Title

CoCoTools: open-source software for building connectomes using the CoCoMac anatomical database.

Permalink
https://escholarship.org/uc/item/7xr225f7

Journal
Journal of cognitive neuroscience, 26(4)

ISSN
0898-929X

Authors
Blumenfeld, Robert S
Bliss, Daniel P
Perez, Fernando
et al.

Publication Date
2014-04-01

DOI
10.1162/jocn_a_00498

Journal of
Cognitive Neuroscience

CoCoTools: Open-source software for building connectomes using the CoCoMac anatomical database

Journal:	*Journal of Cognitive Neuroscience*
Manuscript ID:	JOCN-2012-0726.R3
Manuscript Type:	Original
Date Submitted by the Author:	14-Sep-2013
Complete List of Authors:	Blumenfeld, Robert; UC Berkeley, Helen Wills Neuroscience Institute Bliss, Daniel; UC Berkeley, Helen Wills Neuroscience Institute Perez, Fernando; UC Berkeley, Helen Wills Neuroscience Institute D'Esposito, Mark; UC Berkeley, Helen Wills Neuroscience Institute
Keywords:	Neuroanatomy, Connectomics, Primate, Methods, Connectivity

CoCoTools: Open-source software for building connectomes using the CoCoMac anatomical database

Robert S. Blumenfeld[1,*,†], Daniel P. Bliss[1,†], Fernando Perez[1], Mark D'Esposito[1]

[1] Helen Wills Neuroscience Institute, University of California, Berkeley, CA 94720

*Address for correspondence:
Robert S. Blumenfeld
Helen Wills Neuroscience Institute
132 Barker Hall
University of California at Berkeley
Berkeley, CA 94720, USA
Email: rsblume@berkeley.edu

Keywords: CoCoMac, macaque, graph theory, connectivity, connectome, anatomy,

RUNNING HEAD: CoCoTools: open-source connectome software

Word count:

Citations:

†Contributed equally, listed in reverse alphabetical order

Acknowledgements: We would like to thank Rembrandt Bakker for his advice and for adding the region_centers functionality to CoCoMac 2.0. Secondly, we would like thank Gleb Bezgin for his help early in the project and Olaf Sporns for his very helpful comments and discussion. We are grateful for the thorough and valuable feedback we received from our two anonymous reviewers.

ABSTRACT

Neuroanatomical tracer studies in the non-human primate macaque monkey are a valuable resource for cognitive neuroscience research. These data ground theories of cognitive function in anatomy and, with the emergence of graph-theoretical analyses in neuroscience, there is high demand for these data to be consolidated into large-scale connection matrices ("macro-connectomes"). Because manual review of the anatomical literature is time-consuming and error-prone, computational solutions are needed to accomplish this task. Here we describe the "CoCoTools" open-source Python library, which automates collection and integration of macaque connectivity data for visualization and graph theory analysis. CoCoTools both interfaces with the CoCoMac database, which houses a vast amount of annotated tracer results from 100 years (1905-2005) of neuroanatomical research, and implements coordinate-free registration algorithms, which allow studies that use different parcellations of the brain to be translated into a single graph. We show that using CoCoTools to translate all of the data stored in CoCoMac produces graphs with properties consistent with what is known about global brain organization. Moreover, in addition to describing CoCoTools' processing pipeline, we provide worked examples, tutorials, links to online documentation, and detailed appendices to aid scientists interested in using CoCoTools to gather and analyze CoCoMac data.

INTRODUCTION

Anatomical connectivity underlies every function in the brain and topological differences in connectivity between brain regions as well as differences in cytoarchitecture form the basis for how functions are differentiated in the brain (Brodmann, 1909; see Zilles & Amunts, 2010 for review). Therefore neuroanatomical studies that investigate the cytoarchitecture and white matter fiber-tract connections provide cognitive and systems neuroscience with invaluable information about the anatomical substrate of function. Moreover, as the interest in the field for examining the functions of large-scale networks has grown, so too has the demand for reliable large-scale data depicting the inter-regional anatomical connections of the primate brain (i.e. the macro-connectome). Anatomists have characterized many of the macaque monkey brain's long-range connections over the last century in tracer experiments. In these studies, molecular or viral tracers are injected into an area of interest. These tracers are taken up and actively transported by neurons. In the case of anterograde tracer, molecules or viruses enters the somas of cells in the injected area and are actively transported to their presynaptic terminals, labeling areas to which they project. In the case of retrograde tracer, molecules or viruses enters presynaptic terminals and are transported to the soma, labeling areas that send information to the injected area (Köbbert et al., 2000; Kuypers & Ugolini, 1990).

For large-scale descriptions of macaque cortex[1] to be constructed from these data, results from many individual studies – the vast majority of which investigate just a small part of the brain – must be integrated. Achieving such integration requires confronting the "parcellation problem" (Stephan & Kötter, 1998) – the fact that across original reports, incompatible mapping schemes and naming conventions for similar cytoarchitectonic regions are abundant and that aside from one published atlas (Paxinos, Huang, & Toga, 2000), none of these brain parcellations are referenced to spatial coordinates. Thus as a result, there is no straightforward way to spatially align the vast majority of results from the literature. Though progress has been made in confronting this "parcellation problem" – for example, the landmark work by Felleman and

[1] The methods discussed here were designed to handle the registration of cortical areas and connections. A discussion of subcortical nuclei is beyond the scope of this manuscript. However, data from subcortical nuclei are available in CoCoMac, and the methods discussed here are equally capable of gathering, registering, plotting and analyzing intra-subcortical and cortical-subcortical connections.

Van Essen (Felleman & Van Essen, 1991; see also Young, 1993) – it remains a major obstacle. Advances with the monkey macro-connectome lag behind the national (e.g., "Human Connectome Project," 2012) and international (e.g., "International Neuroinfomatics Coordinating Facility," 2012) efforts to define the mouse and human macro-connectomes.

The CoCoMac (*Collation of Connectivity on the Macaque Brain*) database (Kötter, 2004; Stephan et al., 2001) and the Objective Relational Transformation (ORT) algorithm (Kötter & Wanke, 2005; Stephan & Kötter, 1998, 1999; Stephan et al., 2001; Stephan, Zilles, & Kötter, 2000) were designed to provide computational solutions for the "parcellation problem". CoCoMac houses results from over 400 original reports (from Brodmann's groundbreaking 1905 work to studies published as recently as 2005), which supply more than 200 different mapping schemes and 30,000 anatomical connections. CoCoMac data fall into two categories: 1) stated spatial relationships between brain regions (mapping statements) and 2) stated presence or absence of white-matter connections between brain regions (connectivity statements). Critically, CoCoMac mapping and connectivity statements express qualitative, rather than quantitative, spatial relationships between regions and are therefore amenable to algorithmic solutions that use qualitative spatial reasoning. Thus, combined with these algorithms, the data stored in CoCoMac can, in theory, serve as the basis for large-scale topological macro-connection matrices (i.e. "graphs") of the macaque brain.

However, despite the effort to curate CoCoMac, the resources available for accessing its data are inadequate. Users can browse the website for the database (http://cocomac.org) but several important data fields are difficult to find and sometimes not clearly stated, such as which area was injected for each reported connection in each tracer experiment. Furthermore, data are provided only in HTML and XML format, neither of which is amenable to direct analysis, and large queries to the server often crash or take a prohibitively long time to complete. An improved version of the CoCoMac web interface is under development at the INCF (CoCoMac 2.0: Bakker, Wachtler, & Diesmann, 2012). When it is complete, the CoCoMac 2.0 website may address many of these limitations of the CoCoMac 1.0 site, but as of this writing, CoCoMac 2.0 is not fully operational.

The other tool designed to help solve the parcellation problem, the ORT algorithm, is a spatial reasoning algorithm designed to register the disparate CoCoMac data into a consistent mapping space (Stephan & Kötter, 1998, 1999; Stephan et al., 2000, 2001; Kötter & Wanke, 2005). ORT uses the principles of region connection calculus (Cohn, Bennett, Gooday, & Gotts, 1997), a qualitative spatial reasoning system, to translate connections between two areas in a given input mapping space to a user-chosen output mapping space. First, ORT uses CoCoMac's logical mapping statements to define the qualitative spatial relationships between the input map areas and those in the output map. ORT then applies a set of logical transformation rules to the connections in the input-space to define one or more connections within the output space. Figure 1 illustrates how ORT combines mapping data — in the form of relation codes (RCs, left column) — and connectivity data — extension codes (ECs) — to infer connectivity in a new mapping space. In this hypothetical example, the goal is to determine whether a connection from Region E to Region C exists, but no published reports have directly tested for this. Other studies, however, have tested for the connection between Region E and a Region A that has been stated to cover some of the same patch of cortex as C, but in a different mapping scheme. We can use ORT to translate the E-to-A connection to an E-to-C one, because A encompasses C (RC=L) and when the E-to-A connection was tested, all of A was filled with tracer (EC=C). Drawing a conclusion for this simplified scenario does not require the complicated set of steps that comprise ORT, but the advantage of the automated algorithm becomes clear when the goal is to translate all 30,000 connections in CoCoMac. Unfortunately, an implementation of ORT has not been made available to the public, and descriptions of ORT in the literature are inconsistent, incomplete, and make assumptions that do not hold with CoCoMac data. These will be described in detail below.

In summary, although CoCoMac and ORT have laid the groundwork for macaque connectomics, new software is required to make the neuroanatomical data and coordinate-free registration algorithms amenable to analysis by the broader neuroscientific community. In the following sections we describe the CoCoTools library, a set of open-source tools written in Python for querying, storing and analyzing CoCoMac data (http://cocotools.github.io). These tools are extensible, allowing for community development and customization for diverse research objectives. CoCoTools performs, in an automated way, CoCoMac queries, local caching of data to speed up subsequent access, and parsing of CoCoMac mapping and connectivity output to build user-friendly graph objects. CoCoTools includes two coordinate-free registration methods: a version of ORT that utilizes the best features from all descriptions published by Kötter, Stephan, and colleagues (Kötter & Wanke, 2005; Stephan & Kötter, 1998, 1999; Stephan et al., 2001, 2000) and a version that modifies several aspects of the algorithm in order to avoid problematic assumptions (modified ORT, or mORT). These algorithms are implemented using directed graphs based on those in the NetworkX Python library (Hagberg, Swart, & Schult, 2008). These directed graphs can be further studied with the wealth of graph theory algorithms in NetworkX, as well as some absent from NetworkX that we have included in our package.

CoCoTools will be of use both to cognitive neuroscientists seeking to understand the connections of a particular brain region, computational neuroscientists interested in building large-scale simulations of the macaque brain (Ghosh, Rho, McIntosh, Kötter, & Jirsa, 2008; Knock et al., 2009), as well as scientists that use graph theory to study the large-scale topology of the brain. We provide IPython Notebooks (Perez & Granger, 2007) as part of CoCoTools (see http://cocotools.github.io) that contain executable versions of the CoCoTools demonstrations we present below. We encourage readers to use the notebooks as a tutorial, to modify them, and to use them as a starting point for building their own macaque macro-connection matrices.

METHODS

High-level Conceptual model and the problem of coordinate-free registration

This manuscript puts forward software and algorithmic methods for registering anatomical connectivity datasets from disparate mapping schemes to form large-scale connection matrices or graphs. The fundamental challenge in registering these datasets is that they are not based in a standard coordinate space, so Euclidean spatial information cannot be used. Our method, based on the seminal work by Stephan and Kötter, addresses this challenge and accomplishes coordinate-free registration by utilizing the logic of "region connection calculus" (Cohn et al., 1997). First, the spatial mapping relationships and anatomical connections in the literature are restated in a relative and qualitative, rather than quantitative spatial, manner (see Kötter and Wanke, 2005, for a discussion of this distinction). This turns the spatial registration problem into a relational logic problem that can be solved by a few decision rules formalized in an algorithm. At its conceptual core, this algorithm performs two operations. First it uses the mapping data – the logical relationships among brain regions – to bring into alignment the overlapping cortex shared by different parcellation schemes. After these two parcellation schemes are aligned, each connection expressed in one parcellation scheme is passed through a decision algorithm that tests whether there is sufficient evidence to conclude that tracer expressed in the original space would also be present in the target space. This process is repeated iteratively for every connection and mapping scheme supplied to the algorithm. The final output is a set of connections (which can be expressed as a matrix or graph) in a user-assigned target space.

How CoCoTools provides a solution

In practice, to accomplish this translation between maps, our software provides high-level commands that allow the user to gather, pre-process, and apply logical rules to CoCoMac mapping and connectivity data. These data are already stated in logical/relational terms, thanks to the meticulous work of Kötter and his lab. In CoCoTools, first connectivity and mapping data are downloaded from the CoCoMac website (http://cocomac.org). Then, pre-processing brings the data into a form and level of robustness suitable for coordinate-free registration algorithms to complete without error. Throughout the CoCoTools pipeline, the data take the form of graphs, which express the mapping relationships and anatomical connectivity between brain regions in a manner convenient for analysis. After the data have been pre-processed, the logical decision rules for coordinate-free registration can be applied. As mentioned above, these rules determine whether tracer expressed in the input space would also be present in the target space, were the two spaces to be brought into alignment with each other. As we will detail below, we have implemented two different approaches to this registration process: a version of the ORT algorithm based on the most complete and accurate descriptions published by Kötter, Stephan, and colleagues (Kötter & Wanke, 2005; Stephan & Kötter, 1998, 1999; Stephan et al., 2001, 2000) and a version that modifies several aspects of the algorithm in order to avoid problematic assumptions (modified ORT, or mORT), resulting in a conceptually simpler algorithm. In addition, CoCoTools provides graphing utilities that facilitate visualization of the results.

The CoCoMac Database:

CoCoMac data are statements drawn from the neuroanatomical literature about brain regions defined based on cytoarchitectonic properties. Each brain region in CoCoMac is linked to a specific study and coded by a unique acronym that includes the last initials of the authors, the last two digits of the year of publication, and the name of the region. For example, Brodmann's (Brodmann, 1905) Area 9 is coded as 'B05-9', and Petrides and Pandya's (Petrides & Pandya, 1994) Area 46 is coded as PP94-46.

CoCoMac stores mapping and connectivity statements. Mapping statements describe the spatial relationship between two regions, and each one is assigned a relation code (RC), which takes one of four values (see Figure 1): "I": the regions are identical. "L": the first region is larger than and fully contains the second. "S": the first region is smaller than and is fully contained by the second. "O": the two regions overlap partially. Connectivity statements describe the extent of staining in two regions that resulted from injecting tracer in one of them. These data are captured in extension codes (ECs), for which there are four possible values (see Figure 1): "C": staining covered the region completely. "P": staining covered only part of the region. "X": staining exists in the region, but with unknown extent. "N": staining covered no part of the region. "U": whether staining covered the region is unknown. Connectivity statements are expressed as an ordered pair of ECs, one for the source region (i.e. where the somas of the stained neurons are) and one for the target region (i.e. where the axons of the stained neurons terminate). Whether anterograde or retrograde tracer was used to test for each connection reported in CoCoMac cannot be downloaded in an automated way from the current form of the database, and even after manual searches a clear answer cannot always be obtained. Thus, although the direction of connectivity is specified in the downloadable data, which region or regions were injected to achieve each result cannot always be inferred. This limits the extent to which anatomical connections can be translated, as will be explained below.

Along with RCs and ECs, CoCoMac stores precision of description codes (PDCs), which indicate the level of detail provided in the original paper. CoCoMac also reports connectivity strength for the subset of studies that collected this information (e.g. Walker, 1940). For further explanation of the codes in CoCoMac, readers are encouraged to review the introduction in Stephan et al. (2001).

