second state
gender1	the gender of obsid1 ("M" or "F")
gender2	the gender of obsid2 ("M" or "F")
leaderage1	the approximate age (i.e. year - yrborn) for obsid1 in the year
leaderage2	the approximate age (i.e. year - yrborn) for obsid2 in the year
yrinoffice1	a running count for the tenure of obsid1, starting at 1.
yrinoffice2	a running count for the tenure of obsid2, starting at 1.

Chance-Corrected Measures of Foreign Policy Similarity (FPSIM, v. 2):

The FPSIM data set provides measures of foreign policy similarity of dyads based on alliance ties (Correlates of War, version 4.1) and UN General Assembly voting (Voeten, version 17) for all members of the Correlates of War state system. The alliance data cover the time period from 1816 to 2012, and the UN voting data from 1946 to 2015. The similarity measures include various versions of Ritter and Signorino's *S* (weighted/non-weighted by material capabilities; squared/absolute distance metrics) as well as the chance-corrected measures Cohen's (1960) kappa and Scott's (1955) pi. The measures based on alliance data come in two versions: one is based on valued alliance ties and the other is based on binary alliance ties. Data were last updated on December 7, 2017, and this description was effectively plagiarized (with his blessing) from Frank Haege's Dataverse.

These data are directed dyad-years with 17 columns and 1,872,198 observations. They will almost certainly be the largest data set I nudge/ask you to download remotely. The file containing this information is 18.6 MB in size. To reduce size further, these decimal points have also been rounded to three spots.

Haege generated all estimates of dyadic foreign policy similarity, except for the taub column. That was generated separately, by me.

COLUMN	DESCRIPTION
year	the year
ccode1	the Correlates of War state code for the first state
ccode2	the Correlates of War state code for the second state
taub	Tau-b (valued alliance data)
srsvas	unweighted *S* (squared distances, valued alliance data)
srswvas	weighted *S* (squared distances, valued alliance data)
srsvaa	unweighted *S* (absolute distances, valued alliance data)
srswvaa	weighted *S* (absolute distances, valued alliance data)
kappava	Kappa (squared distances, valued alliance data)
piva	Pi (squared distances, valued alliance data)
srsba	Unweighted *S* (binary alliance data)
srswba	Weighted *S* (binary alliance data)
kappaba	Kappa (binary alliance data)
piba	Pi denominator (binary alliance data)
srsvvs	Unweighted *S* (squared distances, valued UN voting data)
srsvva	Unweighted *S* (absolute distances, valued UN voting data)
kappavv	Kappa (squared distances, valued UN voting data)
pivv	Pi (squared distances, valued UN voting data)

Author(s)

Steven V. Miller

References

Barbieri, Katherine, Omar M. G. Keshk, and Brian Pollins. 2009. "TRADING DATA: Evaluating our Assumptions and Coding Rules." *Conflict Management and Peace Science*. 26(5): 471-491.

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

Haege, Frank. 2011. "Choice or Circumstance? Adjusting Measures of Foreign Policy Similarity for Chance Agreement." *Political Analysis* 19(3): 287-305.

Examples

```
## Not run:
# Here's where the data are going to be downloaded.
system.file("extdata", package="peacesciencer")
# Now, let's download the data.
download_extdata()

## End(Not run)
```

---

false_cow_dyads *False Correlates of War Directed Dyad-Years*

Description

This is a simple data set that communicates directed dyads in the Correlates of War data that appear in the same year, but not in any particular day in the year. They are used in an anti-join in the `create_dyadyears()` function in this package.

Usage

```
false_cow_dyads
```

Format

A data frame the following four variables.

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

`in_ps` a constant that equals 1 if these data would appear in `create_dyadyears()` if you were not careful to remove them.

Details

Think of the directed Suriname and Republic of Vietnam dyad here as illustrative here. The Republic of Vietnam exits the Correlates of War state system on April 30, 1975 whereas Suriname enters the state system on November 25, 1975. Both appear in the same year, but not at the same time.

`false_gw_dyads` *False Gleditsch-Ward Directed Dyad-Years*

Description

This is a simple data set that communicates directed dyads in the Gleditsch-Ward data that appear in the same year, but not in any particular day in the year. They are used in an anti-join in the `create_dyadyears()` function in this package.

Usage

```
false_gw_dyads
```

Format

A data frame the following four variables.

`gwcode1` a numeric vector for the Gleditsch-Ward state code for the first state

`gwcode2` a numeric vector for the Gleditsch-Ward state code for the second state

`year` a numeric vector for the year

`in_ps` a constant that equals 1 if these data would appear in `create_dyadyears()` if you were not careful to remove them.

Details

Think of the directed Suriname and Republic of Vietnam dyad here as illustrative here. The Republic of Vietnam exits the Correlates of War state system on April 30, 1975 whereas Suriname enters the state system on November 25, 1975. Both appear in the same year, but not at the same time.

`filter_prd` *Filter dyad-year data to include just politically relevant dyads*

Description

`filter_prd()` filters a dyad-year data frame to just those that are "politically relevant." This is useful for discarding unnecessary (and unwanted) observations that just consume space in memory.

Usage

```
filter_prd(data)
```

Arguments

`data`	a dyad-year data frame (either "directed" or "non-directed")

Details

"Political relevance" can be calculated a few ways. Right now, the function considers only "direct" contiguity and Correlates of War major power status. You can employ maximalist definitions of "direct contiguity" to focus on just the land-contiguous. This function is inclusive of any type of contiguity relationship.

As of version 0.5, `filter_prd()` is a shortcut for `add_contiguity()` and/or `add_cow_majors()` if the function is executed in the absence of the data needed to create politically relevant dyads. See the example below for what this means.

Value

`filter_prd()` takes a dyad-year data frame, assuming it has columns for major power status and contiguity type, calculates whether the dyad is "politically relevant", and subsets the data frame to just those observations.

Author(s)

Steven V. Miller

References

Weede, Erich. 1976. "Overwhelming preponderance as a pacifying condition among contiguous Asian dyads." *Journal of Conflict Resolution* 20: 395-411.

Lemke, Douglas and William Reed. 2001. "The Relevance of Politically Relevant Dyads." *Journal of Conflict Resolution* 45(1): 126-144.

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

A <- cow_ddy %>% add_contiguity() %>% add_cow_majors() %>% filter_prd()

A

# you can also use it as a shortcut for the other functions required
# to calculate politically relevant dyads.
B <- cow_ddy %>% filter_prd()

B

identical(A,B)
```

`gml_dirdisp` *Directed dispute-year data (Gibler, Miller, and Little, 2016)*

Description

These are directed dispute-year data from the most recent version (2.2.1) of the Gibler-Miller-Little (GML) militarized interstate dispute (MID) data. They are used internally for merging into full dyad-year data frames.

Usage

