Webinar 1

Introduction: Scope and Research Potential for Multidisciplinary Biodiversity Modeling and Analysis

Goals

Introduce biological and computational workflow concepts for integrating: biological specimen (species occurrence) data, phylogenetic trees, and multi-species distribution models!

Learning Objectives

Biological Objectives:

✓ **Provide context, rationale and scope for the biological research to be presented.**

Technical Objectives :

✓ **Provide context and differentiate BiotaPhy technology from desktop approaches (software, scaling, capabilities, integration)**

Webinar organization

1. Webinar Series Overview: topics and dates
2. Biological Scope Overview:
   a) Types of Data
   b) BiotaPhy One
      i. Background and Workflows
   c) BiotaPhy Two
      i. Launched Workflows
      ii. Proposed Workflows

Webinar organization

3. **Technical Scope Overview:**
   a) **BiotaPhy's architecture**
   b) **Tutorial Overview**
4. **Session Summary, Q&A and Discussion**

Webinar Series Overview

- Introduction: Scope and Research Potential for Multidisciplinary Biodiversity Modeling and Analysis (09/21/2022)
- Resolving Nomenclature: Making Appropriate Taxonomic Choices (09/28/2022)
- Clean Your Dirty Data (10/05/2022)

Webinar Series Overview

- Georeferencing with GEOLocate (10/12/2022)
- Big Data Munging (a.k.a. Splitting and merging occurrence data by taxa from multiple sources) (10/19/2022)
- Species Distribution Modeling 1 (10/26/2022)
- Species Distribution Modeling 2 (11/02/2022)

Webinar Series Overview

- Introducing Presence-Absence Matrices for Large-Scale Analyses (11/9/2022)
- Phylogenetic Diversity: Integrating Phylogenies with Species and Biogeographic Data (11/16/2022)
- Hypothesis Testing and Randomization (11/30/2022)