The CoCoMac 1.0 database, hosted at http://cocomac.org, provides weak support for automated download of statements from many maps at once. Such download requires knowledge of an obscure coding syntax, the initially returned data must be reorganized into a format suitable for analysis, and the server is prone to failure. Furthermore, CoCoMac does not implement any kind of inter-map registration, such as the ORT algorithm. CoCoTools redresses these issues, as will be discussed below.

Overview of Coordinate-free Registration, ORT and mORT:

Alongside the development of the CoCoMac database, Kötter, Stephan and colleagues designed a formal algorithm that utilizes the qualitative spatial reasoning logic of region connection calculus (Cohn et al., 1997) as well as graph-theoretical path finding metrics (Floyd, 1962) for solving the parcellation problem using the relational syntax of CoCoMac's mapping data (RCs) and connectivity data (ECs). The approach that they developed and described in many published works (Kötter & Wanke, 2005; Stephan & Kötter, 1998, 1999; Stephan et al., 2001) is termed Objective Relational Transformation (ORT). The goal of ORT is to use ECs and RCs to translate anatomical connections from one input map (or between two input maps) to connections between regions in an output map. For the purpose of providing a general overview here (see Appendix A for a formal description and Appendix B for additional discussion), we present ORT as having two discrete steps: gathering data and applying the rules of translation. Given an anatomical connection between two areas in an input space ("map A") and a desired output space ("map B") that we want to register to, ORT gathers the necessary RCs and ECs to define the relational mapping between the relevant source and target areas of map A and map B. Then, ORT applies a set of decision rules to the gathered ECs and RCs, to determine the translated EC (or ECs) in map B. This process is performed separately for each connection passed to ORT.

Our implementation of ORT combines features from several publications. Thus, although CoCoTools' ORT holds true to the core logic set out by Stephan and Kötter, it is different from each individual original published description. We follow most closely the logic of (Stephan & Kötter, 1998). Although similar to later iterations (e.g. Kötter & Wanke, 2005; Stephan & Kötter, 1999; Stephan et al., 2000), this version gathers more mapping and connectivity information than the others to create a complete description of the shared space on standard cortex that areas in different parcellation schemes occupy. Our implementation does not specially treat cases where the injection has spread across multiple brain sites (c.f. Stephan et al., 2001), as we believe this issue is best handled not in the algorithm itself, but in the careful manual selection of data submitted to the algorithm.

It is important to note that ORT does not translate anatomical *connections* per se, but more precisely translates the extent of stain present in the source and target regions that result from an injection of tracer to one of these regions. The extent of staining in an anatomical study does not always indicate the extent of connectivity. Anatomists need to verify this relationship by injecting anterograde tracer in the origin and retrograde tracer in the termination point for each reported connection. Unfortunately, few studies perform this test for every one of their reported connections, and, as discussed above, the current layout of CoCoMac 1.0 makes this information difficult, and in some cases impossible, to obtain. Thus, without a tremendous amount of additional manual curation of results, ORT's treatment of staining as equivalent to connectivity is unjustified. All that can logically be concluded from the CoCoMac data amenable to automated download is whether each connection of interest is present, absent, or unknown. This updated logic forms the basis of mORT in CoCoTools.

Rather than use the full range of EC values to describe stain extent, mORT uses only three labels for connections: Present, Absent, and Unknown. Thus, mORT's approach is spatially more conservative, and requires a simpler region connection calculus that allows the source and target areas to be translated in one step. This causes mORT to properly handle cases that are mistranslated by ORT (see Appendix B for technical discussion). However, in practice, we have found that ORT's mistranslations often have only minor consequences with large datasets (see Results: *Demonstration of ORT - mORT Differences*). Thus, we include both algorithms in CoCoTools.

Overview of CoCoTools:

We have designed the CoCoTools project as a software library that provides streamlined, robust and flexible tools that scientists can use to systematically and efficiently mine the CoCoMac database in its current form. CoCoTools offers several important benefits to researchers: 1) it provides an easy-to-use interface that can perform large queries of the CoCoMac database, with local caching of results to speed up repeated access; 2) it parses the raw CoCoMac output, returned in XML format, into graph structures that retain metadata of the original response but are directly amenable to mathematical analysis; 3) it provides a documented, open-source processing pipeline and an implementation of the ORT algorithm and the alternative mORT algorithm, to integrate results from different spatial maps. These algorithms will allow users to synthesize results from across the literature to form customized graphs of the macro-connections of macaque cortex.

CoCoTools offers users as much control over data processing as they need: those interested in the low-level details are able to supply optional arguments to functions and alter the source code, but the default functionality of the package is designed to free users from these details. A few simple commands produce brain networks from a user-specified set of studies (or all studies) that can be visualized and used to test neuroscientific hypotheses. Many packages in the Python standard library (Jones, Oliphant, & Peterson, 2001; Perez & Granger, 2007) make this seamless transition from CoCoMac to brain graphs possible. Aside from the standard library, CoCoTools depends on the third-party packages NumPy (a mathematics toolbox), matplotlib (for visualization: Hunter, 2007), and NetworkX (Hagberg et al., 2008).

Downloading and Using CoCoTools:

CoCoTools is available for download at github (https://github.com/cocotools/CoCoTools). We provide a README.MD text file for installation instructions. Once installed, all of CoCoTools' code and functionality become available. CoCoTools will remain version-controlled and as changes and improvements are made, they will be made available on the github site. A versioned release (v0.1) of the repository, which corresponds with this manuscript, has been made available for download. Users can opt to link their local copy to the current version on github for easy updating and for contributing code. Instructions for installing git, starting a github account and creating a local linked repository are provided here (main help: https://help.github.com/, installing git: https://help.github.com/articles/set-up-git and linking to a repo: https://help.github.com/articles/fork-a-repo).

CoCoTools Website, Documentation and Tutorials:

We have created a website for CoCoTools (http://cocotools.github.io) that provides detailed documentation, tutorials, downloading and installation instructions as well as other features. Users are encouraged to refer to the website. The documentation (i.e. "doc strings") for the core functions in the processing pipeline can be found online (http://cocotools.github.io/api.html). We also include a full tutorial in Appendix C.

CoCoTools Processing Pipeline: Querying

Querying CoCoMac in CoCoTools requires no familiarity with database syntax; it is performed with a single command, and the results are automatically parsed to construct NetworkX directed graphs. Several features of CoCoTools optimize queries. For instance, query results are cached in a local database, so they can be accessed independently of the CoCoMac server later. Storage in and retrieval from the local database is automatic, so the user need not have any understanding of SQL or XML, the data formats used in this process. If the user does have such understanding, however, CoCoTools' methods can be fully customized. In addition, CoCoTools' query software is resilient against CoCoMac server timeouts, which are frequent. Furthermore, CoCoTools is divided into independent modules, so if future versions of CoCoMac use a different server design, necessary updates to CoCoTools will be limited to the query module. Queries for both connectivity and mapping statements are executed using the **multi_map_ebunch** function. A list of brain maps can be passed to **multi_map_ebunch** as an optional second argument. If no list is given, the entire corpus is queried (see Figure 2 inset A). Because some studies produce a server timeout when all their areas are queried at once, they must be queried separately using the **query_maps_by_area** function. Lists of brain maps provided in CoCoTools enable users to call the appropriate query functions on particular brain maps without knowing in advance which cause server failure.

CoCoTools Processing Pipeline: MapGraph and ConGraph objects:

MapGraph and **ConGraph** are CoCoTools objects that hold query results as directed graphs. **MapGraph** instances hold mapping data and express brain regions as nodes, spatial relationships as directed (but always reciprocal) edges, and RCs as edge attributes. **ConGraph** instances hold connectivity data and express brain regions as nodes, anatomical connections as directed edges, and ECs for the source and target are edge attributes (see Figure 2 inset B).

CocoTools Pipeline: Pre-Processing: Cleaning the MapGraph:

While developing CoCoTools we discovered and flagged hundreds of omissions and many errors in CoCoMac mapping data. The **clean_data** method of MapGraph searches the input mapgraph for these problem edges or nodes, rectifies these problems, and returns a cleaned mapgraph (see Figure 2 inset C1). For example, this method will add many new mapping relationships to PHT00, RM, CP99 and several other maps (approximately 490 new edges in total) if these maps are present in the input mapgraph. Similarly, **clean_data** searches for and then removes approximately 20 edges and 18 nodes that we have identified as incorrect. Many of the fixes implemented by **clean_data** ensure that subsequent steps, especially registration run properly.

CocoTools Pipeline: Pre-Processing: Remove Hierarchies

ORT and mORT tacitly assume that all regions in the same brain map must not overlap and that each map's areas exhaust the cortical space within the map's borders: there can be no gaps without names between areas. Depending on which assumptions are violated, CoCoTools' registration will either crash or potentially produces errors. Unfortunately, CoCoMac data violate all of these assumptions. To resolve this dilemma, the CoCoTools command, **keep_only_one_level_of_resolution** builds a hierarchy for each map based on its spatial relationships, and removes from each map all but one level of resolution – ensuring that the areas kept in the graph are disjoint (see Figure 2 inset C2). By default, this command keeps the level with the most anatomical connections, and then translates connections from the other levels to the one that is retained.

CocoTools Pipeline: Pre-Processing: The Deduction of Formerly Unknown Relations:

Only a small number of spatial relationships have been stated in the literature and stored in CoCoMac, but these can be used to infer unstated relationships (Stephan et al., 2000). In CoCoTools, the **deduce_edges** command applies a classic technique from graph theory – Floyd's algorithm (Floyd, 1962) to infer chains of relationships among brain areas (see Figure 2 inset C3). Our default procedure is most similar to the method of Kötter & Wanke, 2005, with a few minor differences (see Appendix D for details). Aside from those that depend on the CoCoMac server, this is the one command within CoCoTools that does not complete within a few seconds on modern desktop computers: Its use of Floyd's algorithm requires approximately N^3 computations, with N being the number of nodes. That this method takes several hours to complete with the full CoCoMac dataset is outweighed by the huge number of unstated mapping relationships it deduces, numbering in the hundreds of thousands. Without these deduced relationships, many translations would be impossible.

CocoTools Pipeline: Coordinate-Free Registration: Initialize EndGraph

The first step in the registration process is to initialize an **EndGraph** object. **EndGraph** objects contain the machinery to perform coordinate-free registration and, being graphs, hold the results as nodes representing regions in the target space and directed edges representing anatomical connections (see Figure 2 inset D).

CocoTools Pipeline: Coordinate-Free Registration: Applying ORT or mORT

CoCoTools implements two coordinate-free registration methods: ORT and mORT. Both methods are executed using the high-level command **add_translated_edges**, This command takes as input, mapping data (mapgraph), connectivity data (congraph), the CoCoMac name for the output space and either 'original' (for ORT) and 'modified' (for mORT) and returns an endgraph in the given output space that is populated with the edges from congraph that have translated via ORT or mORT into the output space (see Figure 2 inset E). In ORT, each anatomical connection in the input congraph is decomposed into its source area and its target area. ORT then translates the extent of tracer present in the source area of the input space to the corresponding source area or areas in the output space, and similarly translates tracer present in the target area of the input space to the corresponding target area or areas in the output space. As mentioned above, ORT uses the RCs in the mapgraph between the areas in the input and output space to derive the necessary mapping relationships between input and output spaces and the EC values for edges in the congraph to provide stain extent. In mORT, a connection code is derived from the ECs in the input congraph and only three labels are used to classify stain: Present, Absent, and Unknown. Connections in mORT are then translated as a whole.

CocoTools Pipeline: Post-Processing: remove 'Absent' and' Unknown' edges:

The process of coordinate-free registration will result in some edges in the output space having the attribute of "Absent" or "Unknown". In most cases, users will want to remove these edges. The **strip_absent_and_unknown_edges** function searches the input endgraph for edges with the attribute of 'Absent' or 'Unknown', removes these edges and returns a digraph (see Figure 2 inset F).

CoCoTools Visualization

CoCoTools has several visualization options that grant users control over the spatial layout (e.g., coronal, axial, or sagittal), color, and transparency of nodes (which represent brain areas). Plotting utilities draw upon and extend the third-party Python libraries NetworkX and matplotlib (Hunter, 2007). To visualize translated brain graphs, CoCoTools users need only supply the function **graph_view** with a CSV file containing node names, coordinates, and colors (in RGB format). Coordinates for the PHT00 (Paxinos et al., 2000), FV91 (Felleman & Van Essen, 1991), and RM brain maps (Kötter & Wanke, 2005) are included with our package (see below for descriptions of these maps). Coordinates and color schemes for these maps were supplied courtesy of the region centers service of the scalable brain atlas (http://scalablebrainatlas.incf.org/services/regioncenters.php), which is a part of the ongoing work of the CoCoMac 2.0 team. The colors schemes supplied in these files are for aesthetic purposes.

Registration Target Maps Selected for CoCoTools Use Case Demonstration:

We selected several prominent parcellation schemes to serve as the target space for ORT and mORT registrations in our examples. The first is PHT00, the whole-brain stereotaxic atlas by Paxinos et al. (Paxinos et al., 2000), which makes finer distinctions between areas than most prior studies. Thus, most connectivity statements from other brain maps in CoCoMac refer to regions that are larger than PHT00's regions, which limit the amount of connectivity statements that can be successfully translated to PHT00.

The other parcellation schemes we chose are FV91, RM, B05, and BB47. FV91, from Felleman and Van Essen (Felleman & Van Essen, 1991), was designed as part of a pioneering manual effort to construct a connectome of macaque visual areas. For this reason, FV91 contains a wealth of native connections. In contrast, most other connections in CoCoMac occur between two different parcellation schemes (e.g. Area 46 of Petrides and Pandya, 1994 connects to Area 31 in Morecraft, Cipolloni, Stilwell-Morecraft, Gedney, & Pandya,et al., 2004). The RM map comes from Kötter and Wanke (2005)[2]. It is based on gross anatomical features like gyri and sulci rather than cytoarchitecture and was designed for cross-study and cross-species comparison. B05 (Brodmann, 1905) is fairly low-resolution, but widely cited, and therefore has an abundance of mapping statements in CoCoMac that can be used for translation. Finally, BB47 (von Bonin & Bailey, 1947) is similar to B05 in its resolution and availability of mapping statements.

Small-Worldness Calculations

We examined the small-world properties of the ORT and mORT endgraphs as one means of assessing the face validity of CoCoTools' the selected use cases. Small-worldness compares a graph's clustering coefficient (γ: the density of connections between neighboring nodes) and its characteristic path length (λ: the average geodesic distance between pairs of nodes in a graph) to a random graph matched in number of nodes, number of edges, and degree distribution. A graph is said to be "small world" if its clustering coefficient (γ_{measured}) is much greater than the clustering coefficient of a matched random graph (γ_{random}), yet its characteristic path length ($\lambda_{\text{measured}}$) is similar to that of a matched random graph (λ_{random}) (Watts &

[2] This map was erroneously called "R00" in the CoComac database due to a collation error.

Strogatz, 1998). We derived a metric from the original Watts and Strogatz (1998) description that labels a graph as small world if:

$$\frac{\gamma_{measured} - \gamma_{random}}{\gamma_{random}} > \frac{\lambda_{measured} - \lambda_{random}}{\lambda_{random}}$$

.

