BiG Dyads

Code Book

VERSION: 1 JANUARY 2022

BiG Dyads Code Book

Version: 1 January 2022 (V01JAN2022)

Kasra Ghorbaninejad, PhD

Borders in Globalization
https://biglobalization.org/

University of Victoria

BiG Dyads Quick Facts

Number of dyads	770
Number of land dyads	333
Number of sea dyads	437
Number of indicators	27
Number of variables	47

Major updates since last version (V.29JAN2021):

- New 'Mountains Dataset' and few error corrections.

What Is BiG Dyads?

With co-funding from the Jean Monnet Network, we expect the BiG Dyads database to be a major contribution to border scholarship, and to bolster and facilitate further research in the various fields of comparative border studies. This relational database brings the six themes of the Borders in Globalization project together and allows our researchers to not only structure and exploit the current data that we have collected, but also open up new research avenues and, in turn, help grow the database itself.

Our ultimate goal is to develop up to 50 indicators per specific area of knowledge in border studies: (1) border culture, (2) sustainability across borderlands, (3) border security, (4) border history, (5) governance of borders, and (6) border flows/mobility. We have started with all nation members of the United Nations and are expanding to include dyads that are recognised by at least one other nation-state. Following the rules and standards set by the database manager, our leads and their research assistants formulate indicators exploring their respective areas of expertise. Once the indicators are finalised, the data is collected, normalised in accordance with database standards, and inserted into the BiG Dyads. We also work with colleagues and partner organisations to constantly combine resources and, whenever possible, integrate existing research and datasets into our project

BiG Dyads takes close to 800 terrestrial and maritime border dyads worldwide as the main organising principle of its relational framework. (A dyad refers to a segment of border shared by two particular states; e.g., the US international boundary includes two territorial dyads, US–Canada and US–Mexico.) A relational database gives researchers the power to query data points, compare them, and arrive at useful information. Integrating such data points across multiple border-studies themes can often lead to insights otherwise hard, if not impossible, to gain.

Through the BiG Dyads database we aim to:

- make our research available to fellow researchers and, after an inaugural stage, to the public
- help store data already gathered and, where possible, quantify qualitative data
- exploit data to shed light on trends and insights hitherto hidden from observation
- formulate new research questions on the basis of current and prospective indicators
- publish findings which have been made possible through the database

For more information, please see https://biglobalization.org/outputs/big-dyads-database. To get in contact, please e-mail borders@uvic.ca and include the keyword ‘database’ in the subject line.

Table of Contents

1. Description of Database 6
1.1. Datasets, Indicators, and Variables 6
1.2. Composite Indicators (Indices) 6
1.2.1. Equally-Weighted Indices 6
1.2.1.1. Knowability Index 6
1.2.1.2. Applicability Index 6
1.2.2. Unequally-Weighted Indices 7
1.3. Querying 7
1.3.1. Formulating Queries: An Example 7
1.4. Database Platform 8
1.4.1. Spreadsheet Presentation 8
1.5. Data Collection 9
1.6. Normal Forms 9
1.7. Data Collection Do’s and Don’ts 9
1.7.1. Banned Characters 10
1.7.2. Shell Scripts 11
2. Variables 12
2.1. List of Variables 12
2.2. Key Attributes 13
2.3. Missing and Inapplicable Data, and Cut-Off Dates 14
3. Description of Datasets 15
3.1. Foundations of Dyads Dataset 16
3.2. Dates Dataset 18
3.3. Conflicts Dataset 20
3.4. Economic Inequality Dataset 21
3.5. Transboundary Watersheds Dataset 23
3.6. Peace Parks Dataset 25
3.7. Mountains Dataset 28
Appendices 28
Appendix A: Country Alphabetical Order Number 28
Appendix B: Shell Scripts 31
Appendix C: Sources 34

1. Description of Database

BiG Dyads is a relational database (RDB). In simple terms, an RDB organises a number of datasets in relation to one another and allows for easy and fast accessing, querying, and updating of data stored. More specifically, not only are records (rows) in a database crossed by attributes (columns), but also can multiple tables be related to one another.

The BiG Dyads is an RDB that takes the 'dyad'—the boundary-line between two current nation-states—as its main[1] building block, each dyad forming a unique record that is crossed by indicators drawn from the six BiG themes: (1) border culture, (2) sustainability across borderlands, (3) border security, (4) border history, (5) governance of borders, and (6) border flows/mobility.

1.1. Datasets, Indicators, and Variables

Under the supervision of the database manager, BiG-affiliated researchers design and develop their own datasets under the theme that overlaps with their professional interests. General guidelines and templates are available for each dataset, and the results are standardised and approved by the database manager before they become integrated into the separate theme-based tables that come together and form BiG Dyads. Each dataset is comprised of indicators that provide attributes of a dyad. Each dataset indicator may directly correspond to a database variable or, in most cases, the indicator may be broken down into multiple variables: this is done mainly to separate textual/qualitative data from numeric/quantitative data and thus allow easy manipulation of both in querying. For economy of operations, textual data is in certain cases further translated into Boolean or other finite-set variables, or a combination of both.

1.2. Composite Indicators (Indices)

1.2.1. Equally-Weighted Indices

1.2.1.1. Knowability Index

This index gives an equal weight of 1.0 to all the database indicators. However, if the value of an indicator for a dyad is missing (-888) or inapplicable (-999), the dyad is downgraded by -0.5 or -1.0, respectively. Accordingly, the sum total of the indicators divided by their number produces a number equal to or lesser than 1.0. The closer this product to 1.0, the 'better known' the dyad in question is; and the farther the number from 1.0, the less data we have about the dyad (either because for certain indicators the data cannot be found [missing] or cannot be evaluated by [inapplicable]). A product of 1.0 means perfect knowability for a dyad within the parameters of the database. With this index we can compare a dyad against all other dyads in terms of knowability, i.e. how much data we have on any given dyad.