```
gml_dirdisp
```

Format

A data frame with 10330 observations on the following 39 variables.

`dispnum` the dispute number

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

`midongoing` a constant of 1 for ongoing disputes

`midonset` a numeric vector that equals 1 for the onset year of a given dispute

`sidea1` is the first state (in `ccode1`) on the side that took the first militarized action?

`sidea2` is the second state (in `ccode2`) on the side that took the first militarized action?

revstate1 is the first state (in ccode1) a revisionist state in the dispute?

revstate2 is the second state (in ccode2) a revisionist state in the dispute?

revtype11 what is the revtype1 value for ccode1?

revtype12 what is the revtype1 value for ccode2?

revtype21 what is the revtype2 value for ccode1?

revtype22 what is the revtype2 value for ccode2?

fatality1 what is the fatality value for ccode1?

fatality2 what is the fatality value for ccode2?

fatalpre1 what is the fatalpre value for ccode1?

fatalpre2 what is the fatalpre value for ccode2?

hiact1 what is the hiact value for ccode1?

hiact2 what is the hiact value for ccode2?

hostlev1 what is the hostlev value for ccode1?

hostlev2 what is the hostlev value for ccode2?

orig1 is ccode1 an originator (1) of the dispute or a joiner (0)?

orig2 is ccode2 an originator (1) of the dispute or a joiner (0)?

hiact the highest level of action observed in the dispute

hostlev the hostility level of action observed in the dispute

mindur the minimum length of the dispute (in days)

maxdur the maximum length of the dispute (in days)

outcome the dispute-level outcome

settle the settlement value for the dispute

fatality the ordinal fatality level for the dispute

fatalpre the fatalities (with precision, if known) for the dispute

stmon the start month of the dispute (dispute-level)

endmon the end month of the dispute (dispute-level)

recip was the dispute reciprocated (i.e. did Side B also have a militarized action)?

numa the number of participants on Side A

numb the number of participants on Side B

ongo2010 was the dispute ongoing as of 2010?

version a version identifier

Details

Data are the directed dispute-year data made available in version 2.1.1 of the GML MID data.

I would caution against using the revtype variables. They are not informative. They are however included for legacy reasons.

References

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

gml_mid_ddlydisps *Directed Leader-Dyadic Dispute-Year Data with No Duplicate Leader-Dyad-Years (GML, v. 2.2.1, Archigos v. 4.1)*

Description

These are directed leader-dyadic dispute year data derived from the Gibler-Miller-Little (GML) Militarized Interstate Dispute (MID) project. Data are from version 2.2.1 (GML-MID) and version 4.1 (Archigos). These were whittled to where there is no duplicate dyad-years. Its primary aim here is merging into a dyad-year data frame.

Usage

```
gml_mid_ddlydisps
```

Format

A data frame with 10708 observations on the following 12 variables.

`dispnum` a numeric vector for the dispute number

`ccode1` a numeric vector for the focal state in the dyad

`ccode2` a numeric vector for the target state in the dyad

`obsid1` a character vector for the leader of the focal state in the dyad, if avialable

`obsid2` a character vector for the leader of the target state in the dyad, if avialable

`year` a numeric vector for the dispute-year

`gmlmidongoing` a numeric vector for whether there was a dispute ongoing in that year

`gmlmidonset` a numeric vector for whether it was the onset of a new dispute (or new participant-entry into a recurring dispute)

`sidea1` is `ccode1` on side A of the dispute?

`sidea2` is `ccode2` on side A of the dispute?

`orig1` is `ccode1` an originator of the dispute?

`orig2` is `ccode2` an originator of the dispute?

`obsid_start1` the ID of the leader at the dispute onset for `ccode1`

`obsid_start2` the ID of the leader at the dispute onset for `ccode2`

`obsid_end1` the ID of the leader at the dispute conclusion for `ccode1`

`obsid_end2` the ID of the leader at the dispute conclusion for `ccode2`

Details

The process of creating these is described at one of the references below. Importantly, these data are somewhat "naive." That is: they won't tell you, for example, that Brazil and Japan never directly fought each other during World War II. Instead, it will tell you that there were two years of overlap for the two on different sides of the conflict and that the highest action for both was a war. The data are thus similar to what the `EUGene` program would create for users back in the day. Use these data with that limitation in mind.

Data were created by first selecting on unique onsets. Then, where duplicates remained: retaining highest fatality, highest hostility level, highest estimated minimum duration, reciprocated observations over unreciprocated observations, and, finally, the lowest start month.

Be mindful that Archigos' leader data are nominally denominated in Gleditsch-Ward states, which are standardized to Correlates of War state system membership as well as the data can allow. There will be some missing leaders after 1870 because Archigos is ultimately its own system.

References

Miller, Steven V. 2021. "How to (Meticulously) Convert Participant-Level Dispute Data to Dyadic Dispute-Year Data in R." URL: http://svmiller.com/blog/2021/05/convert-cow-mid-data-to-dispute-year/

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

`gml_mid_ddydisps` *Directed Dyadic Dispute-Year Data with No Duplicate Dyad-Years (GML, v. 2.2.1)*

Description

These are directed dyadic dispute year data derived from the Gibler-Miller-Little (GML) Militarized Interstate Dispute (MID) project. Data are from version 2.2.1. These were whittled to where there is no duplicate dyad-years. Its primary aim here is merging into a dyad-year data frame.

Usage

```
gml_mid_ddydisps
```

Format

A data frame with 9262 observations on the following 25 variables.

`dispnum` a numeric vector for the dispute number

`ccode1` a numeric vector for the focal state in the dyad

`ccode2` a numeric vector for the target state in the dyad

`year` a numeric vector for the dispute-year

`gmlmidongoing` a numeric vector for whether there was a dispute ongoing in that year

`gmlmidonset` a numeric vector for whether it was the onset of a new dispute (or new participant-entry into a recurring dispute)

`sidea1` is `ccode1` on side A of the dispute?

`sidea2` is `ccode2` on side A of the dispute?

`fatality1` a numeric vector for the overall fatality level of `ccode1` in the dispute

`fatality2` a numeric vector for the overall fatality level of `ccode2` in the dispute

`fatalpre1` a numeric vector for the known fatalities (with precision) for `ccode1` in the dispute

`fatalpre2` a numeric vector for the known fatalities (with precision) for `ccode2` in the dispute

`hiact1` a numeric vector for the highest action of `ccode1` in the dispute

`hiact2` a numeric vector for the highest action of `ccode2` in the dispute

`hostlev1` a numeric vector for the hostility level of `ccode1` in the dispute

`hostlev2` a numeric vector for the hostility level of `ccode2` in the dispute

`orig1` is `ccode1` an originator of the dispute?

`orig2` is `ccode2` an originator of the dispute?

`fatality` a numeric vector for the fatality level of the dispute

`hostlev` a numeric vector for the hostility level of the MID

`mindur` a numeric vector for the minimum duration of the MID

`maxdur` a numeric vector for the maximum duration of the MID

`recip` a numeric vector for whether a MID was reciprocated

`stmon` a numeric vector for the start month of the MID

Details

The process of creating these is described at one of the references below. Importantly, these data are somewhat "naive." That is: they won't tell you, for example, that Brazil and Japan never directly fought each other during World War II. Instead, it will tell you that there were two years of overlap for the two on different sides of the conflict and that the highest action for both was a war. The data are thus similar to what the `EUGene` program would create for users back in the day. Use these data with that limitation in mind.

References

Miller, Steven V. 2021. "How to (Meticulously) Convert Participant-Level Dispute Data to Dyadic Dispute-Year Data in R." URL: http://svmiller.com/blog/2021/05/convert-cow-mid-data-to-dispute-year/

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

gml_mid_dirleaderdisps *Directed Leader-Dyadic Dispute-Year Data (GML, v. 2.2.1, Archigos v. 4.1)*

Description

These are directed leader-dyadic dispute year data derived from the Gibler-Miller-Little (GML) Militarized Interstate Dispute (MID) project. Data are from version 2.2.1 (GML-MID) and version 4.1 (Archigos). The data are all relevant dyadic leader pairings in conflict, allowing users to employ their own case exclusion rules to the data as they see fit.

Usage