Alternative measures of small-worldness exist, and we have calculated the most prominent for our graphs. One is from Humphries et al. (2006). This criterion scales the observed graph's clustering coefficient and characteristic path length by those of a matched random graph. The observed graph is small world if

$$\frac{\gamma_{measured}/\gamma_{random}}{\lambda_{measured}/\lambda_{random}} > 1.$$

Sporns and Zwi (2004) propose another alternative. Their criterion compares the observed graph against a matched lattice graph as well as a random graph. Lattice graphs are characterized by high clustering and path length. For this metric, scaled clustering and characteristic path length are calculated separately, and the graph is considered small-world if:

$$\frac{\gamma_{measured} - \gamma_{random}}{\gamma_{lattice}/\gamma_{random}} < 1$$

$$\frac{\lambda_{measured} - \lambda_{random}}{\lambda_{lattice}/\lambda_{random}} < 1$$

RESULTS

Usability and timing of commands:

Table 1A-D shows the timing data for each of the core CoCoTools commands with the entire CoCoMac database as input. This testing was performed on an Intel Core i5 CPU Laptop with 1.33 GHz processor and 4GB of RAM operating a 64-bit Windows 7 Operating System. As was discussed in the Methods section, all functions run quickly except for the *deducing new mapping relationships* process which takes 1044 minutes. Table 1E shows CoCoTools' test coverage. This is expressed, for each python file in our main package, as the percentage of lines of code covered by tests. This table shows very high coverage for most of the package.

Effect of Output Space on EndGraph properties:

In separate analyses, we registered the full CoCoMac corpus to five target spaces using mORT and ORT (the matrices are available as ".csv" files from the CoCoTools website (http://cocotools.github.io/downloads.html#data-download). We chose PHT00, RM, FV91, B05, and BB47, because these mapping spaces are representative of the variability within CoCoMac (Table 2A). As shown in Table 2B, Coordinate-free registration adds edges to these target spaces. The increase in edges is especially striking for PHT00, as before registration this dataset had only 49 connectivity statements and after ORT or mORT the endgraphs have over 650 connections.

In terms of the number of edges present relative to the number of edges possible given the number of nodes, the B05 and BB47 endgraphs are very dense (Table 2C). This supports the contention that many translatable edges exist for these relatively old (and therefore thoroughly studied), coarse, and small maps. In contrast, the PHT00 endgraphs, which used the very same input mapping and connectivity data, are relatively sparse, with approximately 3.6% density. This low density is explained by PHT00's recent introduction and fine delineation of areas: the newer a map is, the fewer studies there are that have studied its connections, and the finer a map is, the harder it is to infer connections, as input data are comparably coarse (see Methods: *Registration Target Maps Selected for CoCoTools Demonstrations*). When removing the nodes without edges, this value increases to 7%. The FV91 target space yielded an intermediate density. Table 2 further illustrates that only a few nodes in all the EndGraphs lack translated connections (Table 2D); thus, CoCoMac has adequate data in most cases to allow ORT and mORT to translate information.

Effects of Registration Method on EndGraph properties:

Table 2 indicates the ORT and mORT produced highly similar EndGraphs for all the output spaces tested. This is reflected in a very similar number of overall edges (Table 2B compare the two middle columns) as well as a very high number of edges in common (Table 2B right-most column). The adjacency matrices (Figure 3) show this as well.

Visualization of mORT EndGraphs:

Figure 4 visualizes the mORT-registered endgraphs for PHT00 and RM, and in Figure 5, subgraphs within these are highlighted. These plots were created using the CoCoTools **graph_view** function (see Methods, Appendix C for description).

Small-Worldness of EndGraphs:

"Small-worldness" is a ubiquitous property of biological systems and has been documented within the human fMRI connectome (see Bassett & Bullmore, 2006 Bullmore & Sporns, 2009 for review) as well as in a single-study macaque anatomical dataset (Sporns & Zwi, 2004). Therefore, testing for small-worldness is one reasonable initial means for assessing whether our results are consistent with what is known about the brain's global organization. That is, if our methods are valid we would expect that given reasonable inputs, a large CoCoTools brain graph should demonstrate small-world properties. Small-worldness is qualitatively defined as having path length similar to a random graph of equal size, but clustering much greater than the same random graph. Because there is debate about how best to calculate small-worldness, we used three different criteria (Humphries, Gurney, & Prescott, 2006; Sporns & Zwi, 2004; Watts & Strogatz, 1998). The results are presented in Table 3, which indicates "small-worldness" by all three metrics for the PHT00, RM, FV91, and B05 endgraphs. The BB47 endgraphs met the Humphries et al. (2006) criteria and our metric based on the Watts and Strogatz (1998) description, but failed to meet the criteria proposed by Sporns and Zwi (2004). The BB47 endgraph yielded a scaled characteristic path length of 1.51, which is above the Sporns and Zwi (2004) maximum criterion of 1.00. However though, BB47 is a relatively small graph and the Sporns and Zwi (2004) heuristic was used and developed on larger graphs.

Demonstration of ORT - mORT Differences:

In developing CoCoTools, we realized that ORT has the potential to be overly liberal in assigning edges as present (i.e. EC='C', 'P' or 'X') or absent (i.e. EC= 'N') (see Appendix A). We developed mORT as a more conservative approach which, we posited, should not make these mistranslations. Our results from Figure 3 and Table 2 do indicate a high degree of similarity between the outputs of ORT and mORT for the spaces we selected. However there were differences, and here we explore one such case for which ORT is overly liberal in assigning a present connection to an edge. By exploring this case, we hope to illustrate the conceptual differences between ORT and mORT (See Figure 6; see also: ORT_walkthough.ipynb on https://github.com/cocotools/CoCoTools/tree/master/examples). In this Figure, we can see that for the connection from BB47-OB to BB47-FD, given all the data in CoCoMac, ORT produced a present connection, with P,X as the ECs for source and target respectively, mORT however produced an 'Unknown' connection. Figure 6 inset A shows the input data (as a table and a cartoon) for which ORT and mORT produced different results. In working through this data, one must first consider the ECs and RCs for each B88 connection (illustrated in cartoon Figure 6 inset B). For example, note that there is only 1 connection with valid stain in both the source and the target (i.e. the connection from B88-V2 to B88-32), however the stain in the target region (B88-32) has only an extent of 'X' and it is only overlapping the target region in the output space (BB47-FD). Logically and according to ORT's connection calculus, this one connection should lead to an "unknown" connection (with 'P', 'U", as the source and target ECs). However, given this data, and the other inputs, ORT mistakenly produces a present connection (with ‘P’, ‘X’ as the source and target ECs). We can see in Figure 6 inset C how this happens. As mentioned previously, ORT does not translate “connections” but rather stain, and each occurrence of stain (each EC) in both the source and the target are considered independently (demonstrated by the cartoon on the right side of Figure 6C). In ORT, after all the relevant ECs are gathered for each region, the most extensive EC is selected and translated into the output space. In this case, ORT successfully translates the “P” EC in B88-V2 to a “P” in BB47-OB and the “X” EC in BB47-FD to an “X” in BB47-FD. Notice how individually, these two translations of stain are logically correct, but the conclusion that ORT draws; that a connection is present, is incorrect. That is, ORT has successfully translated the most extensive ECs into BB47 space, but failed to accurately translate the connections. MORT, on the other hand, translates connections without relying on the extent of stain and gleans from the input data only whether there is a connection between the regions and not the ECs (Figure 6 insets B and D). It then integrates this information with the RCs. According to the simpler connection calculus of mORT, no ‘Present’ input connection will be translated as ‘Present’ in the output space if either the source or target is overlapping to the corresponding source or target in the output space (Figure 6 inset E). Such is the case with BB47-FD and B88-32, and thus mORT will correctly translate this connection as ‘Unknown’.

DISCUSSION

With the emergence of graph-theoretical methods and the growing shift from single-region to network-level analyses of brain imaging data, there is a great demand for reliable and large-scale descriptions of primate neuroanatomy. The CoCoMac database and the ORT coordinate-free registration algorithm lay the groundwork for providing such descriptions to the neurosciences. However, access to the CoCoMac database is limited, and an implementation of the ORT algorithm is not available to the general public. Furthermore, as we have described in the Methods, ORT, on its own, is incapable of properly handling all the data stored in CoCoMac. In this manuscript, we have illustrated how the CoCoTools Python library solves these limitations and provides easy-to-use tools for the creation of macaque macro-connectomes. Results acquired with CoCoTools will aid qualitative reviews of the neuroanatomical literature as well as quantitative graph-theory analyses.

CoCoTools is implemented as an open source Python library with a high-level interface that constructs the necessary low-level operations for the user without the need for manual oversight of every step. The Python language was a sound implementation choice as it is freely available, highly portable, and rapidly growing as a general purpose tool in multiple areas in neuroscience. The standard Python distribution provides built-in tools for querying web servers, parsing outputs, and creating local databases. The NetworkX library, developed at the Theory Division of Los Alamos National Laboratory (Hagberg et al., 2008), provides the necessary computational and graph-theory tools required to work with CoCoMac data structures and to implement the ORT algorithm (and our modified version of ORT). The combination of the general-purpose nature of the Python language with a wealth of specialized packages for scientific computing makes it an excellent choice for problems such as this, that require a mix of numerical/mathematical approaches and other ’classic’ computing tasks involving network protocols, database interaction, and XML processing.

CoCoTools implements two coordinate-free registration algorithms. Our work borrows heavily from Klaas Stephan and Rolf Kötter, who were the first to realize that neuroanatomical data can be divided into mapping and connectivity information, and that these data can be translated in a coordinate-free way. Given that the vast majority of studies in the anatomical literature cannot be mapped to a coordinate-space, this insight paved the way for the development of the ORT algorithm, which uses logical qualitative relationships to register data from different studies together.

Nevertheless, the challenges that ORT and any other coordinate-free registration algorithm faces are significant. For instance, a viable translation method must produce an unambiguous result from any combination of connection and mapping input. Translation must be robust when not every connection in input maps has been tested and not every mapping space covers the whole brain. Although the developers of ORT alluded to methods for handling many of these issues, the only versions of ORT they specified in full cannot perform adequately when faced with the full CoCoMac dataset. For instance, the most recent and widely cited description of ORT (Stephan et al., 2000) fails to consider the complete relationship between the original space and the target space (see the Methods). Even more problematic is that fact that all versions of ORT fail when input from the same mapping space contains overlapping regions or regions at different hierarchical levels. Aside from problems inherent to the ORT algorithm, CoCoMac data are rife with errors and omissions.

In our Methods and Results we guide the user through solutions that we have developed for these problems. For example, CoCoTools implements the Stephan and Kötter (1998) version of ORT, which reasoning (see Methods: *Overview of Coordinate-free Registration, ORT and mORT*) and testing in Python imply is the most robust. Furthermore, we added several pre-processing routines that correct CoCoMac errors and omissions, remove intra-map spatial hierarchies, and deduce mapping statements absent from the literature. Overall in CoCoTools, we implemented different features from the different original sources and even made subtle modifications to the original processing. Thus the CoCoTools ORT implementation differs significantly from any one of the original descriptions. However it remains true to the core logic of ORT and to our knowledge no other implementation of ORT is openly available.

Beyond the additions that do not fundamentally change the logic of ORT, we include a modified version of ORT (mORT) that uses simpler logic, is more conservative and does not make any assumptions about the extent of stain. We show in the Results, that mORT does successfully handle certain cases where ORT mistranslates. However, our results from Table 2 and Figure 3 indicate only small differences overall between ORT and mORT endgraphs. Within each of the five target spaces, ORT and mORT endgraphs had a majority of edges in common and evinced similar densities. This evidence that logically different approaches to the same translation problem infer similar cortex-wide connectivity bolsters the plausibility of our final matrices and points to overall consistency of our method. Although, based on these results, it is reasonable to expect only small differences between mORT and ORT in other endgraphs, given the complexity of the problems surrounding coordinate-free registration it is nonetheless useful to have multiple different approaches available in CoCoTools.

The data in Table 2 reveal that in all target spaces, ORT and mORT added many edges. Because the target spaces we chose are representative of the diversity of maps in CoCoMac, these results are a good indicator of how much information CoCoTools can provide compared to a manual literature search or CoCoMac query. For instance, in CoCoMac, the B05 brain map is initially associated with 85 connectivity statements, only two of which are native, and of its 28 nodes, only 10 are associated with connections. If one manually searched the macaque literature using just B05 as a reference, one would uncover just this supply of connections. In contrast, using CoCoTools and mORT yields an EndGraph with 216 edges and 24 nodes with edges. This is marked improvement over the manual method, and results are even better for other target spaces. As shown in Figure 5 for the PHT00 endgraph, selected posterior/visual and prefrontal edges have connections that resemble known anatomy: visual/posterior regions show a dorsal-ventral segregation of edges, and prefrontal nodes are disproportionately connected within the frontal lobe and to high-level sensorimotor regions rather than primary sensory regions. In addition, by most criteria, the endgraphs exhibit small world properties in the target-spaces we tested. Future analyses that integrate just data known to be of high quality, rather than the full CoCoMac corpus, are likely to have even greater success than the initial demonstrations we present here.

The connectivity data provided by CoCoMac and CoCoTools will be important for cognitive neuroscience research as there is a growing realization that the functions of the brain do not operate in isolation but rather need to be understood and studied in the context of a network (Sporns, Chialvo, Kaiser, & Hilgetag, 2004). Moreover, with the introduction of graph-theory methods, neuroscientists are now able to use quantitative metrics with this data to understand the network properties of the brain with increasing precision and scope. However it is important to appreciate the empirical limitations of CoCoMac and its applicability to graph-theory metrics. For instance, as mentioned above in our methods and as discussed by others (Modha & Singh, 2010; Stephan et al., 2001), CoCoMac does contain errors and omissions that are not completely resolved in CoCoTools. Additionally, important fields, such as injection site, promised in the original descriptions of CoCoMac (Stephan et al., 2001) are difficult to access online. Thus, as a caveat, it is important for users to examine the original sources to ensure quality control. We are actively examining the original data, tracking errors and added fixes in CoCoTools (i.e. see clean_data) and we encourage others to contribute to this curation. Currently, all the errors we have encountered we have treated via the **clean_data** method. Beyond data quality, it is important to consider that graph-theory metrics are subject to their own set of constraints (e.g. graph size, density) and therefore all endgraphs will not be appropriate for all graph-theory metrics. It is important to evaluate the constraints of the data and of the methods when working with CoCoTools. In summary though, although CoCoMac and coordinate-free methods have specific and complex limitations, these methods remain useful even as new methods and new data become available.

Finally, we note that at the time of this writing, an improved version 2.0 of the CoCoMac server is under development and can already be accessed in preliminary form at http://cocomac.g-node.org. We expect some of the limitations of the version 1.0 server to be addressed by this new system, but for end users of our tool, there will be an easy update path, limited just to the CoCoTools query module. We thus expect our CoCoTools library to continue to be a useful complement to the work done by the CoCoMac team, providing users with flexible tools for registering, processing, and analyzing CoCoMac connectomes. Additionally, there are other valuable published approaches that complement CoCoTools. Several excellent studies by Bezgin and colleagues (Bezgin, Reid, Schubert, & Kötter, 2009; Bezgin, Vakorin, van Opstal, McIntosh, & Bakker, 2012; Bezgin, Wanke, Krumnack, & Kötter, 2008) offer sophisticated methods for combining CoCoMac data with spatial information gleaned from the F99 macaque atlas, which is freely available with the Caret software (D. C Van Essen et al., 2001). One nice feature of this functionality is that it can be used to register macaque data from CoCoMac onto the human cortical surface using the RM map in the macaque and the PALs atlas in the human (David C Van Essen, 2005). A third resource (Modha & Singh, 2010) is a freely available brain graph derived from CoCoMac using a simplified algorithm that makes no distinctions between areas with the same name from different maps. These approaches, and ours, have non-overlapping strengths and weaknesses, so which tool is best will depend on the scientific questions at hand. Because CoCoTools is being released as an open-source package, the opportunity exists for users to identify and correct its weaknesses and expand the range of research objectives it can support. Community-driven development is the only means by which a final, complete monkey connectome will be achieved.