When presented in a spreadsheet (such as Excel), the formula to return the knowability index is as follows:

```
=SUM([variables]-(COUNTIF(H[last-dyad-row]:AQ[last-dyad-row],"-999"))-((COUNTIF(H[last-dyad-row]:AQ[last-dyad-row],"-888")/2)))/[variables]
```

1.2.1.2. Applicability Index

This index gives an equal weight of 1.0 to all the database dyads. However, if the indicator in question is inapplicable (-999) for a dyad, the indicator is downgraded by -1.0. Accordingly, the sum total of the dyads divided by their number produces a number equal to or lesser than 1.0. The closer this product to 1.0, the 'better known' the dyad in question is; and the farther the number from 1.0, the less applicable the indicator in question is to the dyads we have in the database. A product of 1.0 means perfect applicability for an indicator within the parameters of the database. With this index we can compare an indicator against all other indicators in terms of applicability, i.e. how much data each indicator makes available towards knowing any given dyad.

The spreadsheet formula (e.g. Excel) to return the applicability index is as follows:

```
=SUM([dyads]-(COUNTIF(H[first-dyad-row]:H[last-dyad-row],"-999")))/[dyads]
```

1.2.2. Unequally-Weighted Indices

We are in the process of developing unequally-weighted indices such as a porosity index for dyads. Once incorporated into the database, these indices will be documented in BiG Dyads Code Book.

1.3. Querying

Structured Query Language or SQL (pronounced as one word or letters), as the name suggests, is a logical language used extensively to manage a relational database management system (RDBMS). This language is used to:

- "Create the database and table structures
- "Perform basic data management chores (add, delete and modify)
- "Perform complex queries to transform raw data into useful information."[2]

SQL is the medium through which we execute commands in a relational database. Since its logic follows that of 'natural language', i.e., languages like English, we can think of what we want from the database in the form of a statement and then almost translate it phrase by phrase into SQL. Then the database responds to us by either returning results in the form of rows retrieved from its tables or telling us that there are no records, in other words, no rows fulfilling our query conditions. (Needless to say, if a SQL statement is syntactically defective, an error message is shown, which is also a response in the sense that the database administrator knows they have to fix the way their query is formulated.)[3]

1.3.1. Formulating Queries: An Example

For example, if we want to find all the dyads in our database that are:

- Disputed

```
SELECT `H1_date_establishment_basic`, COUNT(`DyadID`) AS `No of Dyads Formed`
FROM `Dyads`
WHERE `Se1_border_disputed` = 1
AND `UNG_region` BETWEEN 3100 AND 3599
AND `F2_desert` != 1
```

```
AND `H1_date_establishment_basic` > 1900
AND `H1_date_establishment_basic` <= 1950
GROUP BY `H1_date_establishment_basic`
ORDER BY `H1_date_establishment_basic` ASC;
```

- Located in Asia
- Not crossed by a desert
- And were established sometime in the first half of the twentieth century;
- Further, we want to list the total number of these dyads by the year in which their current shape was established in ascending order.

These five conditions could be written out as the query you see below. In this sample, we have lines identified with numbers with lines 4 to 8 specifying those five conditions:
Line 3 shows we want borders that are disputed with the Boolean for Se1 variable marked 1 or 'true'; line 4 says that the selected dyads must be within the two mentioned numerical identifiers, which mark the limits of all Asian dyads in our database; line 5 states, in another binary choice, that the F2 variable for deserts must not equal 1—one might ask why we did not simply say it must equal 0 (or 'false'), and that is because we do not want to rule out values such as -999 which is used for maritime dyads in this case (for more info on this value and what it means, see 2.3. Missing and Inapplicable Data, and Cut-off Dates); lines 6 and 7 say the date of establishment of basic for the dyads in question must be greater than the year 1900 and lesser than or equal to the year 1950.

Now let us turn to the remaining lines: the first three lines mean we want to select two fields—i.e., H1 variable and DyadID—from the 'Dyads' table in our database. On line 1, we also want to count the number of DyadIDs that meet the specified criteria as 'summary rows' showing their total; further, we want to give a new column name (which is "No of Dyads Formed") to these summary rows for convenience. The last two lines mean, respectively, that the total number of dyads (as counted in line 1) must be grouped round each year in our result set and those years must be listed in ascending order. (The ORDER BY clause lists the result set in ascending order by default, which makes the use of the ASC keyword optional.)

Even though this query has to fulfil several conditions, it is still quite simple and straightforward since it is executed on one table only; more complex queries join several tables and are executed across many more dimensions, which is one of the features of a fully-fledged relational database as opposed to a spreadsheet with basic functions.

1.4. Database Platform

A relational database management system (RDBMS) is a software acting as an interface between the database and its administrators who can manage and query it. MySQL, the platform that BiG Dyads uses, is known as the "most popular Open Source SQL relational database management system." In other words, MySQL not only utilises SQL to query data but also "it is possible for anyone to use and modify the software" itself. Apart from popularity (which means there is

a greater community who use and can troubleshoot it) and being Open Source, criteria such as scalability and query-response performance were considered, for all of which MySQL scores very high. Last but not least, this platform was fully supported by the University of Victoria Infrastructure Services, which made opting for MySQL as the RDBMS of choice for BiG Dyads all the easier.

1.4.1. Spreadsheet Presentation

Any presentation of BiG Dyads datasets in the form of spreadsheets (such as Microsoft Excel) are, by necessity, unnormalised, i.e. they are not in First Normal Form (1NF).

1.5. Data Collection

BiG Dyads is comprised of theme-specific datasets which expand over time. Each dataset is named after the initial(s) of the theme to which it belongs. At the data collection level, each dataset exists in a tabular instance. This instance is used to verify results and return to the data collector for corrections, &c, if necessary. Later, all the datasets are imported into their respective tables and, after establishing relations between them, uploaded onto MySQL. Therefore, there is eventual consistency between tabular instances and related MySQL tables that form BiG Dyads.

1.6. Normal Forms

Database normalisation entails "creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency."[4] There are several levels of normalisation, aka Normal Forms, each of which builds upon the previous level. E.g., if the first set of rules is observed, the database is said to be in 'first normal form', with the second rule, it'll be in 'second normal form', and so on.