```
gml_mid_dirleaderdisps
```

Format

A data frame with 11686 observations on the following 16 variables.

`dispnum` a numeric vector for the dispute number

`ccode1` a numeric vector for the focal state in the dyad

`ccode2` a numeric vector for the target state in the dyad

`obsid1` a character vector for the leader of the focal state in the dyad, if avialable

`obsid2` a character vector for the leader of the target state in the dyad, if avialable

`year` a numeric vector for the dispute-year

`gmlmidongoing` a numeric vector for whether there was a dispute ongoing in that year

`gmlmidonset` a numeric vector for whether it was the onset of a new dispute (or new participant-entry into a recurring dispute)

`sidea1` is `ccode1` on side A of the dispute?

`sidea2` is `ccode2` on side A of the dispute?

`orig1` is `ccode1` an originator of the dispute?

`orig2` is `ccode2` an originator of the dispute?

`obsid_start1` the ID of the leader at the dispute onset for `ccode1`

`obsid_start2` the ID of the leader at the dispute onset for `ccode2`

`obsid_end1` the ID of the leader at the dispute conclusion for `ccode1`

`obsid_end2` the ID of the leader at the dispute conclusion for `ccode2`

Details

The process of creating these is described at one of the references below. Importantly, these data are somewhat "naive." That is: they won't tell you, for example, that Brazil and Japan never directly fought each other during World War II. Instead, it will tell you that there were two years of overlap for the two on different sides of the conflict and that the highest action for both was a war. The data are thus similar to what the `EUGene` program would create for users back in the day. Use these data with that limitation in mind.

Be mindful that Archigos' leader data are nominally denominated in Gleditsch-Ward states, which are standardized to Correlates of War state system membership as well as the data can allow. There will be some missing leaders after 1870 because Archigos is ultimately its own system.

References

Miller, Steven V. 2021. "How to (Meticulously) Convert Participant-Level Dispute Data to Dyadic Dispute-Year Data in R." URL: http://svmiller.com/blog/2021/05/convert-cow-mid-data-to-dispute-year/

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

`gml_mid_disps` *Abbreviated GML MID Dispute-level Data (v. 2.2.1)*

Description

This is an abbreviated version of the dispute-level Gibler-Miller-Little (GML) MID data.

Usage

```
gml_mid_disps
```

Format

A data frame with 2436 observations on the following 7 variables.

`dispnum` a numeric vector for the CoW-MID dispute number

`styear` a numeric vector for the start year of the MID

`stmon` a numeric vector for the start month of the MID

`outcome` a numeric vector for the outcome of the MID

`settle` a numeric vector for the how dispute was settled

`fatality` a numeric vector for the fatality level of the dispute

`mindur` a numeric vector for the minimum duration of the MID

`maxdur` a numeric vector for the maximum duration of the MID

`hiact` a numeric vector for the highest action of the MID

`hostlev` a numeric vector for the hostility level of the MID

`recip` a numeric vector for whether a MID was reciprocated

Details

These data are purposely light on information; they're not intended to be used for dispute-level analyses, per se. They're intended to augment the directed dyadic dispute-year data by adding in variables that serve as exclusion rules to whittle the data from dyadic dispute-year to just dyad-year data.

References

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

`gml_part` *Participant Summaries of the GML-MID Data*

Description

These are the participant summaries of the most recent GML-MID data. The data also include leaders at the onset and conclusion of a participant episode in the GML MID data.

Usage

```
gml_part
```

Format

A data frame with 5217 observations on the following 21 variables.

`dispnum` the dispute ID in the GML MID data

`ccode` the Correlates of War code for the participant

`styear` the start year for the participant

`stmon` the start month for the participant

`stday` the start day for the participant

`endyear` the end year for the participant

`endmon` the end month for the participant

`endday` the end day for the participant

`obsid_start` an observational ID from `archigos` for the leader at the participant onset

`obsid_end` an observational ID from `archigos` for the leader at the participant conclusion

`dummy_stday` a "dummy" start day for the participant. See details for more.

`dummy_endday` a "dummy" end day for the participant. See details for more.

`sidea` was participant on Side A of the dispute

`hiact` highest action for participant in dispute(-episode)

`orig` was participant an originator?

`anymiss_leader_start` a dummy variable for disputes that equals 1 for a dispute in which *any* participant has a missing leader ID at the start date.

`anymiss_leader_end` a dummy variable for disputes that equals 1 for a dispute in which *any* participant has a missing leader ID at the end date.

`allmiss_leader_start` a dummy variable for disputes that equals 1 for a dispute in which *all* participants have a missing leader ID at the start date.

`allmiss_leader_end` a dummy variable for disputes that equals 1 for a dispute in which *all* participants have a missing leader ID at the end date.

Details

Information about leaders come from Archigos (v. 4.1). GML MID Data are version 2.2.1. The `data-raw` directory contains information about how these data were generated. There is invariably going to be some guesswork here because dates are sometimes not known with precision. Sometimes, a dispute coincides even with a leadership change when dates are known with precision. The source script includes a discussion of these cases and shows how the data were generated with all these caveats in mind.

Do note that participants can have several episodes within a dispute. Sometimes participants switch sides (e.g. Romania in World War 2). Sometime participants drop in and out of a long-running dispute (e.g. Syria, prominently, in MID#4182).

"Dummy" start days and end days are there to serve as a parlor trick in assigning disputes to leaders in leader-level analyses. Where days are known with precision, the dummy day is that number. In most cases, where the day is not known with precision coincides with a month that has no leader transition. Thus, the start day that gets imputed is going to be the first of the month (for the dummy start day) or the last of the month (for the dummy end day). Cases where there was a leader transition (or two) that month may require some more sensitive imputing. For example, our best guess is Antonio Guzmán Blanco of Venezuela is president for the end of MID#1639, given his role in trying to negotiate a conclusion to the dispute. Archigos has him leaving office on the 7th, so that's the end day that gets imputed for him. Again, these are here to serve as a parlor trick in assigning disputes to leaders for leader-level analyses. Be careful about using these data for calculating dispute-participant duration. In fact: don't do that.

References

Gibler, Douglas M., Steven V. Miller, and Erin K. Little. 2016. "An Analysis of the Militarized Interstate Dispute (MID) Dataset, 1816-2001." International Studies Quarterly 60(4): 719-730.

Goemans, Henk E., Kristian Skrede Gleditsch, and Giacomo Chiozza. 2009. "Introducing Archigos: A Dataset of Political Leaders" *Journal of Peace Research* 46(2): 269–83.

---

grh_arms_races *Conventional Arms Races During Periods of Rivalry*

Description

This is a simple data set of 71 arms races reported by Gibler et al. in their 2005 article in *Journal of Peace Research*.

Usage

```
grh_arms_races
```

Format

A data frame the following five variables.

`race_id` the arms race identifier

`ccode1` a numeric vector for the Correlates of War state code for the first state

`ccode2` a numeric vector for the Correlates of War state code for the second state

`styear` the start year for the arms race

`endyear` the end year for the arms race

Details

Data are taken from the appendix of Gibler, Rider, and Hutchison's 2005 article in *Journal of Peace Research*. Read the article and appendix for more information about coding procedures.

References

Gibler, Douglas M., Toby J. Rider, and Marc L. Hutchison. 2005. "Taking Arms Against a Sea of Troubles: Conventional Arms Races during Periods of Rivalry" *Journal of Peace Research* 42(2): 131–47.

`gwcode_democracy` *Democracy data for all Gleditsch-Ward states*

Description

These are democracy data for all Correlates of War state system members.

Usage