REFERENCES

Bakker, R., Wachtler, T., & Diesmann, M. (2012). CoCoMac 2.0 and the future of tract-tracing databases. *Frontiers in Neuroinformatics*, *6*. doi:10.3389/fninf.2012.00030

Bassett, D. S., & Bullmore, E. (2006). Small-world brain networks. *The Neuroscientist: a review journal bringing neurobiology, neurology and psychiatry*, *12*(6), 512–523. doi:10.1177/1073858406293182

Bezgin, G., Reid, A. T., Schubert, D., & Kötter, R. (2009). Matching spatial with ontological brain regions using Java tools for visualization, database access, and integrated data analysis. *Neuroinformatics*, *7*(1), 7–22. doi:10.1007/s12021-008-9039-5

Bezgin, G., Vakorin, V. A., van Opstal, A. J., McIntosh, A. R., & Bakker, R. (2012). Hundreds of brain maps in one atlas: Registering coordinate-independent primate neuro-anatomical data to a standard brain. *NeuroImage*, *62*(1), 67–76. doi:10.1016/j.neuroimage.2012.04.013

Bezgin, G., Wanke, E., Krumnack, A., & Kötter, R. (2008). Deducing logical relationships between spatially registered cortical parcellations under conditions of uncertainty. *Neural networks: the official journal of the International Neural Network Society*, *21*(8), 1132–1145. doi:10.1016/j.neunet.2008.05.010

Brodmann, K. (1905). Beitraege zur histologischen Lokalisation der Grosshirnrinde. III. Mitteilung. Die Rindenfelder der niederen Affen. *Journal für Psychologie und Neurologie*, *4*, 177–226.

Brodmann, K. (1909). *Vergleichende Lokalisationslehre der Grosshirnrinde*.

Bullmore, E., & Sporns, O. (2009). Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature reviews. Neuroscience*, *10*(3), 186–198. doi:10.1038/nrn2575

Cohn, A. G., Bennett, B., Gooday, J., & Gotts, N. M. (1997). Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. In *PROCEEDINGS OF THE DIMACS INTERNATIONAL WORKSHOP ON GRAPH DRAWING, 1994. LECTURE NOTES IN COMPUTER SCIENCE* (pp. 89–4).

Felleman, D. J., & Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex (New York, N.Y.: 1991)*, *1*(1), 1–47.

Floyd, R. W. (1962). Algorithm 97: Shortest path. *Commun. ACM*, *5*(6), 345–. doi:10.1145/367766.368168

Ghosh, A., Rho, Y., McIntosh, A. R., Kötter, R., & Jirsa, V. K. (2008). Noise during rest enables the exploration of the brain’s dynamic repertoire. *PLoS computational biology*, *4*(10), e1000196. doi:10.1371/journal.pcbi.1000196

Hagberg, A. A. N. L., Swart, P. A. N. L., & Schult, D. U. (2008). *Exploring network structure, dynamics, and function using networkx*. Retrieved from http://www.osti.gov/energycitations/servlets/purl/960616-jwnOFP/

Human Connectome Project. (2012, February 15). Retrieved February 15, 2012, from http://www.humanconnectomeproject.org/

Humphries, M. D., Gurney, K., & Prescott, T. J. (2006). The brainstem reticular formation is a small-world, not scale-free, network. *Proceedings. Biological sciences / The Royal Society*, *273*(1585), 503–511. doi:10.1098/rspb.2005.3354

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, *9*(3), 90–95. doi:10.1109/MCSE.2007.55

International Neuroinfomatics Coordinating Facility. (2012, February 15). Retrieved February 15, 2012, from http://www.incf.org/

Jones, E., Oliphant, T., & Peterson, P. (2001). *SciPy: Open Source Scientific Tools for Python*. Retrieved from http://www.scipy.org

Knock, S. A., McIntosh, A. R., Sporns, O., Kötter, R., Hagmann, P., & Jirsa, V. K. (2009). The effects of physiologically plausible connectivity structure on local and global dynamics in large scale brain models. *Journal of Neuroscience Methods*, *183*(1), 86–94. doi:10.1016/j.jneumeth.2009.07.007

Köbbert, C., Apps, R., Bechmann, I., Lanciego, J. L., Mey, J., & Thanos, S. (2000). Current concepts in neuroanatomical tracing. *Progress in Neurobiology*, *62*(4), 327–351. doi:10.1016/S0301-0082(00)00019-8

Kötter, R. (2004). Online retrieval, processing, and visualization of primate connectivity data from the CoCoMac database. *Neuroinformatics*, *2*(2), 127–144. doi:10.1385/NI:2:2:127

Kötter, R., & Wanke, E. (2005). Mapping brains without coordinates. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, *360*(1456), 751–766. doi:10.1098/rstb.2005.1625

Kuypers, H. G., & Ugolini, G. (1990). Viruses as transneuronal tracers. *Trends in neurosciences*, *13*(2), 71–

75.

Modha, D. S., & Singh, R. (2010). Network architecture of the long-distance pathways in the macaque brain. *Proceedings of the National Academy of Sciences of the United States of America*, *107*(30), 13485–13490. doi:10.1073/pnas.1008054107

Morecraft, R. J., Cipolloni, P. B., Stilwell-Morecraft, K. S., Gedney, M. T., & Pandya, D. N. (2004). Cytoarchitecture and cortical connections of the posterior cingulate and adjacent somatosensory fields in the rhesus monkey. *The Journal of comparative neurology*, *469*(1), 37–69. doi:10.1002/cne.10980

Paxinos, G., Huang, X.-F., & Toga, A. W. (2000). *The rhesus monkey brain□: in stereotaxic coordinates*. London: Elsevier/Academic Press.

Perez, F., & Granger, B. E. (2007). IPython: A System for Interactive Scientific Computing. *Computing in Science Engineering*, *9*(3), 21 –29. doi:10.1109/MCSE.2007.53

Petrides, M., & Pandya, D. N. (1994). Comparitive architectonic analysis of the human and macaque frontal cortex. In *Handbook of Neuropsychology*. Amsterdam: Elsevier.

Sporns, O., Chialvo, D. R., Kaiser, M., & Hilgetag, C. C. (2004). Organization, development and function of complex brain networks. *Trends in cognitive sciences*, *8*(9), 418–425. doi:10.1016/j.tics.2004.07.008

Sporns, O., & Zwi, J. D. (2004). The small world of the cerebral cortex. *Neuroinformatics*, *2*(2), 145–162. doi:10.1385/NI:2:2:145

Stephan, K. E., Kamper, L., Bozkurt, A., Burns, G. A., Young, M. P., & Kötter, R. (2001). Advanced database methodology for the Collation of Connectivity data on the Macaque brain (CoCoMac). *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, *356*(1412), 1159–1186. doi:10.1098/rstb.2001.0908

Stephan, K. E., & Kötter, R. (1998). A formal approach to the translation of cortical maps. In *Neural Circuits and Networks* (pp. 205–226). Berlin: Springer.

Stephan, K. E., & Kötter, R. (1999). One cortex – many maps: An introduction to coordinate-independent mapping by Objective Relational Transformation (ORT). *Neurocomputing*, *26–27*(0), 1049–1054. doi:10.1016/S0925-2312(99)00103-4

Stephan, K. E., Zilles, K., & Kötter, R. (2000). Coordinate-independent mapping of structural and functional data by objective relational transformation (ORT). *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, *355*(1393), 37–54. doi:10.1098/rstb.2000.0548

Van Essen, D. C, Drury, H. A., Dickson, J., Harwell, J., Hanlon, D., & Anderson, C. H. (2001). An integrated software suite for surface-based analyses of cerebral cortex. *J Am Med Inform Assoc*, *8*(5), 443–59.

Van Essen, David C. (2005). A Population-Average, Landmark- and Surface-based (PALS) atlas of human cerebral cortex. *NeuroImage*, *28*(3), 635–662. doi:10.1016/j.neuroimage.2005.06.058

Von Bonin, B., & Bailey, P. (1947). *The Neocortex of Macaca Mulata*. Urbana, Illinois: University of Illinois Press.

Walker, A. E. (1940). A cytoarchitectural study of the prefrontal area of the macaque monkey. *The Journal of Comparative Neurology*, *73*(1), 59–86. doi:10.1002/cne.900730106

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of “small-world” networks. *Nature*, *393*(6684), 440–442. doi:10.1038/30918

Young, M. P. (1993). The Organization of Neural Systems in the Primate Cerebral Cortex. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, *252*(1333), 13–18. doi:10.1098/rspb.1993.0040

Zilles, K., & Amunts, K. (2010). Centenary of Brodmann’s map--conception and fate. *Nature Reviews. Neuroscience*, *11*(2), 139–145. doi:10.1038/nrn2776

Table 1: Timing of CoCoTools functions and Test Coverage

A) Query

process	command	time
mapping query	multi_map_ebunch('Mapping', coco.MAPPING_NON_TIMEOUTS)	18.4 s
connectivity query	multi_map_ebunch('Connectivity', coco.MAPPING_NON_TIMEOUTS)	33.9 s
mapping query region by region	query_maps_by_area('Mapping', coco.MAPPING_NON_TIMEOUTS)	12.3 s
connectivity query region by region	query_maps_by_area('Connectivity', coco.MAPPING_NON_TIMEOUTS)	15.8 s

B) Pre-Processing

process	command	time
clean data	clean_data	68.1 ms
remove hierarchies	keep_only_one_level_of_resolution	1.45 s
deduce new mapping relations	deduce_edges	1044 min

C) Coordinate-free Registration

process	command	time
ORT	endg.add_translated_edges(mapg, cong, 'PHT00', 'original')	6.85 s
MORT	endg.add_translated_edges(mapg, cong, 'PHT00', 'modified')	6.52 s

D) Post-Processing

process	command	time
remove absent and unknown	strip_absent_and_unknown_edges	31.7 ms

E) Test Coverage

name	coverage
cocotools.brain_maps	100%
cocotools.congraph	92%
cocotools.endgraph	82%
cocotools.iotools	49%
cocotools.mapgraph	80%
cocotools.query	92%
cocotools.query_by_area	90%
cocotools.stats	43%
cocotools.utils	96%

Table 2: Description of CoCoMac data and registration results

A) Description of CoCoMac data

	total nodes	connectivity statements	nodes w/ connect. statements	mapping statements	avg. mapping state./ node
PHT00	135	49	48	536	3.80
RM	43	0	0	762	17.72
FV91	54	430	39	424	7.85
B05	28	85	10	438	15.64
BB47	38	301	38	906	23.84

B) Edges

	native edges before registration	edges after ORT	edges after mORT	edges common to ORT and mORT
PHT00	0	664	654	654
RM	0	521	520	520
FV91	369	449	448	448
B05	2	217	216	216
BB47	3	373	370	370

C) Density

	density before registration	density after ORT	density after mORT
PHT00	0%	3.67%	3.62%
RM	0%	33.40%	33.33%
FV91	12.89%	15.69%	15.65%
B05	0.26%	28.70%	28.57%
BB47	0.21%	26.53%	26.32%

D) Nodes w/o edges

	nodes w/o connectivity statements	nodes w/o edges after ORT	nodes w/o edges after mORT
PHT00	93	39	42
RM	43	3	3
FV91	15	2	2
B05	18	4	4
BB47	0	3	4

Table 3: Small-worldness Results

	PHT00	RM	FV91	B05	BB47
clustering coefficient	0.23	0.56	0.54	0.55	0.58
characteristic path length	2.67	1.64	2.35	1.65	1.79
average random clustering coefficient	0.12	0.50	0.29	0.49	00.63
average random characteristic path length	2.5	1.63	2.09	1.63	1.51
small-world (Watts and Strogatz, 1998*)	yes	yes	yes	yes	yes
small-world ratio (Humphries et al. 2006)	1.65	1.11	1.68	1.11	1.39
scaled clustering (Sporns and Zwi, 2004)	0.4	0.48	0.68	0.64	0.62
scaled c. path length (Sporns and Zwi, 2004)	0.34	0.80	0.97	0.87	1.51

*our metric based on the Watts and Strogatz (1998) description

APPENDIX A: Formal description of ORT and mORT

ORT:

Gathering Relevant Data

We begin with a formal description of a projection between two brain regions in brain map “A”

$$Ai_x\ {}^{\mathrm{s}}\mathrm{EC} \rightarrow {}^{\mathrm{r}}\mathrm{EC}Aj_y$$

where Ai_x is the source area, Aj_y is the receiving area, $_{\mathrm{s}}$EC is the EC forAi_x, and $_{\mathrm{r}}$EC is the EC for Aj_y. The ECs describe the extent of tracer observed in the sending and receiving area, respectively.

In this example, the aim of ORT is to express a specific anatomical connection $Ai_0\ {}^{\mathrm{s}}\mathrm{EC} \rightarrow {}^{\mathrm{r}}\mathrm{EC}Aj_0$ in the nomenclature of map B. To accomplish this we first need to determine the mapping between the regions in map A and map B that are relevant to the connection $Ai_0\ {}^{\mathrm{sc}} \rightarrow {}^{\mathrm{rc}}Aj_0$. We begin by gathering all areas in map B that together fully cover Ai_0 and Aj_0. Following the logic of Kötter and Stephan (1998), this can be formalized in set theory notation as

$$saB = \{B_p \in B \mid 1 \leq p \leq b, B_p \cap Ai_0 \neq \emptyset\}$$

$$taB = \{B_q \in B \mid 1 \leq q \leq b, B_q \cap Aj_0 \neq \emptyset\}$$

That is, saB reflects all the regions in Map B that fully cover Ai_0 and taB reflects all the regions in Map B that cover Aj_0.

The Cartesian product $saB\ x\ taB$ describes all the potential connections in map B that can result from translation of $Ai_0\ {}^{\mathrm{s}}\mathrm{EC} \rightarrow {}^{\mathrm{r}}\mathrm{EC}Aj_0$. This Cartesian product is called the ARC by Stephan and Kotter (1998) for “all resulting connections.”

To get the full correspondence between Map A and Map B for both the source and target regions, we now need to project each element in the ARC back to map A. Up to this point, we have gathered only regions in map B that cover Ai_0 and Aj_0. These map B regions may cover other regions of map A, and so we must take into account these other map A regions to express the anatomical connections of interest in the terms of map B. The connections between the full set of source areas in Map A and the full set of target areas in Map A is called, in the terminology of Stephan and Kötter, $saA\ x\ taA$:

$$\{(Ai_x, Aj_y) | Ai_x \in saA, Aj_y \in taA,\ B_p \cap Ai_x \neq \emptyset \wedge B_q \cap Aj_y \neq \emptyset\}$$

The $saA\ x\ taA$ set is computed anew for every anatomical connection that exists between the regions in the ARC.

Note that this step, described originally in Stephan and Kötter (1998), is not mentioned in subsequent ORT descriptions, Rather these subsequent papers consider just the map B regions that cover Ai_0 and Aj_0 (i.e. the ARC) and this will result in inaccurate translations, especially with the full CoCoMac corpus as input. Moreover, when map A connections that were ignored during the translation of the $Ai_0 \rightarrow Aj_0$ connection become input to ORT during later iterations, the resultant projection ECs may conflict with the ECs that generated initially. Aside from being contradictory, both results are likely to be erroneous, as neither was determined based on complete information. There is no mechanism within ORT to reconcile this situation. In CoCoTools, we compute the $saA\ x\ taA$ set and thus our implementation does not suffer from these issues.