For our purposes here, which is first and foremost about gathering clean data, first Normal Form is the most important, at least for a start. NF1 rules, which needs to be borne in mind whilst collecting data, are as follows:

- Each table cell should contain a single value.
- Data in each column should be of the same type.
- Each record (i.e., the entirety of a row which consists of all the individual cells under their respective columns) needs to be unique.
- And, finally, the order of the columns should not matter; in other words, if you re-order columns, no information is lost, no error resulted.

We can also require ourselves to follow further, tailored rules in addition to NF1, because we want to make the database more tightly structured, efficient, and query-friendly. Some of these rules are:

1. When designing variables, try to keep numeric and textual variables separate (i.e., under different columns).
2. Also, turn fixed-value variables into binary, tertiary, &c, variables.
3. Another important measure that ought to be taken is establishing:

* A master list of keywords for the phenomena you study (e.g., geographical names) which is standard PLUS always updated everywhere;
* Controlled vocabulary, especially for textual (i.e., string) variables that are utilised consistently throughout the database.

1.7. Data Collection Do's and Don'ts

Things you must try to stay away from:

- No dirty data, because computers are not that intelligent (yet) to make sense of data contextually the way human beings can pick out signal from noise. Therefore, dirty data means inconsistent and/or poorly structured data: in other words, data that is carelessly treated. A legitimate question at this point would be: what does careful treatment of data entail?
- For one, DO NOT treat dataset tables as if it's another Excel sheet: e.g., no empty middle rows used as 'headers', &c;
- Also, DO NOT leave cells blank (see 2.3. Missing and Inapplicable Data, and Cut-Off Dates).
- DO NOT let pre-formatting stand: e.g., seemingly benign extra spaces, nonprinting characters, mark-up tags, and so forth, can all cause unnecessary headaches;
- And DO NOT leave duplicate rows. In order to rid your dataset of such duplicates, you can filter for unique values first and then confirm that the rows in question are indeed duplicates by checking everything.

And things you must always practise whilst you are collecting data:

- Be precise and, even more importantly, consistent: e.g., as mentioned in the previous slide, extra or, alternatively, missing spaces are both important and can result in problems;
- Consolidate same content under related columns;
- Treat Booleans (i.e., 1 and 0 instead of yes/Y and no/N) and other fixed-value columns meticulously;
- Next, you should spell-check and change text case to standard: e.g., using the find/replace function with text, formulae, formatting, or wildcard characters;
- Also, fix numbers and number signs as well as dates and times: e.g., convert numbers stored as text into digits or reformat dates so dashes, slashes, &c, are not mistaken for the same marks elsewhere;
- And transform and rearrange columns and rows when necessary (which is only applicable to flat or two-dimensional data).

1.7.1. Banned Characters

To preclude problems at the point of data insertion into the database or subsequent querying, adhere to using the characters 0–9 and a–z, A–Z as much as possible. Should you need to use non-Latin alphabets or other rare characters, consult with the database manager first. The following characters must be avoided from being inserted into dataset spreadsheets:

`?!#$&“”‘’()[]{}*|+,-./:;<=>@^`~`

Banned Characters	Alternative/Avoidance Strategy
apostrophe ‘	
comma ,	Rewrite string (non-digit) values to avoid commas; use dash if necessary
back slash \ & slash /	Insert dates with dashes as separators
dash -	Only avoid dashes at the beginning of values
double spaces	Avoid extra spaces, e.g. between sentences
parentheses ()	Avoid parentheses; use one or two dashes before and after

1.7.2. Shell Scripts

To help clean up and insert data, new shell scripts have been written that can be run by the database manager at the point of finalising datasets and entering them into the database. The code for the scripts and the instructions to run each can be found in Appendix B: Shell Scripts.

Endnotes

1 There are other, complementary datasets which are organised around objects other than the dyad. For instance, watersheds or peace parks have their own datasets in separate tables that are related to the dyad-based datasets.

2 Adrienne Watt and Nelson Eng, Database Design – 2nd Edition (Victoria, BC: BCcampus, 2014): Chapter 15. SQL Structured Query Language. Available at: https://opentextbc.ca/dbdesign01/chapter/sql-structured-query-language/. Accessed 27 March 2021.

3 The precursor to SQL was SEQUEL, or Standard English QUEry Language, which further highlights the affinity between SQL and natural language (primarily English but broadly any human language translatable into English).

4 Microsoft Docs: https://docs.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description#normalizing-an-example-table

2. Variables

2.1. List of Variables

UNG_region
Interregional_dyad
Intercontinental_dyad
UN_recognition
Media
F1_terrestrial
F2_desert
F3_river
F4_territorial_sea
F5_eez
F6_continental_shelf
F7_feature
F8_feature_type
F9_dyad_length
H1_year_establishment_basic
H2_treaty_establishment_basic
H3_year_adjustment
H4_treaty_adjustment
H5_year_historical_antecedent
H6_treaty_historical_antecedent
Se1_border_disputed
Se2_conflict
Se3_independence
G1_YYYYGDPs_Const2010USD
G2_YYYYratio_Const2010USD
G3_YYYYGDPs_PPPCurrentUSD
G4_YYYYratio_PPPCurrentUSD
Su1_number_watersheds
nSu2_name_watershed
jSu3_ID_watershed
nSu4_number_watershed_dyads
Su5_total_size_watersheds
Su6_number_peace_parks
nSu7_name_peace_park
jSu8_ID_peace_park
nSu9_number_peace_park_dyads
Su10_transboundary_protected_area
Su11_conservation_landscape_seascape
Su12_transboundary_conservation_migration_area
Su13_park_for_peace
Su14_total_size_protected_areas
Su15_year_legal_document
Su16_name_legal_document
FM1_number_mountains
nFM2_name_mountain
jFM3_ID_mountain
nFM4_number_mountain_dyads

2.2. Key Attributes

- DyadID

DyadID is an ordinal number starting at 1001 and assigned to each dyad record which had already been ranked according to UNG_region. This is used as the primary key[1] of the main table and foreign key[2] to the other individual dataset tables.