```
gwcode_democracy
```

Format

A data frame with 18289 observations on the following 5 variables.

`gwcode` the Gleditsch-Ward system code

`year` a numeric vector for the year

`v2x_polyarchy` the Varieties of Democracy "polyarchy" estimate

`polity2` the the `polity2` score from the Polity project

`xm_qudsest` an extension of the Unified Democracy Scores (UDS) estimates, made possibly by the `QuickUDS` package from Xavier Marquez.

Details

Missing data connote data that are unavailable for various reasons. Either there is no democracy data to code or, in the case of the Polity project, the state system member is outright not evaluated for the variable.

The Polity data are from 2017. The Varieties of Democracy data are version 10. Xavier Marquez' `QuickUDS` estimates (i.e. extensions of Pemstein et al. (2010)) come from a package Marquez makes available on his Github (`https://github.com/xmarquez/QuickUDS`).

References

Coppedge, Michael, John Gerring, Carl Henrik Knutsen, Staffan I. Lindberg, Jan Teorell, David Altman, Michael Bernhard, M. Steven Fish, Adam Glynn, Allen Hicken, Anna Luhrmann, Kyle L. Marquardt, Kelly McMann, Pamela Paxton, Daniel Pemstein, Brigitte Seim, Rachel Sigman, Svend-Erik Skaaning, Jeffrey Staton, Agnes Cornell, Lisa Gastaldi, Haakon Gjerlow, Valeriya Mechkova, Johannes von Romer, Aksel Sundtrom, Eitan Tzelgov, Luca Uberti, Yi-ting Wang, Tore Wig, and Daniel Ziblatt. 2020. "V-Dem Codebook v10" Varieties of Democracy (V-Dem) Project.

Marshall, Monty G., Ted Robert Gurr, and Keith Jaggers. 2017. "Polity IV Project: Political Regime Characteristics and Transitions, 1800-2017." Center for Systemic Peace.

Marquez, Xavier, "A Quick Method for Extending the Unified Democracy Scores" (March 23, 2016). doi: 10.2139/ssrn.2753830

Pemstein, Daniel, Stephen Meserve, and James Melton. 2010. "Democratic Compromise: A Latent Variable Analysis of Ten Measures of Regime Type." *Political Analysis* 18(4): 426-449.

`gw_capitals` *A complete list of capitals and capital transitions for Gleditsch-Ward state system members*

Description

This is a complete list of capitals and capital transitions for Gleditsch-Ward state system members. I use it internally for calculating capital-to-capital distances in the `add_capital_distances()` function.

Usage

```
gw_capitals
```

Format

A data frame with 248 observations on the following 7 variables.

`gwcode` a numeric vector for the Gleditsch-Ward state code

`statenme` a character vector for the state

`capital` a character vector for the name of the capital

`styear` a character vector for the start year. See details section for more information.

`endyear` a character vector for the end year. See details section for more information.

`lat` a numeric vector of the latitude coordinates for the capital

`lng` a numeric vector of the longitude coordinates for the capital

Details

For convenience, the start year for most states is 1816. Samoa, for example, was not a state in 1816. However, the functions that use the `gw_capitals` data will not create observations for states that did not exist at a given point in time.

The data should be current as of the end of 2020.

Cases where a start year is not 1816 indicate a capital transition. For example, Brazil's capital moved from Rio de Janeiro to Brasilia (a planned capital) in 1960. Only 25 states in the data experienced a capital transition. The most recent was Burundi in 2018. Indonesia, as of writing, is planning on a capital transition, but this has not been completed yet.

Kazakhstan renamed its capital for the state leader in 2019. These data retain the name of Astana. This will be changed in the event the software I use records this change.

The capitals data are not without some peculiarities. Prominently, Portugal transferred the Portuguese court from Lisbon to Rio de Janeiro from 1808 to 1821. *This is recorded in the data.* A knowledge of the inter-state conflict data will note there was no war or dispute between, say, Portugal and Spain (or Portugal and any other country) at any point during this time, but it does create some weirdness that would suggest a massive distance between two countries, like Portugal and Spain, that are otherwise land-contiguous.

On Spain: the republican government moved the capital at the start of the civil war (in 1936) to Valencia. However, it abandoned this capital by 1937. I elect to not record this capital transition.

On Myanmar: the Gleditsch-Ward system stands out as having Myanmar entered for the bulk of the 19th century. The capitals recorded for Myanmar (Burma) coincide with capitals of the Konbaung dynasty.

The data also do some (I think) reasonable back-dating of capitals to coincide with states in transition without necessarily formal capitals by the first appearance in the state system membership data. These concern Lithuania, Kazakhstan, and the Philippines. Kaunas is the initial post-independence capital of Lithuania. Almaty is the initial post-independence capital of Kazakhstan. Quezon City is the initial post-independence capital of the Philippines. This concerns, at the most, one or two years for each of these three countries.

`gw_cow_years` *Gleditsch-Ward states and Correlates of War, by year*

Description

This is a complete (I believe) data set on Gleditsch-Ward states and Correlates of War states, a byproduct of a `full_join()` between `gw_states` and `cow_states` that leans largely on the state abbreviation variable.

Usage

```
gw_cow_years
```

Format

A data frame with 18425 observations on the following 6 variables.

`gwcode` a Gleditsch-Ward state code

`stateabb` the state abbreviation, which was the greatest source of agreement between both data sets

`gw_statename` the state name as it appears in the Gleditsch-Ward data

`ccode` a Correlates of War state code

`cow_statename` the state name as it appears in the Correlates of War data

`year` a numeric vector for the year

Details

The `data-raw` directory on the project's Github contains more information about how these data were created. I'm going to use it for internal stuff. The workflow is going to treat the Gleditsch-Ward state system membership codes as more of the "master" codes, for which the user can add Correlates of War identifiers as they see fit. Data are extended to 2020, assuming no changes to state system membership for either data set.

---

gw_ddy *A directed dyad-year data frame of Gleditsch-Ward state system members*

Description

This is a complete directed dyad-year data frame of Gleditsch-Ward state system members. I offer it here as a shortcut for various other functions.

Usage

```
gw_ddy
```

Format

A data frame with 1999558 observations on the following 4 variables.

`gwcode1` a numeric vector for the Correlates of War state code for the first state

`gwcode2` a numeric vector for the Correlates of War state code for the second state

`year` a numeric vector for the year

Details

Data are a quick generation from the `create_dyadyears(system="gw")` function in this package.

gw_mindist *The Minimum Distance Between States in the Gleditsch-Ward System, 1886-2019*

Description

These are non-directed dyad-year data for the minimum distance between states in the Gleditsch-Ward state system from 1886 to 2018. The data are generated from the **cshapes** package.

Usage

```
gw_mindist
```

Format

A data frame with 868813 observations on the following 4 variables.

`gwcode1` the Gleditsch-Ward state system code for the first state

`gwcode2` the Gleditsch-Ward state system code for the second state

`year` the year

`mindist` the minimum distance between states on Jan. 1 of the year, in kilometers

Details

The data are generated from the **cshapes** package. The package authors purport that the data are generated to be compatible with the Gleditsch-Ward system. I trust them on this; indeed, Gleditsch is one of the authors of the **cshapes** package.

Data are automatically generated (by default) as directed dyad-years. I elect to make them non-directed for space considerations. Making non-directed dyad-year data into directed dyad-year data isn't too difficult in R. It just looks weird to see the code that does it.

Previous versions of these data were for the minimum distance as of Dec. 31 of the referent year. These are now Jan. 1. Most of the data I prove elsewhere in this package are to be understood as the data as they were at the *start* of the year. This is how I process, for example, the `capitals` data as they get merged in the `add_capital_distance()` function. However, the script that generates these data are set at Jan. 1 of the year and not Dec. 31. Right now, the **cshapes** does not appear to work on my system and I do not know why. Fortunately, the package authors made these data available.

References

Schvitz, Guy, Luc Girardin, Seraina Ruegger, Nils B. Weidmann, Lars-Erik Cederman, and Kristian Skrede Gleditsch. 2022. "Mapping The International System, 1886-2017: The `CShapes` 2.0 Dataset." *Journal of Conflict Resolution*. 66(1): 144-161.

Weidmann, Nils B. and Kristian Skrede Gleditsch. 2010. "Mapping and Measuring Country Shapes: The `cshapes` Package." *The R Journal* 2(1): 18-24

gw_sdp_gdp *(Surplus and Gross) Domestic Product for Gleditsch-Ward States*

Description

These are state-year level data for surplus and gross domestic product for Correlates of War state system members. Data also include population estimates for per capita standardization.

Usage