BiotaPhy Crew – Principal Investigators

James Beach

Stephen Smith

José Fortes

Douglas Soltis

Pamela Soltis

BiotaPhy Crew – Technical Crew

CJ Grady

Aimee Stewart

Michael Elliott

Srivattsan Sridharan

BiotaPhy Crew – Biological Crew

Dr. Hannah Marx

Hector Figueroa

Elizabeth White

Nicholas Engle-Wrye

Dr. Ryan Folk

Dr. Anthony Melton

Dr. Jon Spoelhof

Maria Cortez

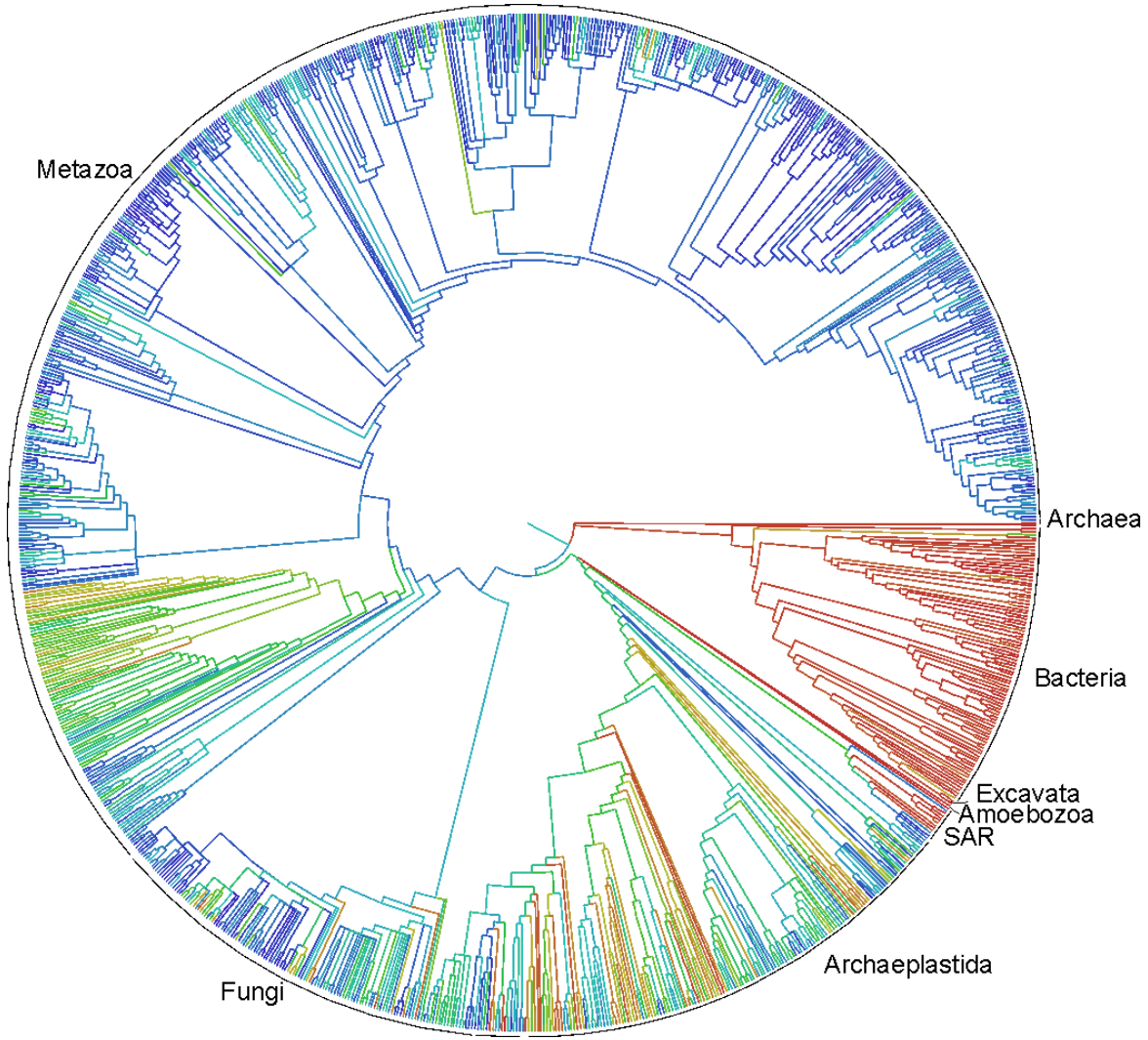
Biological Scope: Types of Data

Natural history specimens contain a wealth of data:

- ✓ Scientific name
- ✓ Date
- ✓ Collector
- ✓ Location – state, county, specific site, GPS coordinates
- ✓ Associated species

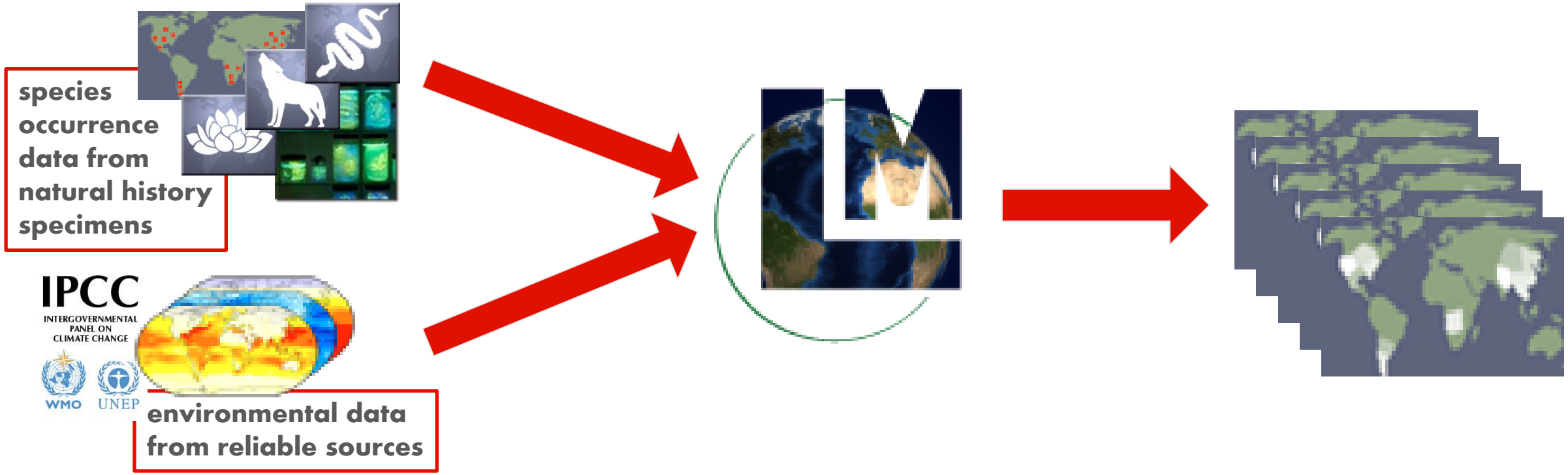
Biological Scope: Types of Data

Phylogenetic trees tell a possible tale about evolutionary history



Biological Scope: Types of Data

Species Distribution Models (SDMs) can be created by combining occurrence and environmental data