- Country_1, Country_2

The two countries the boundaries of which form a distinct dyad. Each dyad is unique (see DyadID above), and the only repetitions of dyads occurs when one dyad is terrestrial and the other maritime (which are assigned two distinct, consecutive DyadIDs). The order in which a dyad is defined—Country 1-Country 2 vs Country 2-Country 1— is not significant.

- UNG_region

This categorisation is based on the Geographic Regions of the United Nations publication "Standard Country or Area Codes for Statistical Use," originally published as Series M, No. 49 and now commonly referred to as the M49 standard.

- World
  - Africa: 1
    - Northern Africa: 11
    - ◊ Sub-Saharan Africa (12–15)
    - Eastern Africa: 12
    - Middle Africa: 13
    - Southern Africa: 14
    - Western Africa: 15
  - Americas: 2
    - ◊ Latin America and the Caribbean (21–23)
    - Caribbean: 21
    - Central America: 22
    - South America: 23
    - Northern America: 24
    - Antarctica: 29
  - Asia: 3
    - Central Asia: 31
    - Eastern Asia: 32
    - South-Eastern Asia: 33
    - Southern Asia: 34
    - Western Asia: 35
  - Europe: 3
    - Eastern Europe: 36
    - Northern Europe: 37 (including Channel Islands)
    - Southern Europe: 38
    - Western Europe: 39
  - Oceania: 4
    - Australia and New Zealand: 41
    - Melanasia: 42
    - Micronesia: 43
    - Polynesia: 44

- Interregional_dyad and Intercontinental_dyad

An Interregional_dyad is a Boolean variable that indicates whether a dyad crosses the boundaries of a UNG_region. If the first and second 2 digits of the UNG_region is non-repeating, then the dyad is inter-regional and Interregional_dyad equals 1.

A handful of inter-regional dyads are also intercontinental dyads when they cross not only the boundary of a region but also that of a continent. If the first and third digits of the UNG_region differ, then the dyad is intercontinental and Intercontinental_dyad equals 1.

- UN_recognition

This is a Boolean which determines whether the dyad is recognised by the United Nations (1) or not (0).

- Media

This column takes a URI pointing to where the media file of the dyad, upon availability, is stored as its variable.

2.3. Missing and Inapplicable Data, and Cut-Off Dates

Following code book best practice, coding missing and inapplicable data instead of leaving them empty or inaccurately supplying a zero ensures that any omission on the part of data collector or database manager results in a blank, i.e. uncoded, cell. In our Code Book, -888 stands for 'missing data' and -999 for 'inapplicable data': missing data means the data was unavailable for some (known or unknown) reason but not structurally missing; inapplicable data is data that is impossible to fill, e.g. the number of maritime dyads for a landlocked country, which can be put down as 0 but -999 is more accurate.

We have also used -1982 as a cut-off point for all maritime dyads that lack dates of establishment (the minus sign is used to differentiate this cut-off point from the year 1982 when the latter is used as a date and not a cut-off point). 1982 was chosen, because in that year the Law of the Sea establishing EEZ's came into effect. If two states have not already an agreement establishing a maritime boundary, we have made the decision to use that cut-off point as the establishment of a de facto maritime border. Therefore, the -1982 instances are where the states still have yet signed an official agreement, whereas the 1982 dates (or any other dates for that matter) are for dyads that have a treaty in the (specified) year.[3]

Endnotes

1 "A table typically has a column or combination of columns that contain values that uniquely identify each row in the table. This column, or columns, is called the primary key (PK) of the table and enforces the entity integrity of the table. Because primary key constraints guarantee unique data, they are frequently defined on an identity column" (Microsoft Docs).

2 "A foreign key (FK) is a column or combination of columns that is used to establish and enforce a link between the data in two tables to control the data that can be stored in the foreign key table. In a foreign key reference, a link is created between two tables when the column or columns that hold the primary key value for one table are referenced by the column or columns in another table. This column becomes a foreign key in the second table" (Microsoft Docs).

3 On EEZ's, see: https://stats.oecd.org/glossary/detail.asp?ID=884. For treaties that established them, see: https://treaties.un.org/Pages/ViewDetailsIII.aspx?src=TREATY&mtdsg_no=XXI-6&chapter=21&Temp=mtdsg3&clang=_en

3. Description of Datasets

3.1. Foundations of Dyads 16
3.2. Dates 18
3.3. Conflicts 20
3.4. Economic Inequality 21
3.5. Transboundary Watersheds 23
3.6. Peace Parks 25
3.7. Mountains 28

3.1. Foundations of Dyads Dataset (Theme: N/A)

Indicator 1: land/maritime border

- Variable 1 label: F1_terrestrial
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether a dyad is terrestrial (1) or maritime (0).

Indicator 2: desert/river

- Variable 1 label: F2_desert
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad crosses a desert (1) or not (0).

- Variable 2 label: F3_river
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad crosses a river (1) or not (0).

Indicator 3: maritime boundaries

- Variable 1 label: F4_territorial_sea
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether a maritime dyad is a territorial sea boundary (1) or not (0). (If the dyad is a land boundary, it is given -999.)

- Variable 2 label: F5_eez
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the maritime dyad is an exclusive economic zone boundary (1) or not (0). (If the dyad is a land boundary, it is assigned -999.)

- Variable 3 label: F6_continental_shelf
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the maritime dyad is a continental shelf boundary (1) or not (0). (If the dyad is terrestrial, it is assigned -999.)

Indicator 4: special features

- Variable 1 label: F7_feature
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether a given dyad contains a special feature, i.e. an enclave/exclave, zone of shared sovereignty, or some other exceptional feature, (1) or not (0).

- Variable 2 label: F8_feature_type
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: names the special feature corresponding to F7_feature.

Indicator 5: dyad length

- Variable 1 label: F9_dyad_length
  * Type: numeric
  * Values: natural numbers > o; -888, -999
  * Definition: gives the length in km of the given dyad. If the dyad is a maritime dyad with both a territorial sea section and an economic exclusive zone, then the length of the EEZ is given.

Design

- This dataset gives more detail on the nature of each dyad, allowing for the separation of dyads into smaller categories based on these features.

- Updating Schedule: this dataset needs to be updated only on an ad hoc basis, when changes to dyads change these aspects of the dyad.