```
gw_sdp_gdp
```

Format

A data frame with 27387 observations on the following five variables.

`gwcode` a numeric vector for the Gleditsch-Ward state code

`year` a numeric vector for the year

`wbgdp2011est` a numeric vector for the estimated natural log of GDP in 2011 USD (log-transformed)

`wbpopest` a numeric vector for the estimated population size (log-transformed)

`sdpest` a numeric vector for the estimated surplus domestic product (log-transformed)

`wbgdppc2011est` a numeric vector for the estimated GDP per capita (log-transformed)

Details

These were provided by Anders on a separate Github repository for this project. Because these data are ultimately being simulated, a user can expect some slight differences between the Correlates of War version of these data (which Anders et al. published) and the Gleditsch-Ward version of these data (which appear to be the one the authors will more vigorously support going forward).

Space considerations compel me to round these data to three decimal points. These "economic" data are routinely the biggest in the package, and it's because of the decimal points. The justification for this is these data are estimated/simulated anyways and the information loss is at the 1/1000th decimal point. This procedure basically cuts the size of the data to be less than 25% of its original size. The original simulations are available for remote download if you'd like. Type `?download_extdata()` for more information.

References

Anders, Therese, Christopher J. Fariss, and Jonathan N. Markowitz. 2020. "Bread Before Guns or Butter: Introducing Surplus Domestic Product (SDP)" *International Studies Quarterly* 64(2): 392–405.

gw_states *Gleditsch-Ward (Independent States) System Membership Data (1816-2017)*

Description

These are the independent states in Gleditsch and Ward's data.

Usage

```
gw_states
```

Format

A data frame with 216 observations on the following 5 variables.

`gwcode` a numeric vector for the Gleditsch-Ward country code

`stateabb` a character vector for state abbreviation

`statename` a character vector for the state name

`startdate` the start date in the data

`enddate` the end date in the data

Details

Data originally provided by Gleditsch with no column names. Column names were added before some light re-cleaning in order to generate these data. "Wuerttemberg" and "Cote D'Ivoire" in the `statename` column needed to be renamed to ensure maximal compliance with CRAN, which raises notes for every non-ASCII character that appears in its package. I do not think this to be problematic at all and, after all, state names should never be a basis for something like a match or merge you would do in **countrycode**.

References

Gleditsch, Kristian S. and Michael D. Ward. 1999. "A Revised List of Independent States since the Congress of Vienna." *International Interactions* 25(4): 393–413.

hief *Historical Index of Ethnic Fractionalization data*

Description

This is a data set with state-year estimates for ethnic fractionalization.

Usage

```
hief
```

Format

A data frame with 8808 observations on the following 5 variables.

`ccode` a Correlates of War state code

`gwcode` a Gleditsch-Ward state code

`year` the year

`efindex` a numeric vector for the estimate of ethnic fractionalization

Details

The `data-raw` directory on the project's Github contains more information about how these data were created.

References

Drazanova, Lenka. 2020. "Introducing the Historical Index of Ethnic Fractionalization (HIEF) Dataset: Accounting for Longitudinal Changes in Ethnic Diversity." *Journal of Open Humanities Data* 6:6 doi: 10.5334/johd.16

LEAD *(An Abbreviation of) The LEAD Data Set*

Description

These are an abbreviated version of the LEAD Data Set, incorporating variables that I think are most interesting or potentially useful from these data.

Usage

```
LEAD
```

Format

A data frame with 3409 observations on the following 12 variables.

`obsid` an observational ID from `archigos`

`leveledu` 0 = primary, 1 = secondary, 2 = university, 3 = graduate

`milservice` did leader have prior military service?

`combat` did leader have prior combat experience in military service?

`rebel` was leader previously part of a rebel group?

`warwin` was leader previously part of a winning war effort as part of military service?

`warloss` was leader previously part of a losing war effort as part of military service?

`rebelwin` was leader previously part of a winning war effort as part of a rebel group?

`rebelloss` was leader previously part of a losing war effort as part of a rebel group?

`yrsexper` previous years of experience in politics before becoming a leader

`physhealth` does leader have physical health issues?

`mentalhealth` does leader have mental health issues?

Details

Data are ported from Ellis et al. (2015). Users who want more of these variables included in **peacesciencer** should raise an issue on Github.

References

Ellis, Carli Mortenson, Michael C. Horowitz, and Allan C. Stam. 2015. "Introducing the LEAD Data Set." *International Interactions* 41(4): 718–741.

`leader_codes` *A Data Set of Leader Codes Across Archigos 4.1, Archigos 2.9, and the LEAD Data*

Description

This is a simple data set that matches, as well as one can, leader codes across Archigos 4.1, Archigos 2.9, and the LEAD data set.

Usage

```
leader_codes
```

Format

A data frame the following four variables.

`obsid` the observation ID in the Archigos data

`leadid` the leader ID in version 4.1 of the Archigos data

`leadid29` the leader ID in version 2.9 of the Archigos data

`leaderid` the leader ID in the LEAD data

Details

These data treat version 4.1 of the Archigos data as the gospel leader data (if you will) for which the observation ID (`obsid`) is the master code indicating a leader tenure period. It also builds in an assumption that various observations that duplicate in the LEAD data should not have duplicated. This concerns Francisco Aguilar Barquer (who appears twice), Emile Reuter (who appears twice), and Gunnar Thoroddsen (who appears three times) in the LEAD data despite having uninterrupted tenures in office. None of the covariates associated with these leaders change in the LEAD data, which is why I assume they were duplicates.

`lwuf` *Leader Willingness to Use Force*

Description

These are the estimates of leader willingness to use force as estimated by Carter and Smith (2020).

Usage

```
lwuf
```

Format

A data frame with 3409 observations on the following 9 variables.

`obsid` an observational ID from `archigos`

`theta1_mean` the mean simulated M1 theta, as estimated by Carter and Smith (2020)

`theta1_sd` the standard deviation of simulated M1 thetas

`theta2_mean` the mean simulated M2 theta, as estimated by Carter and Smith (2020)

`theta2_sd` the standard deviation of simulated M2 thetas

`theta3_mean` the mean simulated M3 theta, as estimated by Carter and Smith (2020)

`theta3_sd` the standard deviation of simulated M3 thetas

`theta4_mean` the mean simulated M4 theta, as estimated by Carter and Smith (2020)

`theta4_sd` the standard deviation of simulated M4 thetas

Details

The letter published by the authors contains more information as to what these thetas refer. The "M1" theta is a variation of the standard Rasch model from the boilerplate information in the LEAD data. The authors consider this to be "theoretically relevant" or "risk-related" as these all refer to conflict or risk-taking. The "M2" theta expands on "M1" by including political orientation and psychological characteristics. "M3" and "M4" expand on "M1" and "M2" by considering all 36 variables in the LEAD data.

The authors construct and include all these measures, though their analyses suggest "M2" is the best-performing measure.

References

Carter, Jeff and Charles E. Smith, Jr. 2020. "A Framework for Measuring Leaders' Willingness to Use Force." *American Political Science Review* 114(4): 1352–1358.

`maoz_powers` *Zeev Maoz' Regional/Global Power Data*

Description

These are Zeev Maoz' data for what states are regional or global powers at a given point time. They are extensions of the Correlates of War major power data, which only codes "major" power without consideration of regional or global distinctions. Think of Austria-Hungary as intuitive of the issue here. Austria-Hungary is a major power in the Correlates of War data, but there is good reason to treat Austria-Hungary as a major power only within Europe. That is what Zeev Maoz tries to do here.

Usage