ORT's Transformation Logic

Our next aim is to translate each $saA\ x\ taA$ set of connections to Map B. In ORT, for each region in $saA\ x\ taA$, we have a set of sending and receiving extension codes (ECs) – one for each connection – as well as two relation codes (RCs) – one that specifies the logical spatial relationship between the Map A and Map B source regions and one that specifies the logical spatial relationship between the Map A and Map B target regions. For ORT to proceed, the multiple ECs for each source and target area in the context of all these connections must be compressed to a single EC. Following Stephan and Kötter (1998), we keep only the most extensive EC for each region (the one associated with the widest spread of tracer in the area). Each iteration of the ORT transformation logic, described next, takes the EC, RC pair for one region in *saA* or *taA* (the sources are processed separately from the targets) as input.

At each iteration, the transformation logic uses the EC and RC supplied to update its conclusion regarding the status of connections in map B. Stephan and Kötter (1998) define look-up tables using the logic of region connection calculus that specify how the conclusion should be updated, starting from a null state. Note that when the input RC is “I” (that is, Ai_x and B_p are identical) or the RC is “L” (that is, Ai_x is larger than B_p), only one step of translation is required. In cases where the RC is “S” (for smaller) or “O” (for overlapping), the ECs from multiple regions will need to be considered, one at a time, before we can know the final extent of tracer in the Map B region. It is important to note that, in the original descriptions of ORT no transformation rules are provided for situations in which a “U” EC is supplied as input. Given the preponderance of unstudied connections in the literature, in CoCoTools, we have added rules to our implementation that allow it to handle “U” ECs as input.

The simpler transformation logic of mORT

In mORT, the process for gathering the relevant data is the same as in ORT. We gather the $saA\ x\ taA$ as in ORT, but now we define the connection between Ai_0 and Aj_0 as

$$Ai_0 \xrightarrow{conn} Aj_0$$

Where a single value of $\xrightarrow{conn}$ denotes the presence, absence or unknown status of a white matter tract being sent from Ai_0 and received by Aj_0.

Expressed in the terms of set theory used for the original ORT descriptions, the potential values of the connection are as follows:

$$\xrightarrow{conn} \in \{\text{Present, Absent, Unknown}\}$$

These values are deduced from the source and target ECs using the following logic:

Present: $(EC_{source} \in \{C, P, X\}) \wedge (EC_{target} \in \{C, P, X\})$

Absent: $EC_{source} \in \{N\} \wedge EC_{target} \in \{C\}$

OR:

$EC_{source} \in \{C\} \wedge EC_{target} \in \{N\}$

All else Unknown

Now, for each connection in $saA\ x\ taA$, we have a $\frac{conn}{\rightarrow}$ code as well as RCs between the source and target regions in maps A and B, as for ORT. These values serve as inputs to the decision logic of mORT.

We can now define the decision logic that will deliver the final output connection code in map B ($\frac{conn}{\rightarrow}_{final}$).

There are two rules:

mORT Rule 1:

$$\frac{conn}{\rightarrow}_{original} = Present \wedge RC_{source} \in \{I, S\} \wedge RC_{target} \in \{I, S\} \longrightarrow \frac{conn}{\rightarrow}_{final} = Present$$

mORT Rule 2:

$$\frac{conn}{\rightarrow}_{original} = Absent \wedge RC_{source} \in \{I, L\} \wedge RC_{target} \in \{I, L\} \longrightarrow \frac{conn}{\rightarrow}_{final} = Absent$$

For all other conditions, the logic will deliver *Unknown*.

Similar to ORT, the decision logic is applied for each connection in $saA\ x\ taA$, and at each iteration the conclusion regarding the status of connections in Map B is updated.

APPENDIX B: Motivation for CoCoTools mORT:

ORT's use of ECs is intended to allow translated brain maps to carry information about the spatial extent, within areas, of connections. However, this use rests on the assumption that the extent of staining in an area is equivalent to the extent of connectivity. In reality, staining and connectivity are not equivalent in CoCoMac data. This is best illustrated with examples.

Suppose R and S are two brain regions. R is injected such that dye spreads through just a part of it, and stain appears in part of S (whether the dye is retrograde or anterograde makes no difference in this illustration). We know that dye was carried to S from R, but we cannot tell which stained neurons in R effected this transfer. It is possible that every stained neuron in R projects to the stained part of S, but it is also possible that very few stained neurons in R connect to S, and that we would have seen the same amount of stain in S had we injected only these. Thus, all we can say about R with certainty is that some or all of the injected part connects to S.

What about the rest of R, which was not injected? Whether stain would have spread through none, part, or all of S had we injected this complementary part of R cannot be determined. Based on this reasoning, we can update our conclusion: Some or all of R connects to some or all of S. That is, the connection is present; no finer resolution can be achieved. This is the same as saying that in order to reflect the true extent of connection, ECs for projections in CoCoMac like this one should have the value X, meaning "exists with unknown extent." In contrast, CoCoMac would misleadingly assign the connection in this example with source and target ECs equal to P, for "partial."

Now imagine a third region, T. After injection of R as described above, T shows no stain. Because, as just explained for S, we cannot make an unambiguous inference about the projections of the non-injected part of R, we cannot conclude that a connection from R to T is absent. We can say that part of R fails to connect to T, but this is useful only if we know exactly which part, so that we can use the result of an injection in the rest of R to construct a complete picture of R's connectivity. However, this sort of quantitative spatial information is discarded by the EC at the outset. Therefore the connection between R and T remains unknown. Genuine absence of a connection is evinced only when all of one region is injected and no stain appears in the other. ORT does not follow this logic, and therefore is more liberal in labeling connections as absent than the data allow. Depending on the mapping scenario in which these ECs are embedded, ORT's misclassifications can produce inaccurate outputs.

Figure 7 illustrates a situation similar to these examples in which ORT is more liberal in assigning connections as present than the data allow. These errors occur because ORT interprets the CoCoMac EC of "C" in the source region to mean that the entire extent of the source region is connected to the target region (see Figure 10C). However in this example, the source area was completely filled with tracer (see Figure 10B) not because of any connection, but because the experimenter injected a large amount of dye there. It is unclear from the experiment how much of the injected area is genuinely connected to the target region, but in reality (Figure 7A) only part of it is. Thus ORT has misclassified what is really an EC of "P" as "C". When this connection is translated to another space (Mapping Space B in Figure 7) this "C" EC leads ORT to produce spurious connections (see Figure 7D).

The approach we have developed to avoid these logical errors, modified ORT (mORT), involves a simpler syntax and transformation logic with three labels for input connections – Present, Absent, and Unknown – that are applied according to the rationale above. In the example from Figure 7, mORT determines that the connections in the finer scheme are unknown (Figure 7E-F), as the original experiment cannot be used to justify a conclusion. This may seem to imply that mORT will generally label fewer connections in the target map as present compared to ORT, but this is not the case, as ORT oversteps not only in its assignments of connection presence, but also when affirming a connection's absence (see the example with areas R and T above).

With large datasets, whether ORT or mORT is used, input data, drawn from many maps, frequently result in multiple contradictory conclusions for the same connection in the target map. These contradictions must be resolved for a single coherent graph to emerge from the translation. In CoCoTools, we choose to finalize each connection based on the input data reported in the literature with the most precision (though users can modify this behavior). The problem with ORT is its Algebra produces spurious connections even when the input data have high precision. Thus, conflict resolution by this means is not guaranteed to yield an accurate conclusion. Some connections that should be present will be labeled absent, and some that should be absent will be labeled present. mORT solves this problem not by requiring manual oversight or an even more complicated set of rules, but by correcting the logical errors at their source.

There are other advantages to mORT. Whereas ORT translates the source and target areas separately using ECs, mORT translates the connection as a whole in one step. ORT's splitting of connections before translating them makes it prone to errors (see Results: *Demonstration of ORT - mORT Differences* and Figure 6) unless the Algebra is expanded, as Stephan et al. (2000) acknowledge. Furthermore, mORT, unlike ORT, does not require that maps label all areas within their borders, so input data for mORT do not require as much pre-processing.

APPENDIX C: Details of CoCoTools' Deduction of Formerly Unknown Relations:

Only a small number of spatial relationships have been stated in the literature and stored in CoCoMac, but these can be used to infer unstated relationships (Stephan et al., 2000). For example, if an area X has been labeled identical to an area Y, and area Y has been defined as identical to Z, then we can infer X is identical to Z. After making the insight that spatial relationships between areas can be represented as a mathematical graph (as in our **MapGraph**), Stephan et al. (2000) applied a classic technique from graph theory – Floyd's algorithm (Floyd, 1962) – to infer chains of relationships among brain areas. Depending on the relationships in each chain, they can be used to deduce a single RC between the area at the beginning of each chain (X in the example above) and the one at the end (Z). CoCoTools implements this procedure (and updates from Kötter & Wanke, 2005) in the **deduce_edges** command within the **MapGraph**. Chains of relationships can be reduced to single RCs with variable levels of ambiguity, but in CoCoTools we allow new relationships to be added to the graph only in unambiguous cases. When different chains imply a relationship between the same two areas, we keep the RC implied by the chain with the fewest areas. An alternative approach would be to use the path with the lowest cumulative PDC value. However during testing, we frequently found only small cumulative PDC differences between chains. Such differences are not hard to unambiguously interpret and are not discriminative. Nonetheless, this approach can be implemented by the user by modifying the private function '_resolve_contradictions' within mapgraph.py.

APPENDIX D: Demonstrations of CoCoTools Pipeline

We will now provide demonstrations of the CoCoTools processing pipeline using the exact commands users will need. The examples presented here as text and static figures can be downloaded as IPython Notebooks and run in an interactive IPython session. These notebooks can be downloaded here (https://github.com/cocotools/CoCoTools/tree/master/examples), and instructions on how to use the IPython notebook interactively is provided here (http://ipython.org/ipython-doc/dev/interactive/htmlnotebook.html). Alternatively, users can view the static notebooks here (https://github.com/cocotools/CoCoTools/tree/master/examples#a-collection-of-notebooks-for-using-cocotools) using the nbviewer utility on github. Users can additionally refer to Figure 2 for a flowchart describing the CoCoTools processing stream. The example, which we will focus most on in this section, queries and processes data from only a few well-described studies in CoCoMac (CoCoTools_tutorial_small_dataset.ipynb). Appendix E contains the Python commands for a second example that analyzes all of the data stored in CoCoMac. This tutorial is also available on the CoCoTools webpage (CoCoTools_tutorial_entire_corpus.ipynb). Interested readers can use these notebooks as a tutorial or modifiable template script, but we recommend starting with the first example, as it will complete faster and produce a smaller, and therefore more easily interpreted, set of results.

Importing and Querying:

Figure 8 demonstrates the Python and CoCoTools commands necessary to import CoCoTools, NetworkX and query the CoCoMac server. Importing CoCoTools and NetworkX, makes their functions available at the command line (Figure 8 inset A). Figure 3 inset B shows that were using the optional second argument to **multi_map_ebunch** that allows us to specify in a Python list, the names of specific studies we wish to query. If no list is given, the entire corpus is queried. Because some studies produce a server timeout when all their areas are queried at once, they must be queried separately using the **query_maps_by_area** function. Lists of brain maps provided in CoCoTools enable users to call the right query functions on particular brain maps without knowing ahead of time which cause server failure. Figure 8 inset C shows how to initialize

MapGraph and **ConGraph** objects and then how to populate them with our query results.

Pre-Processing:

Before coordinate-free registration can be performed, several pre-processing functions must be run, as described in the Methods. These are demonstrated in Figure 9. First, the **clean_data** command is called within a **MapGraph** that has been populated with mapping data. This function will rectify numerous omissions and errors in CoCoMac (Figure 9 inset A) Next, to reduce each brain map to a single level of resolution as required by ORT and mORT, **keep_only_one_level_of_resolution** is performed (Figure 9 inset B). This function infers the spatial hierarchy in each map and removes non-disjoint areas, translating their information to the level that is kept. The final pre-processing step is to deduce new mapping statements. This is done using the **deduce_edges** method within the **MapGraph** (Figure 9 inset C). Aside from those that depend on the CoCoMac server, this is the one command within CoCoTools that does not complete within a few seconds on modern desktop computers: Its use of Floyd's algorithm (see the Methods and Appendix C) requires approximately N^3 computations, with N being the number of nodes. That this method takes several hours to complete with the full CoCoMac dataset is outweighed by the huge number of unstated mapping relationships it deduces, numbering in the hundreds of thousands. Without these deduced relationships, many translations would be impossible.

Coordinate-Free Registration:

Figure 10 demonstrates the process of coordinate-free registration. Whether the ORT or mORT method is to be used, the first step of translation is to initialize an **EndGraph** object (Figure 10 inset A). As mentioned in the methods, **EndGraphs** contain the machinery to perform coordinate-free registration and, being graphs, hold the results as nodes representing regions in the target space and directed edges representing anatomical connections. The only command within the **EndGraph** users need to be familiar with is **add_translated_edges**, which takes four arguments: a populated **MapGraph**, the corresponding **ConGraph** (with anatomical connections for areas in the **MapGraph**), the acronym of the desired target space, and the desired registration method ('original' for ORT and 'modified' for mORT). This command is executed in

Figure 10 inset B.

Here we use PHT00 as the target space. In Figure 10 inset C, we examine the number of connections that are present for PHT00-9/46D within the congraph that we have been working with. The command **small_cong.edges** returns an empty list, which means that our small query (above) and all subsequent pre-processing have not resulted in any connections originating or terminating within PHT00-9/46D. In Figure 10 inset D, we examine the connections for PHT00-9/46D in ‘small_endg’ that was our output for translation. We can see that the command **small_endg.edges** returns 16 edges. This means that ORT translated 16 edges that were not stated directly in the literature. Each one of these edges lists 9/46D as the source region, as indicated by its first position in the tuple. Note also that the ‘PHT00’ acronym is no longer prepended to the edges. ORT or mORT will produce a lot of edges labeled as known-absent (EC=N) or unknown (EC=U). This is a natural and expected outcome of translation, as there will be cases for which there is clear evidence that two regions are not connected or insufficient evidence to determine whether a present edge can be assigned. Even though most users will want these edges removed from their final graph, **add_translated_edges** will not remove these edges by default because users may want to interrogate these edges. The **strip_absent_and_unknown_edges** function will remove these edges however and this is demonstrated in Figure 10 inset E. Indeed, **strip_absent_and_unknown_edges** removed all but 4 connections.

APPENDIX E: Tutorial for Querying and Registering all available CoCoMac data onto PHT00

<Figure 11 About Here>

FIGURE CAPTIONS

Figure1: Cartoon demonstration of CoCoMac and coordinate-free registration. CoCoMac stores mapping statements that describe the spatial relationship between two regions in different parcellation schemes and connectivity statements that describe the extent of staining in two regions in the context of one injection. These statements are combined during coordinate-free registration to allow known connections in one parcellation scheme to be translated to connections in a second parcellation scheme. In coordinate-free registration, one first gathers mapping statements that exhaust the relationships between regions in the two parcellation schemes involved in the connection one wants to translate. All relevant connectivity statements in the input map are then gathered, and a set of logical rules is consulted to conclude whether these imply certain connections in the output map. In this simplified cartoon, the aim is to determine whether Region C makes a connection to Region E. Region A and Region C are part of different parcellation schemes, but based on the mapping data in CoCoMac one can determine that Region C is a subregion of Region A. We then gather the relevant connectivity statements. In this case, we find that there is a connection between Regions A and E. Stain is present throughout Region E (pictured on the right top) as well as Region A. Because C is smaller than A and A connects to E through its entire extent, there is sufficient evidence to conclude that stain in Region E would result from an injection in Region C. In this example, we only needed to integrate the spatial relationships between regions on one side of the connection and the staining extents on the other side of the connection. In most real data, mapping and connectivity integration on both sides of the connection is necessary.