Use and Limits

- Use: this dataset can be used to separate dyads with different characteristics from one another. It allows for the separate analysis of land dyads and maritime dyads, desert/river dyads, different types of maritime dyads, and dyads with special features. This dataset is helpful both in gaining more detail information about each dyad and in allowing for smaller categories of dyads for analysis.

- Limits: this dataset is limited by its binary nature. The aspects of the dyad are marked only as yes or no, leaving no room for extra information. This means that the dataset lacks detail—for example, the dataset does not record to what extent a dyad crosses a river or desert, only that it does so to some degree.

3.2. Dates Dataset (Theme: History [H])

Indicator 1: year of establishment of basic

- Variable 1 label: H1_year_establishment_basic
  * Type: date and time
  * Values: any year in the BCE/CE range; -888, -999
  * Definition: gives the year on which the essential shape of the current dyad was established.

- Variable 2 label: H2_treaty_establishment_basic
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: provides the name of the treaty or event corresponding to H1_year_establishment_basic.

Indicator 2: year of adjustments

- Variable 1 label: H3_year_adjustment
  * Type: date and time
  * Values: any year in the BCE/CE range; -888, -999
  * Definition: gives the year in which the last minor adjustment (i.e. an adjustment that does not fundamentally change the shape of the dyad) to the shape of the current dyad was effected.

- Variable 2 label: H4_treaty_adjustment
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: provides the name of the treaty or event corresponding to H3_year_adjustment.

Indicator 3: historical antecedent

- Variable 1 label: H5_year_historical_antecedent
  * Type: date and time
  * Values: any year in the BCE/CE range; -888, -999
  * Definition: gives the year of establishment of the historical antecedent of the current dyad in such cases where the modern dyad runs on much the same lines as the dyad between predecessor states.

- Variable 2 label: H6_treaty_historical_antecedent
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: provides the name of the treaty or event corresponding to H5_year_historical_antecedent.

Design

- The initial goal of the was to establish the date from which the current border of every dyad can be thought of as stable. Once research began, however, the complexity of the history of borders and the needs of the database made it clear that 3 categories of dates were needed: the establishment of the border, adjustments made to the border, and the historical antecedent of the border. Each of the first 3 categories has two columns, one for numeric and one for textual data.

- Updating Schedule: the indicators should be updated on an ad hoc basis, when changes to borders are made.

Use and Limits

- Use: this dataset can be used to establish the 'stability' of borders around the world. The older the date of establishment of a dyad is, the longer it can be said to have been stable. This can be useful to compare the stability of borders with other indicators such as conflict, inequality, transboundary agreements etc.

- Limits: the dataset is limited by the fact that it takes a synchronic approach and can only include dyads that still exist today. In cases where extinct dyads match up well with current dyads, extinct dyads can be included under the 'Historical Antecedents' indicator, but otherwise are not included in the database. This means that the dataset can only describe the world as it is today, and is not particularly useful for a diachronic analysis of the establishment of dyads.

3.3. Conflicts Dataset (Theme: Security [Se])

Indicator 1: border disputes

- Variable 1 label: Se1_border_disputed
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether one or both of the states on the dyad dispute(s) the position of the border (1), or not (0); and/or if the border has never been officially delimited (1), or not (0); and/or whether one or both of the states on the dyad dispute(s) the ownership of some portion/the entirety of the territory of the other state (1), or not (0).

Indicator 2: conflict

- Variable 1 label: Se2_conflict
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the current shape of a dyad arose out of a military conflict, violent independence, etc, (1) or not (0).

Indicator 3: independence

- Variable 1 label: Se3_independence
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad arose out of an independence/partition regardless of the presence of violence (1) or not (0).

Design

- These indicators were developed in order to track the origin of dyads today and their status vis-à-vis the states straddling them.

- Updating Schedule: the indicators should be updated on an ad hoc basis, when changes to borders are made, when disputes are resolved, new states created etc.

Use and Limits

- Use: this dataset can be used to establish the number of dyads currently disputed, the number of dyads that were created through conflict and/or independence. These numbers can be compared to other factors like the stability of the border, deaths on the border, transboundary agreements etc.

- Limits: this dataset is limited by its binary nature, leaving no room for descriptive detail. This means that the dataset lacks the ability to describe the nature of each data point. For example, the dataset does not give information as to the scale or intensity of border disputes or conflicts; it only records their existence. Furthermore, the dataset only records dyads currently in dispute, and misses dyads which were once in dispute but have since been resolved.

3.4. Economic Inequality Dataset (Theme: Governance [G])

Indicator 1: GDP per capita comparison on the basis of constant 2010 USD

- Variable 1 label: G1_YYYYGDPs_Const2010USD
  * Type: numeric
  * Values: the ratio must be written out in the form of GDPCountry_1:GDPCountry_2; -888, -999
  * Definition: expresses GDP per capita of Country_1 to Country_2 in year YYYY in constant 2010 USD.

- Variable 2 label: G2_YYYYratio_Const2010USD
  * Type: numeric
  * Values: rational numbers > 0; -888, -999
  * Definition: expresses the ratio of GDP per capita of a wealthier country to its poorer neighbour (so that it is always greater than or equal to one) in year YYYY.

Indicator 2: GDP per capita comparison on the basis of PPP Current USD

- Variable 1 label: G3_YYYYGDPs_PPPCurrentUSD
  * Type: numeric
  * Values: the ratio must be written out in the form of GDPCountry_1:GDPCountry_2; -888, -999
  * Definition: expresses GDP per capita of Country_1 to Country_2 (Purchasing Power Parity or PPP) in year YYYY in Current International USD.

- Variable 2 label: G4_YYYYratio_PPPCurrentUSD
  * Type: numeric
  * Values: rational numbers > 0; -888, -999
  * Definition: expresses ratio of GDP per capita (PPP) of a wealthier country to its poorer neighbour (so that it is always greater than or equal to one) in year YYYY.