```
maoz_powers
```

Format

A data frame with 20 observations on the following 5 variables.

`ccode` a numeric vector for the Correlates of War country code

`regstdate` the start date for regional power status

`regenddate` the end date for regional power status

`globstdate` the start date for global power status

`globenddate` the end date for global power status

References

Maoz, Zeev. 2010. Network of Nations: The Evolution, Structure, and Impact of International Networks, 1816-2001. Cambridge University Press.

ps_bib *A* BibTeX *Data Frame of Citations*

Description

This is a BibTeX file, loaded as a data frame, to assist the user in properly citing the source material that is used in this package.

Usage

```
ps_bib
```

Format

A data frame with the following columns.

CATEGORY the BibTeX entry type

BIBTEXKEY the BibTeX unique entry key

ADDRESS another BibTeX field

ANNOTE another BibTeX field

AUTHOR a list of authors for this entry

BOOKTITLE another BibTeX field, for book title (if appropriate)

CHAPTER another BibTeX field, for chapter (if appropriate)

CROSSREF another BibTeX field

EDITION another BibTeX field, for edition of book (if appropriate)

EDITOR another BibTeX field, for book editor (if appropriate)

HOWPUBLISHED another BibTeX field

INSTITUTION another BibTeX field

JOURNAL another BibTeX field, for the journal name (if appropriate)

KEY another BibTeX field

MONTH another BibTeX field

NOTE another BibTeX field

NUMBER another BibTeX field, for journal volume number (if appropriate)

ORGANIZATION another BibTeX field

PAGES another BibTeX field, for pages of the entry

PUBLISHER another BibTeX field, for book publisher (if appropriate)

SCHOOL another BibTeX field

SERIES another BibTeX field

TITLE another BibTeX field, for title of the entry

TYPE another BibTeX field

VOLUME another BibTeX field, for journal volume (if appropriate)

YEAR another BibTeX field, for year of publication

KEYWORDS another BibTeX field, used primarily for selective filtering in this package

URL another BibTeX field, for website (if appropriate)

OWNER another BibTeX field

TIMESTAMP another BibTeX field, used occasionally when I started populating my master file (you will see some old entries here)

DOI another BibTeX field, for a digital object identifier (used rarely)

EPRINT another BibTeX field

JOURNALTITLE another BibTeX field, which I think is actually a BibLaTeX field

ISSN another BibTeX field

ABSTRACT another BibTeX field, for entry abstract (if appropriate)

DATE.ADDED another BibTeX field

DATE.MODIFIED another BibTeX field

Details

See data-raw directory for how these data were generated. The data were created by **bib2df**, which is now a package dependency. I assume the user has some familiarity with BibTeX. Some entries were copy-pasted from my master bibliography file that I started in 2008 or so.

---

ps_cite *Get* BibTeX *Entries Associated with* **peacesciencer** *Data and Functions*

Description

ps_cite() allows the user to get citations to scholarship that they should include in their papers that incorporate the functions and data in this package.

Usage

```
ps_cite(x, column = "keywords")
```

Arguments

x	a character vector
column	a character vector for the particular column of ps_bib the user wants to search. The default here is "keywords", which searches the KEYWORDS column in ps_bib for the most general search. The other option is "bibtexkey", which will search the BIBTEXKEY column in ps_bib. Use the latter option more for pairing with output from ps_version()

Details

The base functionality here is simple pattern-matching on keywords in ps_bib. This simple pattern-matching is in base R. I assume the user has some familiarity with BibTeX.

Value

ps_cite() takes a character vector and scans the ps_bib data in this package to return a BibTeX citation (or citations) for the researcher to use to properly cite the material they are getting from this package. The citations are returned as a full BibTeX entry (or entries) that they can copy-paste into their own BibTeX file.

Author(s)

Steven V. Miller

Examples

```
# Cite the package
ps_cite("peacesciencer")
```

---

ps_data_version *The Version Numbers for Data Included in* **peacesciencer**

Description

This is a simple data set that communicates the version numbers of data included in this package. It's a companion to the data frame ps_bib, and other information functions like ps_cite() and ps_version(). The latter uses this data set.

Usage

```
ps_data_version
```

Format

A data frame the following four variables.

category a category for the type of data

data the name of the particular data source coinciding with the category

version the version number included in **peacesciencer** for this data source

bibtexkey a character key for the BibTeX key corresponding with an appropriate citation in ps_bib

Details

Version numbers that are years should be understood as data sources with no formal version numbering system, per se. Instead, they communicate a year of last update. For example, the Correlates of War does not formally version number its state system data as it does its MID data. Likewise, the Anders et al. (2020) simulations of population and surplus/gross domestic product are not formally versioned, per se. Instead, the data were published and last updated in 2020.

`ps_version` *Get Version Information About Data Included in* **peacesciencer**

Description

`ps_version()` allows the user to see version information about data included in **peacesciencer**.

Usage

```
ps_version(cat)
```

Arguments

`cat`	a category of data type the user wants, as a character

Details

The base functionality here is simple pattern-matching on keywords in `ps_data_version`. This simple pattern-matching is in base R. I assume the user has some familiarity with the types of data included in this package.

The searching is done by category included in the `ps_data_version` data. Users may want to just minimally run `ps_version()` with no argument specified to see for themselves what's in it. Typing `unique(ps_data_version$category)` may also get them started.

The user can consider this a companion function to `ps_cite()`. Whereas `ps_cite()` will return the appropriate citation to use in the bibliography, it may not tell them the version number at all. For example, the classic and suggested citations for the Correlates of War National Material Capabilities data are too Singer et al. (1972) and Singer (1987), though the data included in this package are about 30 years older than the most recent citation of the two.

The information communicated here can/should be included alongside a parenthetical citation. For example, the contiguity data are quite a bit more current than the suggested citation to Stinnett et al. (2002). Thus, a user may want to cite the data in their paper as something like (Stinnett et al. 2002, v. 3.2).

Value

`ps_version()` takes a character vector and scans the `ps_data_version` data in this package to return information about the particular data versions included in **peacesciencer** as well as a suggested citation key for scanning `ps_cite()`. If no category is specified for searching, it just returns all version information for all data included in functions in this package.

Author(s)

Steven V. Miller

Examples

```
# What can you search for...
unique(ps_data_version$category)

# will show the data versions for everything
ps_version()

# will show data versions for particular categories of data
ps_version("democracy")

ps_version("leaders")
```

`rugged` *Rugged/Mountainous Terrain Data*

Description

This is a data set on state-level estimates for the "ruggedness" of a state's terrain.

Usage

```
rugged
```

Format

A data frame with 192 observations on the following 6 variables.

`ccode` a Correlates of War state code

`gwcode` a Gleditsch-Ward state code

`rugged` the terrain ruggedness index

`newlmtnest` the (natural log) percentage estimate of the state's terrain that is mountainous

Details

The `data-raw` directory on the project's Github contains more information about how these data were created. It goes without saying that these data move *slowly* so the data are really only applicable for making state-to-state comparisons and not states-in-time comparisons. The terrain ruggedness index is originally introduced by Riley et al. (1999) but is amended by Nunn and Puga (2012). The mountain terrain data was originally created by Fearon and Laitin (2003) but extended and amended by Gibler and Miller (2014). The data are functionally time-agnostic—use with caution in your state-year analyses—but all data sets seem to benchmark around 1999-2000. I'm not sure it matters *that* much, but it matters a little at the margins, I suppose, if you suspect there are major differences in interpretation of how much more "rugged" the Soviet Union was than Russia, or Yugoslavia than Serbia.

References

Fearon, James D., and David Laitin, "Ethnicity, Insurgency, and Civil War" *American Political Science Review* 97: 75–90.

Gibler, Douglas M. and Steven V. Miller. 2014. "External Territorial Threat, State Capacity, and Civil War." *Journal of Peace Research* 51(5): 634-646.

Nunn, Nathan and Diego Puga. 2012. "Ruggedness: The Blessing of Bad Geography in Africa." *Review of Economics and Statistics*. 94(1): 20-36.

Riley, Shawn J., Stephen D. DeGloria, and Robert Elliot. 1999. "A Terrain Ruggedness Index That Quantifies Topographic Heterogeneity," *Intermountain Journal of Sciences* 5: 23–27.

`show_duplicates` *Show Duplicate Observations in Your Dyad-Year or State-Year Data Frame*

Description

`show_duplicates()` shows which data are duplicated in data generated in **peacesciencer**. It's a useful diagnostic tool for users doing some do-it-yourself functions with **peacesciencer**.

Usage