Figure 2: A flowchart depicting the CoCoTools processing pipeline.

Figure 3: Adjacency and difference (ORT-mORT) matrices demonstrating the similarity and differences between ORT and mORT Endgraphs for each output space tested.

Figure 4: Visualization (using the graph_view function) of results from registration of the entire CoCoMac corpus to the PHT00 and RM brain maps using mORT.

Figure 5: Sub-graphs of the graph in Figure 6 demonstrating known anatomical pathways. In particular, the connections of posterior cortical regions demonstrate separable dorsal and ventral streams and prefrontal regions show preferential connectivity within the prefrontal cortex and to high-level rather than low-level sensory regions.

Figure 6: Case illustrating how the mORT and ORT process differ. In this case ORT produces a spurious edge and mORT does not.

Figure 7: A case where ORT is misled by CoCoMac's ECs. We seek to translate the results of Experiment 1, conducted in Mapping Space A, to Mapping Space B. In A) we depict the true spatial extent of connection between the regions we are interested in. B) In Experiment 1, the source region from Mapping Space A is fully injected with tracer (i.e. EC=C) and part of the target region becomes stained (i.e. EC = P). It is critical to realize that we cannot know from this experiment how much of the injected area is genuinely connected to the target region. C) The problem arises when ORT oversteps and interprets the CoCoMac EC of C in the source region as evidence that its full extent is connected to the target region, and this D) leads ORT to infer that all of the source regions in Mapping Space B are connected to the target region. MORT, does not make this mistake because it does not interpret the extent information provided in CoCoMac's ECs. Rather mORT interprets CoCoMac ECs as either "Present", or "Absent", and for E) Experiment 1, it infers that stain is "Present" in both the source and target regions. Based on this, we do not have enough information to know whether any of the smaller source regions in Mapping Space B are connected to the target region. Thus for F) Experiment 1, mORT assigns "Unknown" for each of these connections.

Figure 8: An Interactive Python Session demonstrating how to query CoCoMac with CoCoTools.

Figure 9: An Interactive Python Session demonstrating the pre-processing steps needed to prepare graph data for coordinate-free registration in CoCoTools.

Figure 10: An Interactive Python Session demonstrating the ORT and mORT coordinate-free registration algorithms that are implemented within CoCoTools.

Figure 11: Demonstration of the CoCoTools processing pipeline using all of the datasets available in CoCoTools. Listed are all of the Python commands need to query, preprocess, co-register and post-process all available CoCoMac datasets to the PHT00 atlas space.

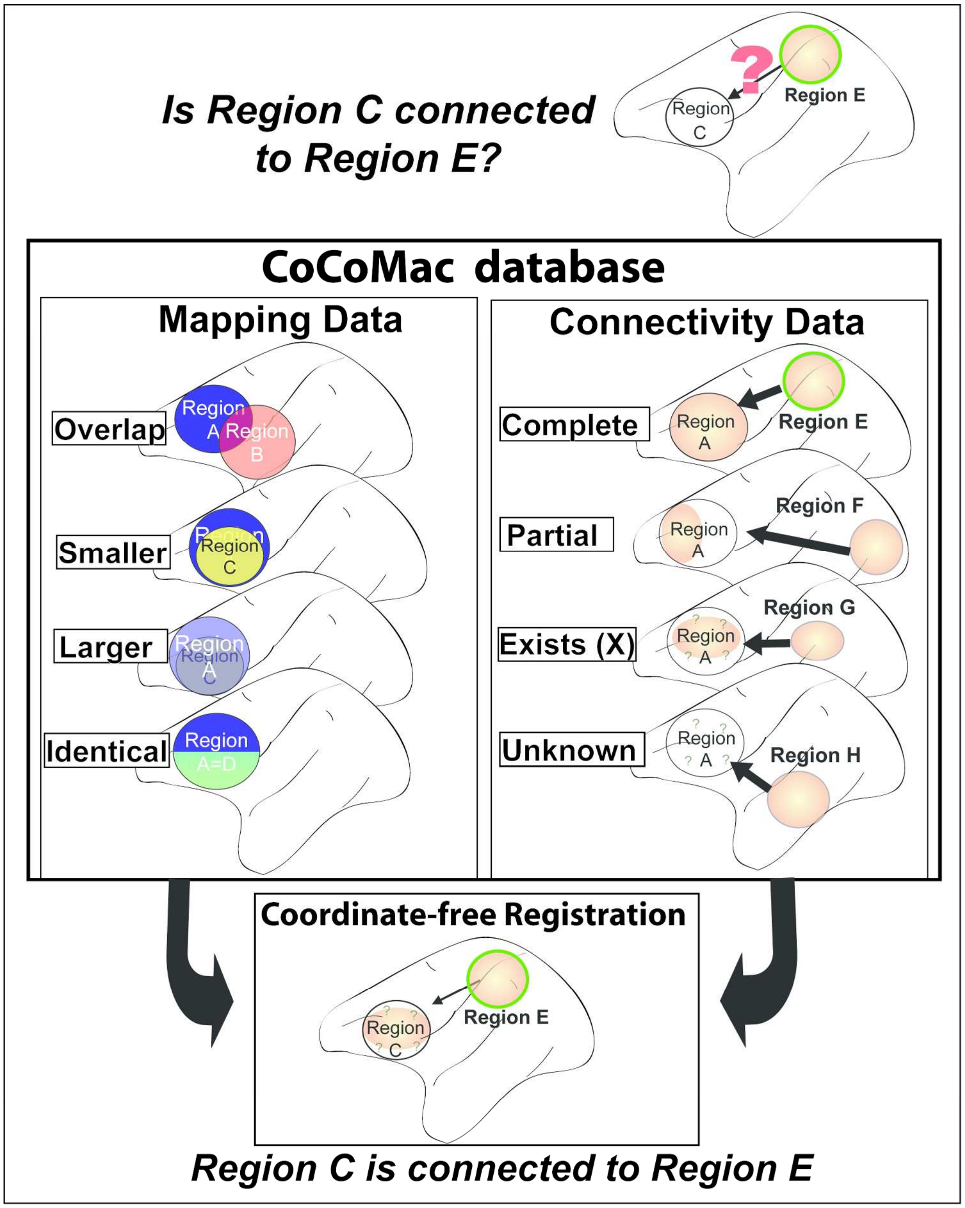
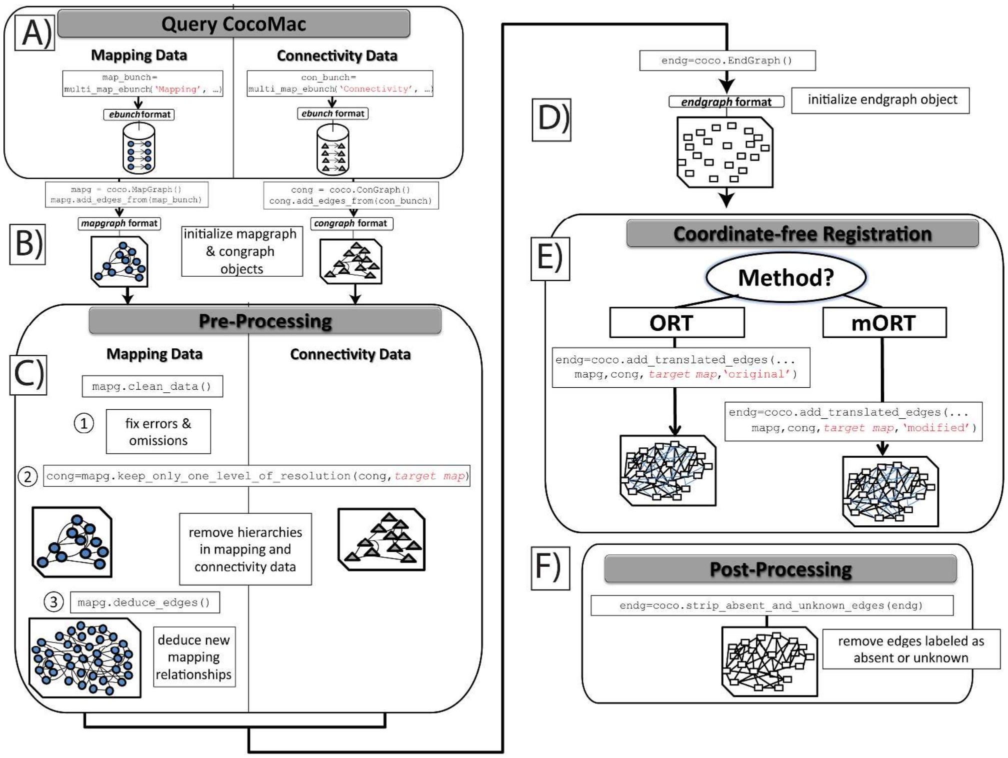


Figure1: Cartoon demonstration of CoCoMac and coordinate-free registration. CoCoMac stores mapping statements that describe the spatial relationship between two regions in different parcellation schemes and connectivity statements that describe the extent of staining in two regions in the context of one injection. These statements are combined during coordinate-free registration to allow known connections in one parcellation scheme to be translated to connections in a second parcellation scheme. In coordinate-free registration, one first gathers mapping statements that exhaust the relationships between regions in the two parcellation schemes involved in the connection one wants to translate. All relevant connectivity statements in the input map are then gathered, and a set of logical rules is consulted to conclude whether these imply certain connections in the output map. In this simplified cartoon, the aim is to determine whether Region C makes a connection to Region E. Region A and Region C are part of different parcellation schemes, but based on the mapping data in CoCoMac one can determine that Region C is a subregion of Region A. We then gather the relevant connectivity statements. In this case, we find that there is a connection between Regions A and E. Stain is present throughout Region E (pictured on the right top) as well as Region A.

Because C is smaller than A and A connects to E through its entire extent, there is sufficient evidence to conclude that stain in Region E would result from an injection in Region C. In this example, we only needed to integrate the spatial relationships between regions on one side of the connection and the staining extents on the other side of the connection. In most real data, mapping and connectivity integration on both sides of the connection is necessary.
203x252mm (300 x 300 DPI)



```
map_bunch=
multi_map_ebunch('Mapping', …)
```

```
con_bunch=
multi_map_ebunch('Connectivity', …)
```

```
mapg = coco.MapGraph()
mapg.add_edges_from(map_bunch)
```

```
cong = coco.ConGraph()
cong.add_edges_from(con_bunch)
```

```
mapg.clean_data()
```

```
cong=mapg.keep_only_one_level_of_resolution(cong,target map)
```

```
mapg.deduce_edges()
```

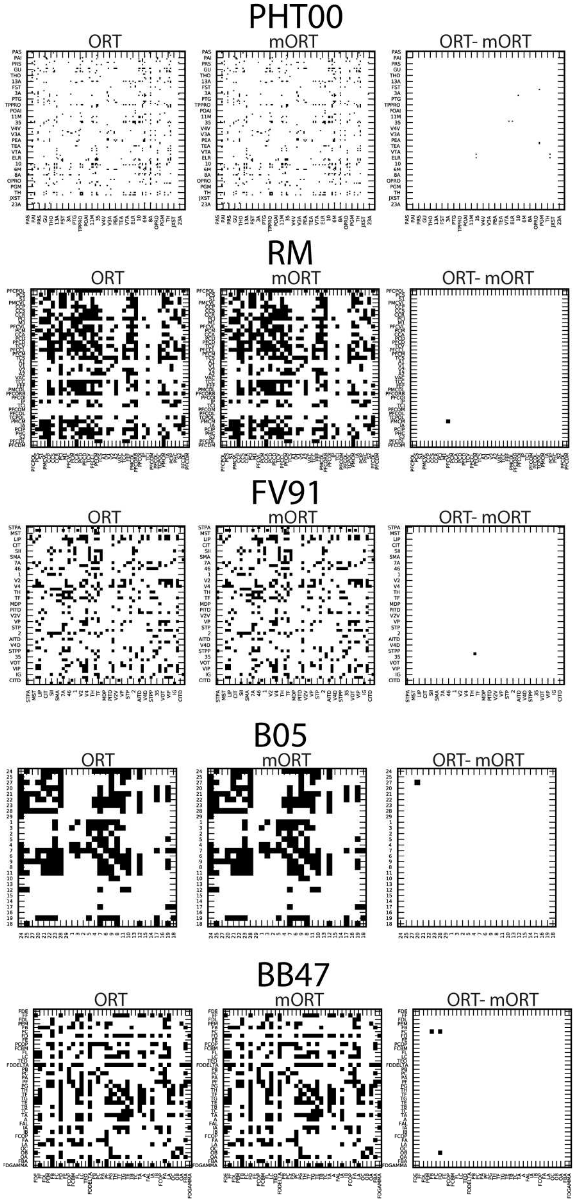
```
endg=coco.EndGraph()
```

```
endg=coco.add_translated_edges(...
   mapg,cong,target map,'original')
```

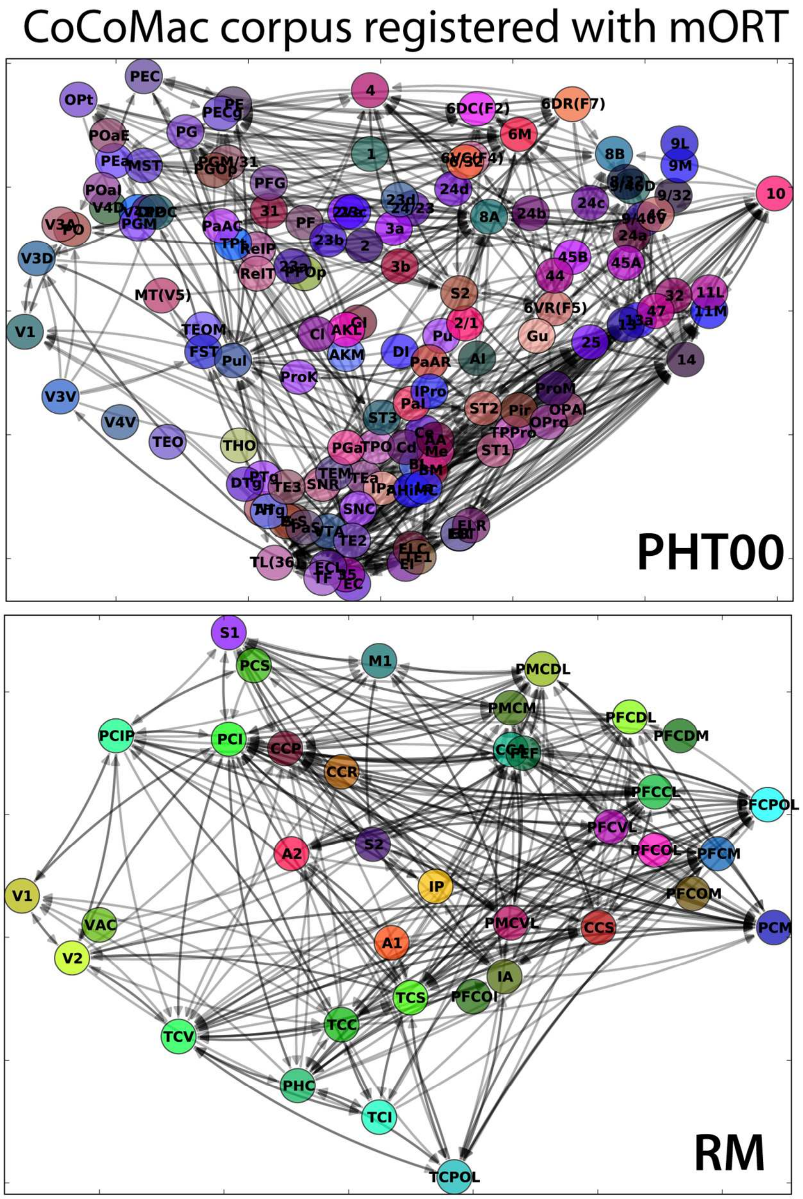
```
endg=coco.add_translated_edges(...
   mapg,cong,target map,'modified')
```

```
endg=coco.strip_absent_and_unknown_edges(endg)
```