Design

- Inequality, especially of its economic variety, is one of the most important topics in border studies. It intersects with many other border-related topics, such as migration and trade; therefore, it would be of considerable value to examine relationships e.g. between inequality and migration (and trade) flows, or inequality and border conflicts. The design of the variables in this dataset were taken from Elizabeth Staudt's Border Inequalities Around the World Dataset, which covered the years 1975, 1980, 1990, 2000, and 2014. We updated the data to the latest available at the time—which was the year 2017—from the sources Staudt had used.

- Updating Schedule: this depends

on research questions and other variables of interest to ensure there are no discrepancies in the period of our variables: e.g. if we examine the relationship between inequality and migration flows, both must come from comparable time periods (not, say, inequality in 2019, on one hand, and migration flows from the 1990s, on the other hand). Updating very often, e.g. every year, is likely to make little sense because inequality between countries does not change very quickly unless a major crisis happens and significantly impacts the GDP of a country. However, as a rule of thumb, should database management priorities and resources allow, it would be desirable to update the dataset variables not long after source data is updated.

Use and Limits

- Use: since inequality intersects with many other border-related topics, the indicators from this dataset can be crossed with other datasets to create interesting research questions. For more information on how to develop queries, see 1.3.

- Limits: as it stands, there is only the latest inequality data in the present dataset. However, data from multiple years (before, including those already present in Staudt's study, and, in the future, after) can be added to the dataset in order to conduct longitudinal studies.

3.5. Transboundary Watersheds Dataset (Theme: Sustainability [Su])

Indicator 1: dyad intersecting watersheds

- Variable 1 label: Su1_number_watersheds
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of watershed(s) that the dyad may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

- Variable 2 label: Su2_name_watershed
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: lists the watershed(s) that the dyad may intersect.

Indicator 2: combined size of watersheds

- Variable 1 label: Su3_total_size_watersheds
  * Type: numeric
  * Values: natural numbers > 0; -888, -999
  * Definition: gives in km^2 the total size of all the watershed(s) that the dyad may intersect.

Junction[1] indicator 3: watershed ID

- Variable 1 label: jSu4_ID_watershed
  * Type: incremental numeric
  * Values: natural numbers > 0
  * Definition: identifies an N number of watersheds incrementally, starting at 1 (1, 2, 3, ..., N).

Non-dyad-based[2] indicator 4: dyads intersecting a watershed

- Variable 1 label: nSu5_name_watershed
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: gives the watershed name.

- Variable 2 label: nSu6_number_watershed_dyads
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of dyads the watershed may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

Design

- An important subset of the theme of sustainability in the BiG project is transboundary watersheds. Finding a central list of well-established watersheds was central to the data collection for this subset. The UN Global Compact provided a major

transboundary dataset that was used as a jumping-off point for the present dataset.

- Updating Schedule: since the nature of transboundary watersheds is relatively stable, revisiting them every 10 years seems reasonable. The clear exception would be any change to the dyads that intersect watersheds, which can be revised on an individual basis.

Use and Limits

- Use: this dataset can be used individually or in conjunction with other Su-datasets to explore the theme of sustainability. Also, the indicators from this dataset can be crossed with other datasets to create interesting research questions. For more information on how to develop queries, see 1.3.

- Limits: N/A.

Endnotes
1 A 'junction' indicator bridges between two database tables (e.g. Dyads table and a non-dyad -based table) by pairing their primary keys, hence the prefix 'j'.
2 A 'non-dyad-based' indicator/variable takes an object other than the dyad (e.g. a watershed or peace park) as its main principle of organisation, hence the prefix 'n'.

3.6. Peace Parks Dataset (Theme: Sustainability [Su])

Indicator 1: dyad intersecting further peace parks

- Variable 1 label: Su7_number_peace_parks
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of peace park(s) that the dyad may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

- Variable 2 label: Su8_name_peace_park
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: lists the peace park(s) that the dyad may intersect.

Indicator 2: typology of transboundary conservation areas

- Variable 1 label: Su9_transboundary_protected_area
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad intersects a transboundary protected area (1) or not (0).

- Variable 2 label: Su10_conservation_landscape_seascape
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad intersects a transboundary conservation landscape/seascape (1) or not (0).

- Variable 3 label: Su11_transboundary_conservation_migration_area
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad intersects a transboundary conservation migration area (1) or not (0).

- Variable 4 label: Su12_parks_for_peace
  * Type: binary numeric
  * Values: 0, 1; -888, -999
  * Definition: determines whether the dyad intersects the special designation 'parks for peace' (1) or not (0).

Indicator 3: combined size of protected areas

- Variable 1 label: Su13_total_size_protected_areas
  * Type: numeric
  * Values: natural numbers > 0; -888, -999
  * Definition: states in km^2 the total size of all the protected areas that the dyad may intersect.

Indicator 4: formal agreement

- Variable 1 label: Su14_year_le-

gal_document
  * Type: date and time
  * Values: any year in the BCE/CE range; -888, -999
  * Definition: states the year of any informal/soft legal document ("arrangement") or a formal/hard legal document ("agreement")—including memoranda of understanding, international treaties, international agreements, international conventions, &c—for the creation, governance, &c, of protected areas (e.g. watersheds, parks, reserves of biosphere) that may exist on the dyad.

- Variable 2 label: Su15_name_legal_document
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: states the title of the formal agreement corresponding to Su14_year_formal_agreement.

Junction indicator 5: peace park ID

- Variable 1 label: jSu16_ID_peace_park
  * Type: incremental numeric
  * Values: natural numbers > 0
  * Definition: identifies an N number of peace parks incrementally, starting at 1 (1, 2, 3, ..., N).

Non-dyad-based indicator 6: further dyads intersecting the peace park

- Variable 1 label: nSu17_name_peace_park
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: states the peace park name.

- Variable 2 label: nSu18_number_peace_park_dyads
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of dyad(s) that the peace park may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

Design

- Peace parks forms another subset of the theme of sustainability. The initial phase of the data collection for this dataset focussed on parks that were intersected with dyads. Classifying different transboundary parks, which was a critical first step, was based on IUCN definitions in their report Transboundary Conservation: A Systemic and Integrated Approach.

- Updating Schedule: since the nature of peace parks is relatively stable, revisiting them every 10 years seems reasonable. The clear exception would be any change to the dyads that intersect watersheds, which can

be revised on an individual basis.