```
show_duplicates(data)
```

Arguments

`data`	a dyad-year data frame or a state-year data frame created in **peacesciencer**.

Details

The function leans on attributes of the data that are provided by the `create_dyadyear()` or `create_stateyear()` function. Make sure that function (or data created by that function) appear at the top of the proverbial pipe.

The data returned will also have a new column called `duplicated`. Thus, an implicit assumption in this function is the user does not have a column in the data with this name that is of interest to the user. It will be overwritten.

Value

`show_duplicates()` takes a dyad-year data frame or state-year data frame generated in **peacesciencer** and shows what observations are duplicated by unique combination of dyad-year or state-year, contingent on what was supplied to it.

Author(s)

Steven V. Miller

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)

gml_dirdisp %>% show_duplicates()
cow_mid_dirdisps %>% show_duplicates()
```

`td_rivalries` *Thompson and Dreyer's (2012) Strategic Rivalries, 1494-2010*

Description

A simple summary of all strategic (inter-state) rivalries from Thompson and Dreyer (2012).

Usage

```
td_rivalries
```

Format

A data frame with 197 observations on the following 10 variables.

`rivalryno` a numeric vector for the rivalry number

`rivalryname` a character vector for the rivalry name

`ccode1` the Correlates of War state code for the state with the lowest Correlates of War state code in the rivalry

`ccode2` the Correlates of War state code for the state with the highest Correlates of War state code in the rivalry

`styear` a numeric vector for the start year of the rivalry

`endyear` a numeric vector for the end year of the rivalry

`region` a character vector for the region of the rivalry, per Thompson and Dreyer (2012)

`type1` a character vector for the primary type of the rivalry (spatial, positional, ideological, or interventionary)

`type2` a character vector for the secondary type of the rivalry, if applicable (spatial, positional, ideological, or interventionary)

`type3` a character vector for the tertiary type of the rivalry, if applicable (spatial, positional, ideological, or interventionary)

Details

Information gathered from the appendix of Thompson and Dreyer (2012). Ongoing rivalries are right-bound at 2010, the date of publication for Thompson and Dreyer's handbook. Users are free to change this if they like. Data are effectively identical to `strategic_rivalries` in **stevemisc**, but include some behind-the-scenes processing (described in a blog post on http://svmiller.com) that is available to see on the project's Github repository. The data object is also renamed to avoid a conflict.

References

Miller, Steven V. 2019. "Create and Extend Strategic (International) Rivalry Data in R". URL: http://svmiller.com/blog/2019/10/create-extend-strategic-rivalry-data-r/

Thompson, William R. and David Dreyer. 2012. *Handbook of International Rivalries*. CQ Press.

---

ucdp_acd *UCDP Armed Conflict Data (ACD) (v. 20.1)*

Description

These are (kind of) dyadic, but mostly state-level data, used internally for doing stuff with the UCDP armed conflict data

Usage

```
ucdp_acd
```

Format

A data frame with 4164 observations on the following 15 variables.

`conflict_id` a conflict identifier, not to be confused with an episode identifier (which I don't think UCDP offers)

`year` a numeric vector for the year

`gwno_a` the Gleditsch-Ward state code for the state on side A of the armed conflict

`gwno_a_2nd` the Gleditsch-Ward state code for the state that actively supported side A of the armed conflict with the use of troops

`gwno_b` the Gleditsch-Ward state code for the actor on side B of the armed conflict

`gwno_b_2nd` the Gleditsch-Ward state code for the state that actively supported side B of the armed conflict with the use of troops

`incompatibility` a character vector for the main conflict issue ("territory", "government", "both")

`intensity_level` a numeric vector for the intensity level in the calendar year (1 = minor (25-999 deaths), 2 = war (>1,000 deaths))

`type_of_conflict` a character vector for the type of conflict ("extrasystemic", "interstate", "intrastate", "II"). "II" is a simple abbreviation of "internationalized intrastate"

`start_date` a date of the first battle-related death in the conflict, not to be confused with the first battle-related death of the episode

`start_prec` the level of precision for `start_date`

`start_date2` a date of the first battle-related death in the episode, not to be confused with the first battle-related death of the conflict

`start_prec2` the level of precision for `start_date2`

`ep_end` a dummy variable for whether the conflict episode ended in the calendar year of observation

`ep_end_date` the episode end date, if applicable

Details

The `data-raw` directory on the project's Github will show how I processed the multiple strings for when there are multiple states on a given side.

References

Gleditsch, Nils Petter; Peter Wallensteen, Mikael Eriksson, Margareta Sollenberg & Havard Strand (2002) Armed Conflict 1946–2001: A New Dataset. *Journal of Peace Research* 39(5): 615–637.

Pettersson, Therese; Stina Hogbladh & Magnus Oberg (2019). Organized violence, 1989-2018 and peace agreements. Journal of Peace Research 56(4): 589-603.

`ucdp_onsets` *UCDP Onset Data (v. 19.1)*

Description

These are state-year level data for armed conflict onsets provided by the Uppsala Conflict Data Program (UCDP).

Usage

```
ucdp_onsets
```

Format

A data frame with 10142 observations on the following eight variables.

`gwcode` a numeric vector for the Gleditsch-Ward state code

`year` a numeric vector for the year

`sumnewconf` a numeric vector for the sum of new conflicts/conflict-dyads

`sumonset1` a numeric vector for the sum of new conflict episodes, whether because this is a new conflict or because there is more than one year since last conflict episode

`sumonset2` a numeric vector for the sum of new conflict episodes, whether because this is a new conflict or because there is more than two years since last conflict episode

`sumonset3` a numeric vector for the sum of new conflict episodes, whether because this is a new conflict or because there is more than three years since last conflict episode

`sumonset5` a numeric vector for the sum of new conflict episodes, whether because this is a new conflict or because there is more than five years since last conflict episode

`sumonset10` a numeric vector for the sum of new conflict episodes, whether because this is a new conflict or because there is more than 10 years since last conflict episode

Details

The user will want to note that the data provided by UCDP are technically not country-year observations. They instead duplicate observations for cases of new conflicts or new conflict episodes. Further, the original data do not provide any information about the conflict-dyad in question to which those duplicates pertain. That means the most these data can do for the package's mission is provide summary information. The user should probably recode these variables into something else they may want for a particular application

References

Gleditsch, Nils Petter; Peter Wallensteen, Mikael Eriksson, Margareta Sollenberg & Havard Strand (2002) Armed Conflict 1946–2001: A New Dataset. *Journal of Peace Research* 39(5): 615–637.

Pettersson, Therese; Stina Hogbladh & Magnus Oberg (2019). Organized violence, 1989-2018 and peace agreements. Journal of Peace Research 56(4): 589-603.

---

whittle_conflicts_duration

Whittle Duplicate Conflict-Years by Conflict Duration

Description

`whittle_conflicts_duration()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will keep the observations with the highest estimated duration.