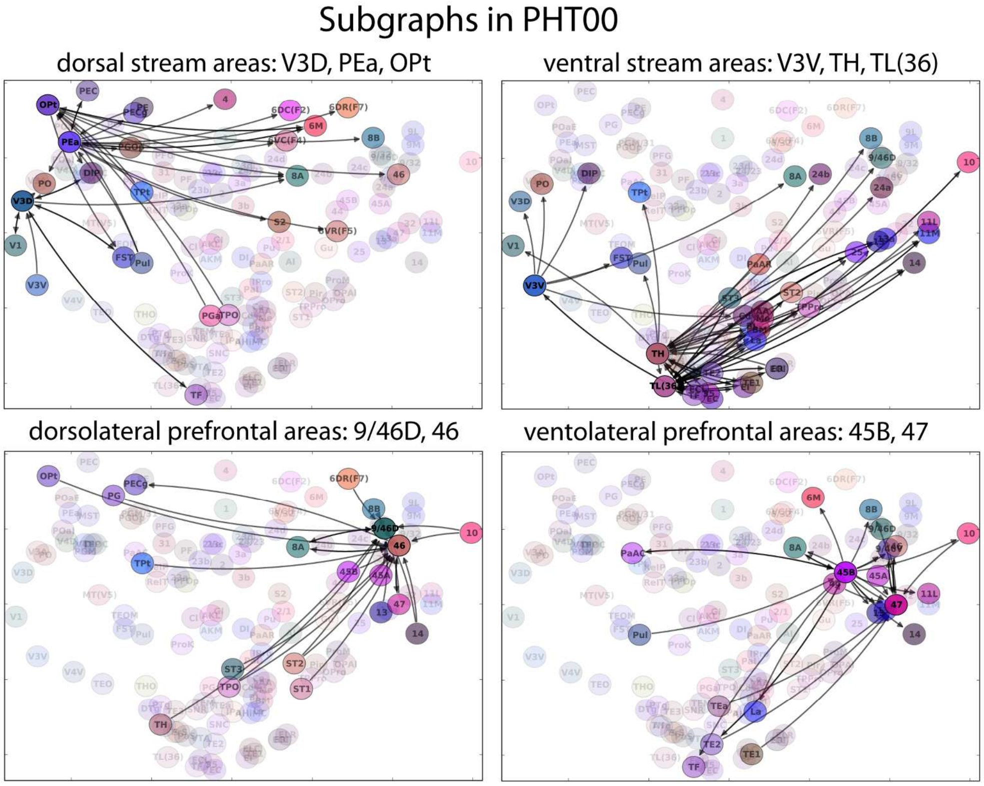
Figure 2: A flowchart depicting the CoCoTools processing pipeline.
273x204mm (300 x 300 DPI)



Adjacency and difference (ORT-mORT) matrices demonstrating the similarity and differences between ORT and mORT Endgraphs for each output space tested.
100x210mm (300 x 300 DPI)



Visualization (using the graph_view function) of results from registration of the entire CoCoMac corpus to the PHT00 and RM brain maps using mORT.
130x180mm (300 x 300 DPI)



Subgraphs in PHT00

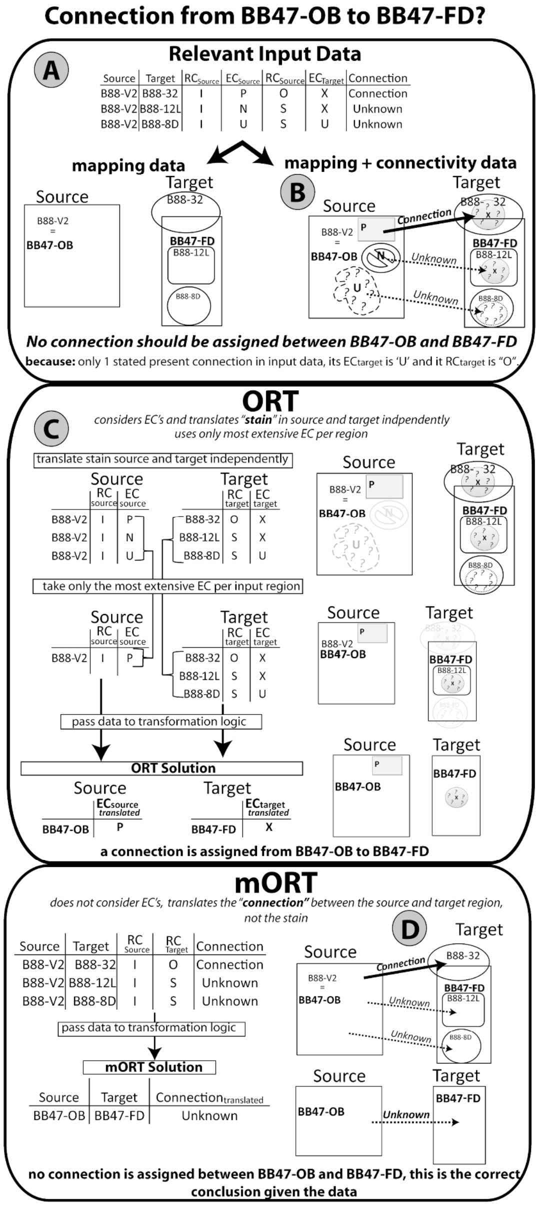
dorsal stream areas: V3D, PEa, OPt

ventral stream areas: V3V, TH, TL(36)

dorsolateral prefrontal areas: 9/46D, 46

ventolateral prefrontal areas: 45B, 47

Sub-graphs of the graph in Figure 6 demonstrating known anatomical pathways. In particular, the connections of posterior cortical regions demonstrate separable dorsal and ventral streams and prefrontal regions show preferential connectivity within the prefrontal cortex and to high-level rather than low-level sensory regions.
197x157mm (300 x 300 DPI)



Connection from BB47-OB to BB47-FD?

Relevant Input Data

Source	Target	RC_{Source}	EC_{Source}	RC_{Source}	EC_{Target}	Connection
B88-V2	B88-32	I	P	O	X	Connection
B88-V2	B88-12L	I	N	S	X	Unknown
B88-V2	B88-8D	I	U	S	U	Unknown

No connection should be assigned between BB47-OB and BB47-FD

because: only 1 stated present connection in input data, its ECtarget is 'U' and it RCtarget is "O".

ORT

*considers EC's and translates "**stain**" in source and target indpendently uses only most extensive EC per region*

translate stain source and target independently

Source	RC source	EC source
B88-V2	I	P
B88-V2	I	N
B88-V2	I	U

Target	RC target	EC target
B88-32	O	X
B88-12L	S	X
B88-8D	S	U

take only the most extensive EC per input region

Source	RC source	EC source
B88-V2	I	P

Target	RC target	EC target
B88-32	O	X
B88-12L	S	X
B88-8D	S	U

pass data to transformation logic

ORT Solution

Source	ECsource translated
BB47-OB	P

Target	ECtarget translated
BB47-FD	X

a connection is assigned from BB47-OB to BB47-FD

mORT

*does not consider EC's, translates the "**connection**" between the source and target region, not the stain*

Source	Target	RC Source	RC Target	Connection
B88-V2	B88-32	I	O	Connection
B88-V2	B88-12L	I	S	Unknown
B88-V2	B88-8D	I	S	Unknown

pass data to transformation logic

mORT Solution

Source	Target	Connectiontranslated
BB47-OB	BB47-FD	Unknown

no connection is assigned between BB47-OB and BB47-FD, this is the correct conclusion given the data

Figure 6: Case illustrating how the mORT and ORT process differ. In this case ORT produces a spurious edge and mORT does not.
123x267mm (300 x 300 DPI)

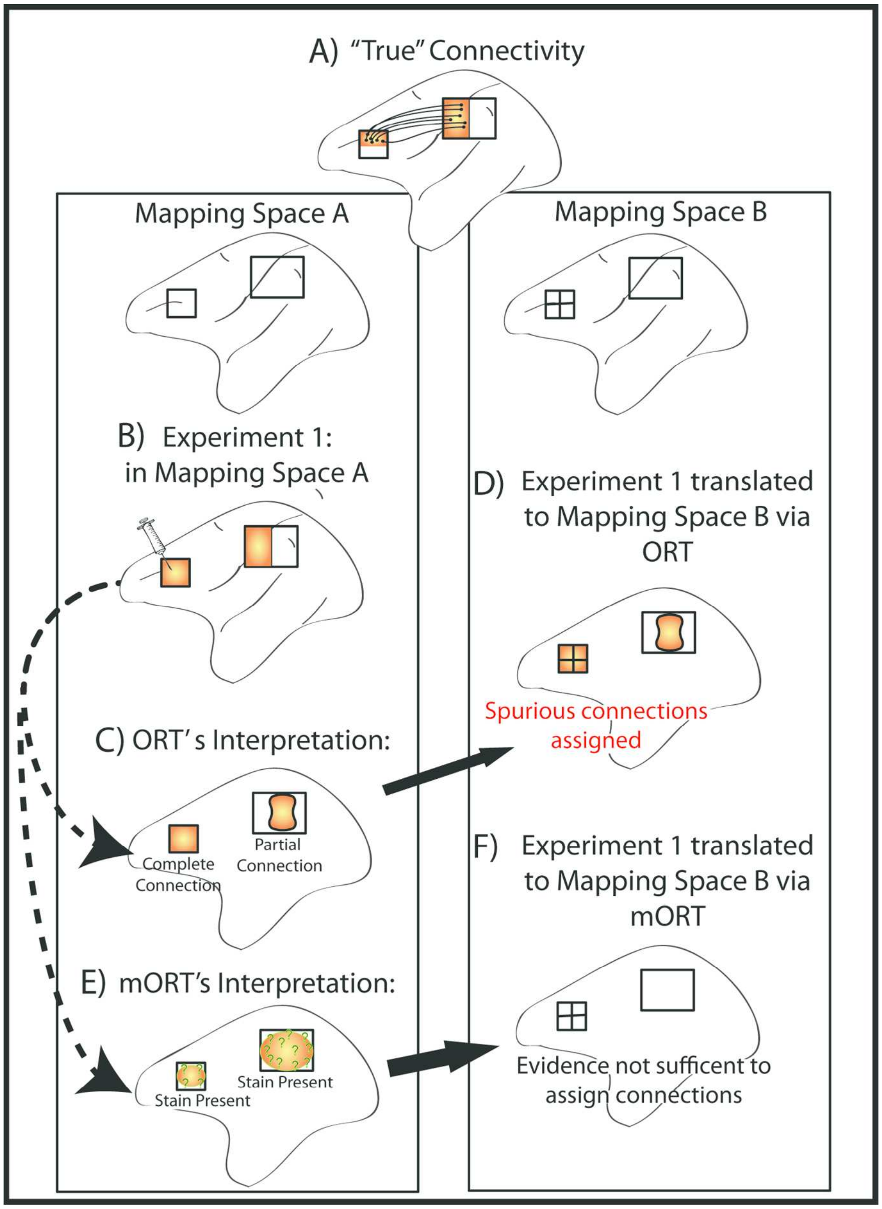
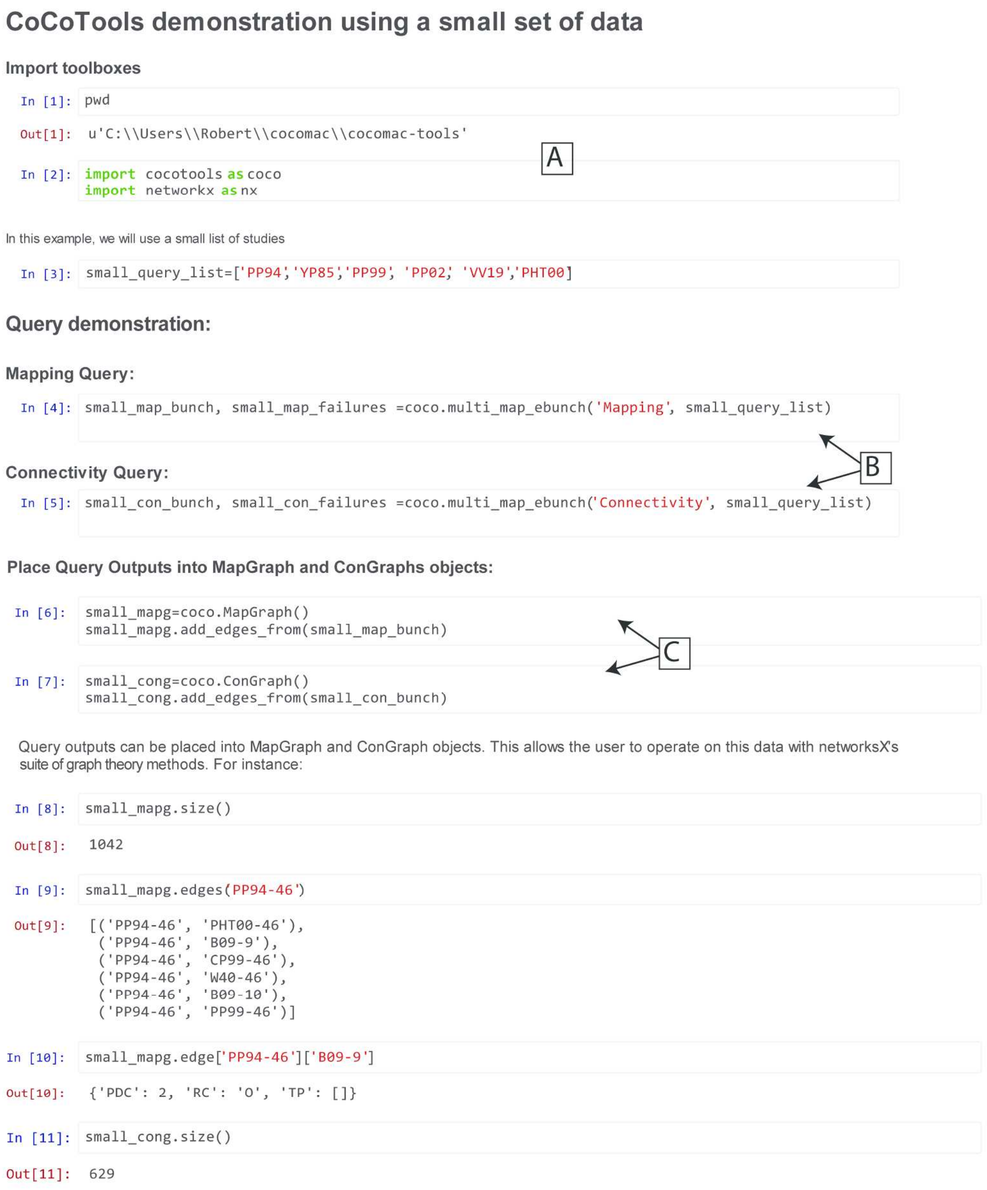


Figure 7: A case where ORT is misled by CoCoMac's ECs. We seek to translate the results of Experiment 1, conducted in Mapping Space A, to Mapping Space B. In A) we depict the true spatial extent of connection between the regions we are interested in. B) In Experiment 1, the source region from Mapping Space A is fully injected with tracer (i.e. EC=C) and part of the target region becomes stained (i.e. EC = P). It is critical to realize that we cannot know from this experiment how much of the injected area is genuinely connected to the target region. C) The problem arises when ORT oversteps and interprets the CoCoMac EC of C in the source region as evidence that its full extent is connected to the target region, and this D) leads ORT to infer that all of the source regions in Mapping Space B are connected to the target region. MORT, does not make this mistake because it does not interpret the extent information provided in CoCoMac's ECs. Rather mORT interprets CoCoMac ECs as either "Present", or "Absent", and for E) Experiment 1, it infers that stain is "Present" in both the source and target regions. Based on this, we do not have enough information to know whether any of the smaller source regions in Mapping Space B are connected to the target region. Thus for F) Experiment 1, mORT assigns "Unknown" for each of these connections.