Use and Limits

- Use: this dataset can be used individually or in conjunction with other Su-datasets to explore the theme of sustainability. Also, the indicators from this dataset can be crossed with other datasets to create interesting research questions. For more information on how to develop queries, see 1.3.

- Limits: N/A.

3.7. Mountains Dataset (Theme: Foundations-Mountains [FM])

Indicator 1: dyad intersecting mountain(s)

- Variable 1 label: FM1_number_mountains
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of mountains/mountain ranges that the dyad may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

Non-dyad-based indicator 2: mountain(s) intersecting a dyad

- Variable 1 label: nFM2_name_mountains
  * Type: string
  * Values: unbounded; -888, -999
  * Definition: lists the mountain(s) that the dyad may intersect.

Junction indicator 3: mountain ID

- Variable 1 label: jFM3_ID_mountain
  * Type: incremental numeric
  * Values: natural numbers > 0
  * Definition: identifies an N number of watersheds incrementally, starting at 1 (1, 2, 3, ..., N).

Non-dyad-based indicator 4: dyads intersecting a watershed

- Variable 1 label: nFM4_number_mountain_dyads
  * Type: numeric
  * Values: natural numbers ≥ 0; -888, -999
  * Definition: counts the total number of dyads the mountain may intersect. This particular variable can function as a combined Boolean (n=0 or n>0) + n' counter (n>0 ⟶ n=n').

Design

- The current formulation is only the beginning for this dataset as more information-rich indicators are yet to be added to the present list.

- Updating Schedule: since the geography, topography, and other characteristics of mountains are relatively stable, revisiting this dataset every 10 years seems

reasonable. The clear exception would be any change to the dyads that intersect mountains, which can be revised on an individual basis.

Use and Limits

- Use: this dataset can be used individually or in conjunction with other F and/or Su-datasets to explore the theme of sustainability, amongst others. Also, the indicators from this dataset can be crossed with other datasets to create interesting research questions. For more information on how to develop queries, see 1.3.

- Limits: N/A.

Appendices

Appendix A: Country Alphabetical Order Number

The name in parenthesis is to be used.

- Abkhazia 001
- Afghanistan 002
- Albania 003
- Algeria 004
- Andorra 005
- Angola 006
- Antigua and Barbuda 007
- Argentina 008
- Armenia 009
- Artsakh, Republic of 010
- Australia 011
- Austria 012
- Azerbaijan 013
- Bahamas 014
- Bahrain 015
- Bangladesh 016
- Barbados 017
- Belarus 018
- Belgium 019
- Belize 020
- Benin 021
- Bhutan 022
- Bolivia 023
- Bosnia and Herzegovina 024
- Botswana 025
- Brazil 026
- Brunei 027
- Bulgaria 028
- Burkina Faso 029
- Burundi 030
- Cambodia 031
- Cameroon 032
- Canada 033
- Cape Verde 034
- Central African Republic 035
- Chad 036
- Chile 037
- China 038
- Colombia 039
- Comoros 040
- Costa Rica 041
- Cote d'Ivoire 042
- Croatia 043
- Cuba 044
- Cyprus 045
- Czechia 046
- Democratic Republic of the Congo (DR Congo) 047
- Denmark 048
- Djibouti 049
- Dominica 050
- Dominican Republic 051
- East Timor 052
- Ecuador 053
- Egypt 054
- El Salvador 055
- Equatorial Guinea 056
- Eritrea 057
- Estonia 058
- Ethiopia 059
- Fiji 060
- Finland 061
- France 062
- Gabon 063
- Georgia 064
- Germany 065
- Ghana 066
- Greece 067
- Grenada 068
- Guatemala 069
- Guinea 070
- Guinea-Bissau 071
- Guyana 072
- Haiti 073
- Honduras 074
- Hungary 075
- Iceland 076
- India 077
- Indonesia 078
- Iran 079
- Iraq 080
- Ireland 081
- Israel 082
- Italy 083
- Jamaica 084
- Japan 085
- Jordan 086
- Kazakhstan 087
- Kenya 088
- Kiribati 089
- Kosovo 090
- Kuwait 091
- Kyrgyzstan 092
- Laos 093
- Latvia 094
- Lebanon 095
- Lesotho 096
- Liberia 097
- Libya 098
- Liechtenstein 099
- Lithuania 100
- Luxembourg 101
- Macedonia (North Macedonia) 102
- Madagascar 103
- Malawi 104
- Malaysia 105
- Maldives 106
- Mali 107
- Malta 108
- Marshall Island 109
- Mauritania 110
- Mauritius 111
- Mexico 112
- Micronesia 113
- Moldova 114
- Monaco 115
- Mongolia 116
- Montenegro 117
- Morocco 118
- Mozambique 119
- Myanmar 120
- Namibia 121
- Nauru 122
- Nepal 123
- Netherlands 124
- New Zealand 125
- Nicaragua 126
- Niger 127
- Nigeria 128
- North Korea 129
- Northern Cyprus 130
- Norway 131
- Oman 132
- Pakistan 133
- Palau 134

- Palestine 135
- Panama 136
- Papua New Guinea 137
- Paraguay 138
- Peru 139
- Philippines 140
- Poland 141
- Portugal 142
- Qatar 143
- Republic of the Congo (Congo) 144
- Romania 145
- Russia 146
- Rwanda 147
- Saint Kitts and Nevis 148
- Saint Lucia 149
- Saint Vincent and the Grenadines 150
- Samoa 151
- San Marino 152
- Sao Tome and Principe 153
- Saudi Arabia 154
- Senegal 155
- Serbia 156
- Seychelles 157
- Sierra Leone 158
- Singapore 159
- Slovakia 160
- Slovenia 161
- Solomon Islands 162
- Somalia 163
- Somaliland 164
- South Africa 165
- South Korea 166
- South Ossetia 167
- South Sudan 168
- Spain 169
- Sri Lanka 170
- Sudan 171
- Suriname 172
- Swaziland (Eswatini) 173
- Sweden 174
- Switzerland 175
- Syria 176
- Taiwan 177
- Tajikistan 178
- Tanzania 179
- Thailand 180
- The Gambia (Gambia) 181
- Togo 182
- Tonga 183
- Transnistria 184
- Trinidad and Tobago 185
- Tunisia 186
- Turkey 187
- Turkmenistan 188
- Tuvalu 189
- UAE 190
- Uganda 191
- Ukraine 192
- UK 193
- USA 194
- Uruguay 195
- Uzbekistan 196
- Vanuatu 197
- Vatican City 198
- Venezuela 199
- Vietnam 200
- Western Sahara 201
- Yemen 202
- Zambia 203
- Zimbabwe 204