Usage

```
whittle_conflicts_duration(data, durtype = "mindur")

wc_duration(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`durtype`	a duration on which to filter/whittle the data. Options include `"mindur"` or `"maxdur"`. The default is `"mindur"`.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

Some conflicts can be of an unknown length and often come with estimates of a minimum duration and a maximum duration. This will concern the `durtype` parameter in this function. In many/most conflicts, certainly thinking of the inter-state dispute data, dates are known with precision (to the day) and the estimate of minimum conflict duration is equal to the estimate of maximum conflict duration. For some conflicts, the estimates will vary. This does importantly imply that using this particular whittle function with the default (`mindur`) will produce different results than using this particular whittle function and asking to retain the highest maximum duration (`maxdur`). Use the function with that in mind.

`wc_duration()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_duration()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have the highest observed dispute-level fatality. This will not eliminate all duplicates, far from it, but it's a sensible cut later into the procedure (after whittling onsets in `whittle_conflicts_onsets()`, and maybe some other things the extent to which dispute-level duration is a heuristic for dispute-level severity/importance.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_duration()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_duration()
```

`whittle_conflicts_fatality` *Whittle Duplicate Conflict-Years by Highest Fatality*

Description

`whittle_conflicts_fatality()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will keep the observations with the highest observed fatality.

Usage

```
whittle_conflicts_fatality(data)

wc_fatality(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

As of writing, the Correlates of War and Gibler-Miller-Little conflict data record some -9s for fatalities. In those cases, dispute-level fatality is momentarily recoded to be .5 (i.e. fatal, but without too many fatalities). This is a missing data problem that Gibler and Miller correct in a forthcoming publication in *Journal of Conflict Resolution*. Until then, this function makes that kind of determination about disputes with missing fatalities.

`wc_fatality()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_fatality()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have the highest observed dispute-level fatality. This will not eliminate all duplicates, far from it, but it's a sensible second cut (after whittling onsets in `whittle_conflicts_onsets()` the extent to which dispute-level fatality is a good heuristic for dispute-level severity/importance.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_fatality()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_fatality()
```

---

whittle_conflicts_hostility

Whittle Duplicate Conflict-Years by Conflict Hostility

Description

`whittle_conflicts_hostility()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will keep the observations with the highest observed hostility.

Usage

```
whittle_conflicts_hostility(data)

wc_hostility(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

`wc_hostility()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_hostility()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have the highest observed dispute-level fatality. This will not eliminate all duplicates, far from it, but it's a sensible second or third cut (after whittling onsets in `whittle_conflicts_onsets()` the extent to which dispute-level hostility is a good heuristic for dispute-level severity/importance.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_hostility()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_hostility()
```

`whittle_conflicts_jds` *Whittle Duplicate Conflict-Years by Just Dropping Something ("JDS")*

Description

`whittle_conflicts_jds()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will just drop something, as a kind of nuclear option.

Usage

```
whittle_conflicts_jds(data)

wc_jds(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

This really should be the absolute last exclusion rules a researcher uses. It's a "nuclear option", if you will. Assuming you've run other case exclusion rules to isolate onsets and severe disputes, what remains at the end should be duplicates that are functionally equivalent observations. Your data cannot have duplicates, and these remaining observations are basically the same. Therefore, just drop something.

`wc_jds()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_jds()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have the lowest start month.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_jds()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_jds()
```

---

whittle_conflicts_onsets

Whittle Unique Conflict Onset-Years from Conflict-Year Data

Description

whittle_conflicts_reciprocation() is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will drop ongoing conflicts in the presence of unique onsets.

Usage

```
whittle_conflicts_onsets(data)

wc_onsets(...)
```

Arguments

data	a data frame with a declared conflict attribute type.
...	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in add_cow_mids() and add_gml_mids().

wc_onsets() is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_onsets()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations with unique onsets where duplicates exist. This will not eliminate all duplicates, far from it, but it's a sensible place to start.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets()

cow_mid_dirdisps %>% whittle_conflicts_onsets()
```

---

whittle_conflicts_reciprocation

Whittle Duplicate Conflict-Years by Conflict Reciprocation

Description

`whittle_conflicts_reciprocation()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will keep the observations that are reciprocated (i.e. have militarized actions on both sides of the conflict).

Usage

```
whittle_conflicts_reciprocation(data)

wc_recip(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

Scholars are free to use this as a heuristic for whittling conflict-year data to be coerced into true dyad-year data, but I would be remiss if I did not offer a caveat about the reciprocation variable in inter-state dispute data. Namely, it is noisy and is not doing what scholars often think it's doing in the inter-state dispute data. Reciprocation is observed only when there is a militarized action on both sides of the conflict. By definition, someone on Side A will have a militarized action. Not every state on Side B does. However, scholars should *not* interpret that as the absence of militarized responses. In a forthcoming article in *Journal of Conflict Resolution*, Doug Gibler and I make the case that reciprocation isn't a useful variable to maintain at all because it can only invite errors (as is often the case in the CoW-MID data) and will obscure the fact that states that are attacked by another side routinely fight back. On many occasions, they also successfully repel the attack. Scholars who uncritically use this variable, certainly for hypothesis-testing on audience costs, are borrowing trouble with this measure.

`wc_recip()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_reciprocation()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have militarized actions on both sides of the conflict. This will not eliminate all duplicates, far from it, but it's a sensible cut later into the procedure (after whittling onsets in `whittle_conflicts_onsets()` the extent to which dispute-level reciprocation is a heuristic for dispute-level severity/importance (after some other considerations).

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_reciprocation()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_reciprocation()
```

`whittle_conflicts_startmonth` *Whittle Duplicate Conflict-Years by Lowest Start Month*

Description

`whittle_conflicts_startmonth()` is in a class of do-it-yourself functions for coercing (i.e. "whittling") conflict-year data with cross-sectional units to unique conflict-year data by cross-sectional unit. The inspiration here is clearly the problem of whittling dyadic dispute-year data into true dyad-year data (like in the Gibler-Miller-Little conflict data). This particular function will keep the observations that have the lowest start month.

Usage

```
whittle_conflicts_startmonth(data)

wc_stmon(...)
```

Arguments

`data`	a data frame with a declared conflict attribute type.
`...`	optional, only to make the shortcut work

Details

Dyads are capable of having multiple disputes in a given year, which can create a problem for merging into a complete dyad-year data frame. Consider the case of France and Italy in 1860, which had three separate dispute onsets that year (MID#0112, MID#0113, MID#0306), as illustrative of the problem. The default process in **peacesciencer** employs several rules to whittle down these duplicate dyad-years for merging into a dyad-year data frame. These are available in `add_cow_mids()` and `add_gml_mids()`.

This really should be one of the last exclusion rules a researcher uses. There is no substantive reason to assume the lower start month matters for the cause of isolating "serious" or "severe" disputes in the presence of duplicates. It's really just a way of isolating which duplicated observation happened first where remaining duplicates are otherwise very similar to each other.

`wc_stmon()` is a simple, less wordy, shortcut for the same function.

Value

`whittle_conflicts_startmonth()` takes a dyad-year data frame or leader-dyad-year data frame with a declared conflict attribute type and, grouping by the dyad and year, returns just those observations that have the lowest start month.

Author(s)

Steven V. Miller

References

Miller, Steven V. 2021. "How peacesciencer Coerces Dispute-Year Data into Dyad-Year Data". URL: http://svmiller.com/peacesciencer/articles/coerce-dispute-year-dyad-year.html

Examples

```
# just call `library(tidyverse)` at the top of the your script
library(magrittr)
gml_dirdisp %>% whittle_conflicts_onsets() %>% whittle_conflicts_startmonth()

cow_mid_dirdisps %>% whittle_conflicts_onsets() %>% whittle_conflicts_startmonth()
```

Index