130x179mm (300 x 300 DPI)



CoCoTools demonstration using a small set of data

Import toolboxes

```
In [1]:  pwd

Out[1]:  u'C:\\Users\\Robert\\cocomac\\cocomac-tools'

In [2]:  import cocotools as coco
         import networkx as nx
```

In this example, we will use a small list of studies

```
In [3]:  small_query_list=['PP94','YP85','PP99', 'PP02', 'VV19','PHT00']
```

Query demonstration:

Mapping Query:

```
In [4]:  small_map_bunch, small_map_failures =coco.multi_map_ebunch('Mapping', small_query_list)
```

Connectivity Query:

```
In [5]:  small_con_bunch, small_con_failures =coco.multi_map_ebunch('Connectivity', small_query_list)
```

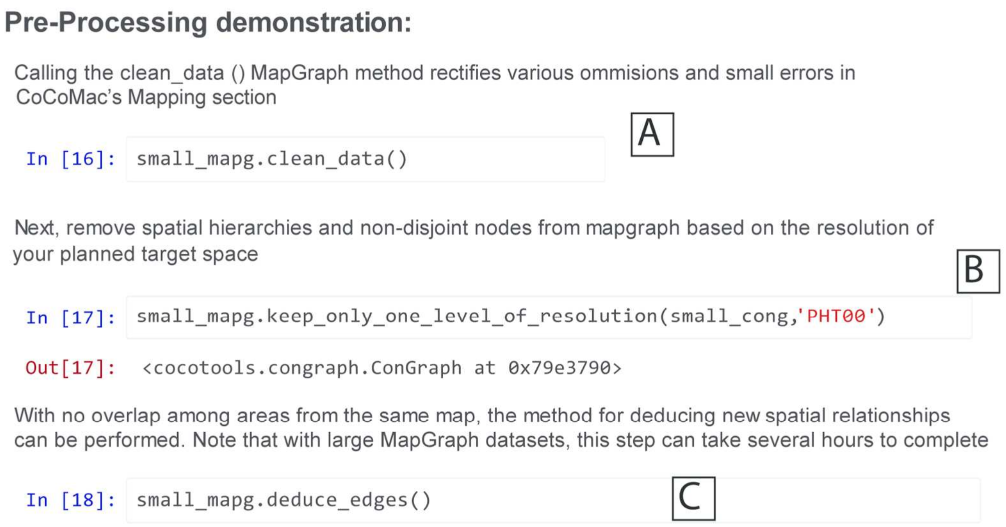
Place Query Outputs into MapGraph and ConGraphs objects:

```
In [6]:  small_mapg=coco.MapGraph()
         small_mapg.add_edges_from(small_map_bunch)

In [7]:  small_cong=coco.ConGraph()
         small_cong.add_edges_from(small_con_bunch)
```

Query outputs can be placed into MapGraph and ConGraph objects. This allows the user to operate on this data with networksX's suite of graph theory methods. For instance:

```
In [8]:  small_mapg.size()

Out[8]:  1042

In [9]:  small_mapg.edges(PP94-46')

Out[9]:  [('PP94-46', 'PHT00-46'),
          ('PP94-46', 'B09-9'),
          ('PP94-46', 'CP99-46'),
          ('PP94-46', 'W40-46'),
          ('PP94-46', 'B09-10'),
          ('PP94-46', 'PP99-46')]

In [10]: small_mapg.edge['PP94-46']['B09-9']

Out[10]: {'PDC': 2, 'RC': 'O', 'TP': []}

In [11]: small_cong.size()

Out[11]: 629
```

Figure 8: An Interactive Python Session demonstrating how to query CoCoMac with CoCoTools.
209x251mm (300 x 300 DPI)



Pre-Processing demonstration:

Calling the clean_data () MapGraph method rectifies various ommisions and small errors in CoCoMac's Mapping section

```
In [16]:  small_mapg.clean_data()
```

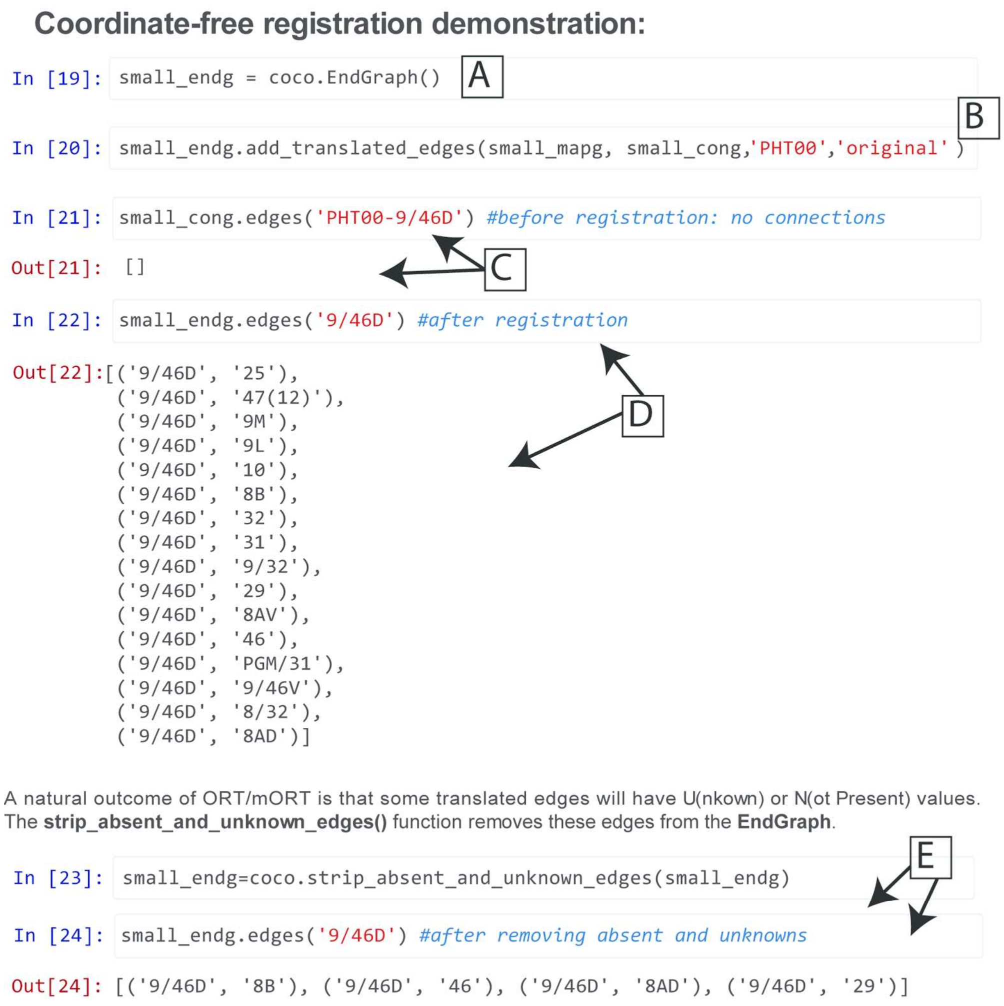
Next, remove spatial hierarchies and non-disjoint nodes from mapgraph based on the resolution of your planned target space

```
In [17]:  small_mapg.keep_only_one_level_of_resolution(small_cong,'PHT00')

Out[17]:  <cocotools.congraph.ConGraph at 0x79e3790>
```

With no overlap among areas from the same map, the method for deducing new spatial relationships can be performed. Note that with large MapGraph datasets, this step can take several hours to complete

```
In [18]:  small_mapg.deduce_edges()
```

Figure 9: An Interactive Python Session demonstrating the pre-processing steps needed to prepare graph data for coordinate-free registration in CoCoTools.
152x79mm (300 x 300 DPI)



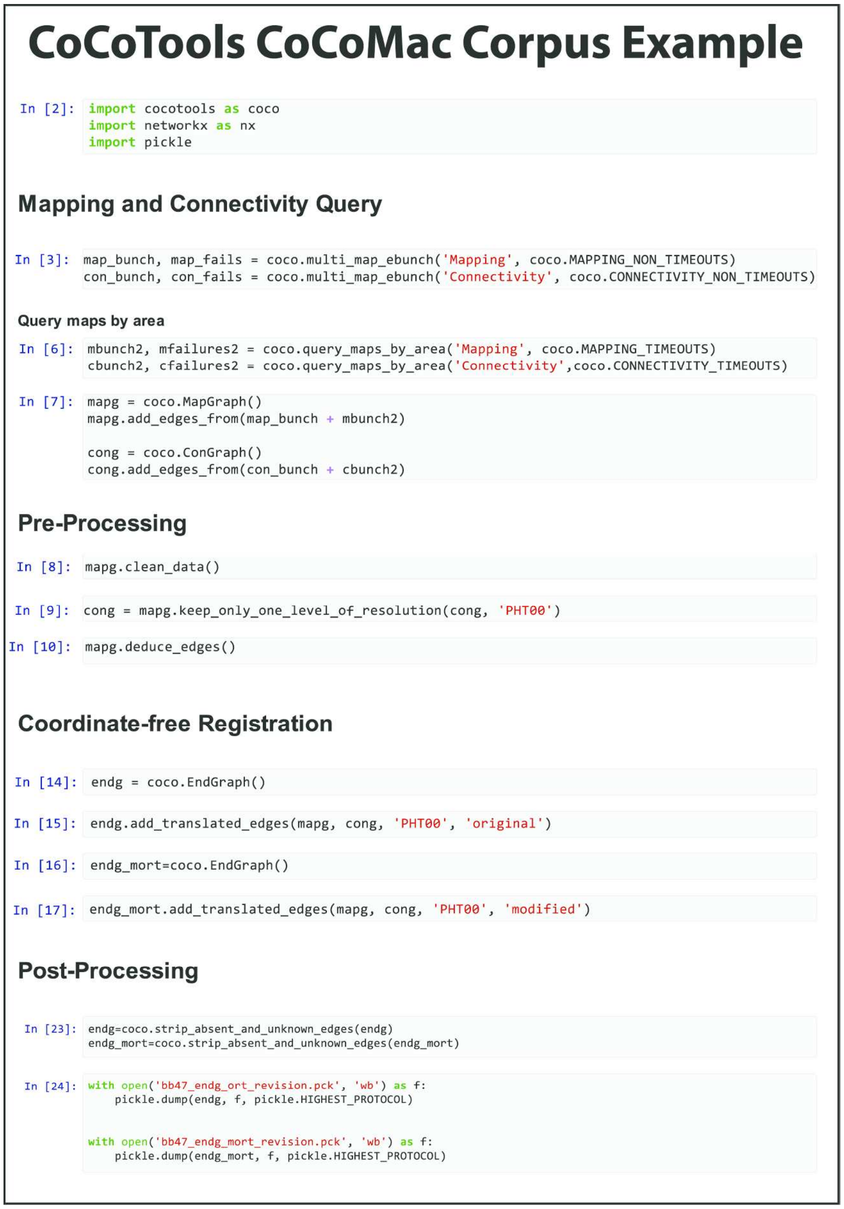
Coordinate-free registration demonstration:

```
In [19]: small_endg = coco.EndGraph()

In [20]: small_endg.add_translated_edges(small_mapg, small_cong,'PHT00','original' )

In [21]: small_cong.edges('PHT00-9/46D') #before registration: no connections

Out[21]:  []

In [22]: small_endg.edges('9/46D') #after registration

Out[22]:[('9/46D', '25'),
         ('9/46D', '47(12)'),
         ('9/46D', '9M'),
         ('9/46D', '9L'),
         ('9/46D', '10'),
         ('9/46D', '8B'),
         ('9/46D', '32'),
         ('9/46D', '31'),
         ('9/46D', '9/32'),
         ('9/46D', '29'),
         ('9/46D', '8AV'),
         ('9/46D', '46'),
         ('9/46D', 'PGM/31'),
         ('9/46D', '9/46V'),
         ('9/46D', '8/32'),
         ('9/46D', '8AD')]
```

A natural outcome of ORT/mORT is that some translated edges will have U(nkown) or N(ot Present) values. The **strip_absent_and_unknown_edges()** function removes these edges from the **EndGraph**.

```
In [23]: small_endg=coco.strip_absent_and_unknown_edges(small_endg)

In [24]: small_endg.edges('9/46D') #after removing absent and unknowns

Out[24]: [('9/46D', '8B'), ('9/46D', '46'), ('9/46D', '8AD'), ('9/46D', '29')]
```

Figure 10: An Interactive Python Session demonstrating the ORT and mORT coordinate-free registration algorithms that are implemented within CoCoTools.
157x155mm (300 x 300 DPI)



CoCoTools CoCoMac Corpus Example

```
In [2]: import cocotools as coco
        import networkx as nx
        import pickle
```

Mapping and Connectivity Query

```
In [3]: map_bunch, map_fails = coco.multi_map_ebunch('Mapping', coco.MAPPING_NON_TIMEOUTS)
        con_bunch, con_fails = coco.multi_map_ebunch('Connectivity', coco.CONNECTIVITY_NON_TIMEOUTS)
```

Query maps by area

```
In [6]: mbunch2, mfailures2 = coco.query_maps_by_area('Mapping', coco.MAPPING_TIMEOUTS)
        cbunch2, cfailures2 = coco.query_maps_by_area('Connectivity',coco.CONNECTIVITY_TIMEOUTS)
```

```
In [7]: mapg = coco.MapGraph()
        mapg.add_edges_from(map_bunch + mbunch2)

        cong = coco.ConGraph()
        cong.add_edges_from(con_bunch + cbunch2)
```

Pre-Processing

```
In [8]: mapg.clean_data()
```

```
In [9]: cong = mapg.keep_only_one_level_of_resolution(cong, 'PHT00')
```

```
In [10]: mapg.deduce_edges()
```

Coordinate-free Registration

```
In [14]: endg = coco.EndGraph()
```

```
In [15]: endg.add_translated_edges(mapg, cong, 'PHT00', 'original')
```

```
In [16]: endg_mort=coco.EndGraph()
```

```
In [17]: endg_mort.add_translated_edges(mapg, cong, 'PHT00', 'modified')
```

Post-Processing

```
In [23]: endg=coco.strip_absent_and_unknown_edges(endg)
         endg_mort=coco.strip_absent_and_unknown_edges(endg_mort)
```

```
In [24]: with open('bb47_endg_ort_revision.pck', 'wb') as f:
             pickle.dump(endg, f, pickle.HIGHEST_PROTOCOL)


         with open('bb47_endg_mort_revision.pck', 'wb') as f:
             pickle.dump(endg_mort, f, pickle.HIGHEST_PROTOCOL)
```

Figure 11: Demonstration of the CoCoTools processing pipeline using all of the datasets available in CoCoTools. Listed are all of the Python commands need to query, preprocess, co-register and post-process all available CoCoMac datasets to the PHT00 atlas space.
170x244mm (300 x 300 DPI)