Appendix B: Shell Scripts

Script 1: row_column_match.sh

This shell script can be used to clean up CSVs or other array files that contain datasets. Using a code editor, the code below must be saved as a .sh file and then run in Terminal. The appropriate commands will follow. If there are any matches for the string(s) searched, a .txt file will be created in the directory you are running Terminal from.

```
#!/usr/bin/env bash
echo $BASH_VERSION

# Prompt for input: 1. enter file name or path that you want searched; 2. enter the literal or regex string
echo File name or path to find matches in:
read file
echo Literal or regex string to find:
read string

# Define variable and test if any matches are to be found; if not, notification is sent to terminal, but if matches exist, their row numbers (as summary rows) and individual column numbers will be output to a .txt file in the home directory. NB: you need to escape minus symbol with brackets, [-], so that it's not confused with an invalid grep option!
matchesFound=$(cat $file | grep -E -c "$string")
if [ $matchesFound -eq 0 ];
then
  echo "No matches exist."
else
  printf "Summary Row No: \n`awk -v awkvar="$string" '$0 ~ awkvar{print NR}' $file`" > results_for_$string.txt
  printf "\nInstance Column No: \n`awk -v awkvar="$string" -F"," '{for(i=1;i<=NF;i++){if ($i ~ awkvar){print i}}}' $file`" >> results_for_$string.txt
fi
```

Script 2: sql_insert.sh

The shell script below can be used to insert data into the database. This complete version of the script specifies not only the values of rows but also the matching column names into which those values must be entered. Similar to the previous script, the code below must be saved as a .sh file and then run in Terminal. The appropriate commands will be printed on your Terminal screen. Once the desired SQL code has been output to a .txt file, the content can be copied and pasted into MySQL query box and then executed from there.

```
#!/usr/bin/env bash
echo $BASH_VERSION

# Prompt for input and: 1. enter CSV path to be imported into DB; 2. its equivalent table name in the DB
```

```
echo Path to CSV:
read CSV_file
echo DB table name to import into:
read DB_table

# Create .txt file that will contain SQL INSERT STATEMENT and enter DB table name
echo "INSERT INTO $DB_table (" > SQL_INSERT_$DB_table.txt

# List out CSV header as INSERT STATEMENT column names and append to .txt file
echo "`head -n 1 $CSV_file`)" >> SQL_INSERT_$DB_table.txt

# Auto-quote string columns in the .txt file, leaving other columns intact
awk -F, 'OFS=FS {for (i=1;i<=NF;i++) {if (match($i, /^[0-9.-]+$/)==0) {printf "\"" $i "\""} else
{printf $i}; if (i<NF) printf OFS}; printf "\n"}' $CSV_file > temp.txt
echo "VALUES" >> SQL_INSERT_$DB_table.txt

# read-while loop to populate INSERT STATEMENT row values from CSV (2nd row to the end)
and replace final comma with semicolon for those RDBMS's that require a concluding semico-
lon at the end of SQL STATEMENT
while read line
do
 echo "($line),"
done < <(tail -n +2 temp.txt) >> SQL_INSERT_$DB_table.txt && sed -i '' '$ s/.$/;/' SQL_IN-
SERT_$DB_table.txt

# Delete temporary .txt file that contained auto-quoted string values
rm temp.txt
```

Script 3: sql_insert_wocol.sh

The script # 3, similar to sql_insert.sh, can be used to insert data into the database. However, it specifies only the values of rows without their matching column names.

```
#!/usr/bin/env bash
echo $BASH_VERSION

# Prompt for input and: 1. enter CSV path to be imported into DB; 2. its equivalent table name
in the DB
echo Path to CSV:
read CSV_file
echo DB table name to import into:
read DB_table

# Create .txt file that will contain SQL INSERT STATEMENT and enter DB table name
echo "INSERT INTO $DB_table VALUES" > SQL_INSERT_$DB_table.txt
```



```
# Auto-quote string columns in the .txt file, leaving other columns intact
awk -F, 'OFS=FS {for (i=1;i<=NF;i++) {if (match($i, /^[0-9.-]+$/)==0) {printf "\"" $i "\""} else
{printf $i}; if (i<NF) printf OFS}; printf "\n"}' $CSV_file > temp.txt

# read-while loop to populate INSERT STATEMENT row values from CSV (2nd row to the end)
and replace final comma with semicolon for those RDBMS's that require a concluding semico-
lon at the end of SQL STATEMENT
while read line
do
 echo "($line),"
done < <(tail -n +2 temp.txt) >> SQL_INSERT_$DB_table.txt && sed -i '' '$ s/.$/;/' SQL_IN-
SERT_$DB_table.txt

# Delete temporary .txt file that contained auto-quoted string values
rm temp.txt
```

Appendix C: Sources

Appendix C is a spreadsheet available as an Excel (.xlsx). To receive a compressed file (.zip) containing the spreadsheet together with BiG Dyads Code Book (.pdf) and Appendix B scripts (.sh), please send an e-mail to borders@uvic.ca with 'code book files' in the subject line.

This page intentionally left blank.

This page intentionally left blank.

Social Sciences and Humanities Research Council of Canada

Conseil de recherches en sciences humaines du Canada

Canada

University of Victoria

University of Lethbridge

TRENT UNIVERSITY

University of Regina

UNIVERSITÉ DE SHERBROOKE

Carleton UNIVERSITY

ENAP

LAURIER Inspiring Lives.

UQÀM

Co-funded by the Erasmus+ Programme of the European Union