Biological Scope: Types of Data

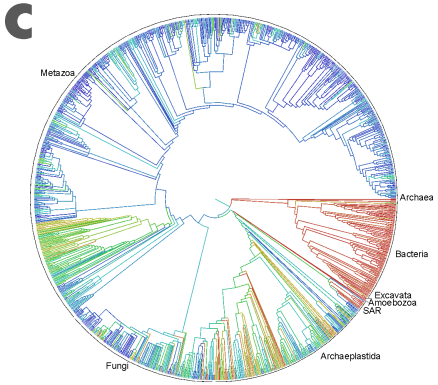
BiotaPhy

Natural history specimen data

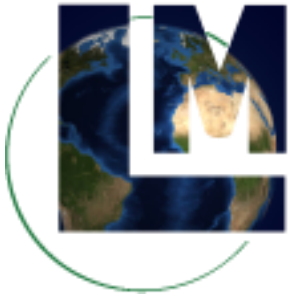
Allen et al.. 2019. Biodiversity synthesis across the green branches of the tree of life. *Nature Plants* 5:11-13.

Integration of ….

Phylogenetic trees



Species distribution models



BIOTAPHY

Biological Scope: BiotaPhy One

BIOTAPHY 1 …

and its 5 possible workflows:

RESOURCES:

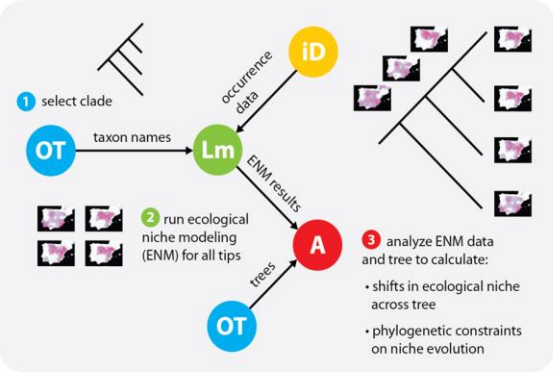
Lm Lifemapper
- ecological niche modeling
- biodiversity and range analysis
- visualization

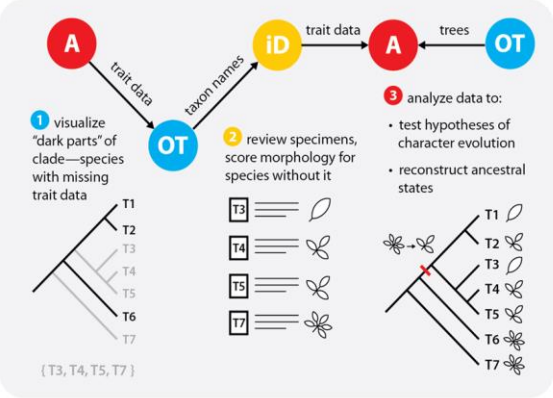
A Analysis Tool Kit
- evolutionary models
- comparative methods
- visualization

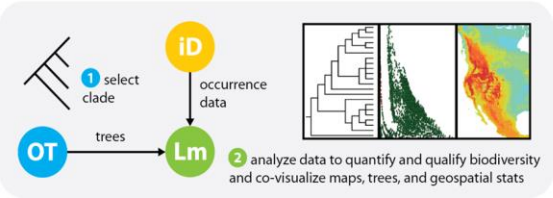
OT Open Tree of Life
- phylogenies
- taxonomy / names
- visualization

iD iDigBio
- trait data
- specimen data / images
- fossil data / images

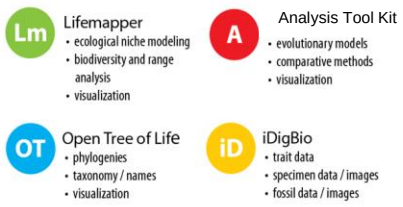
EXAMPLE WORKFLOWS:

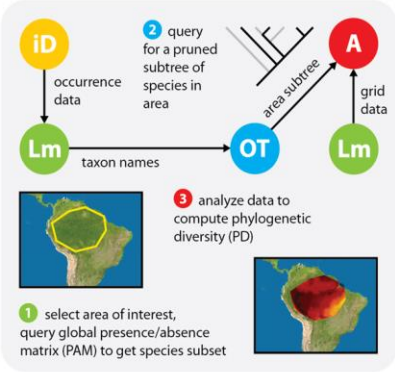


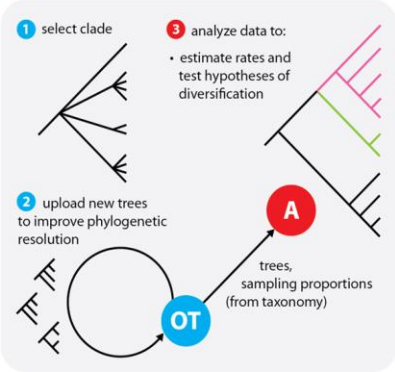




RESOURCES:







Biological Scope: BiotaPhy Two

BIOTAPHY 2 …

Evolved workflows:

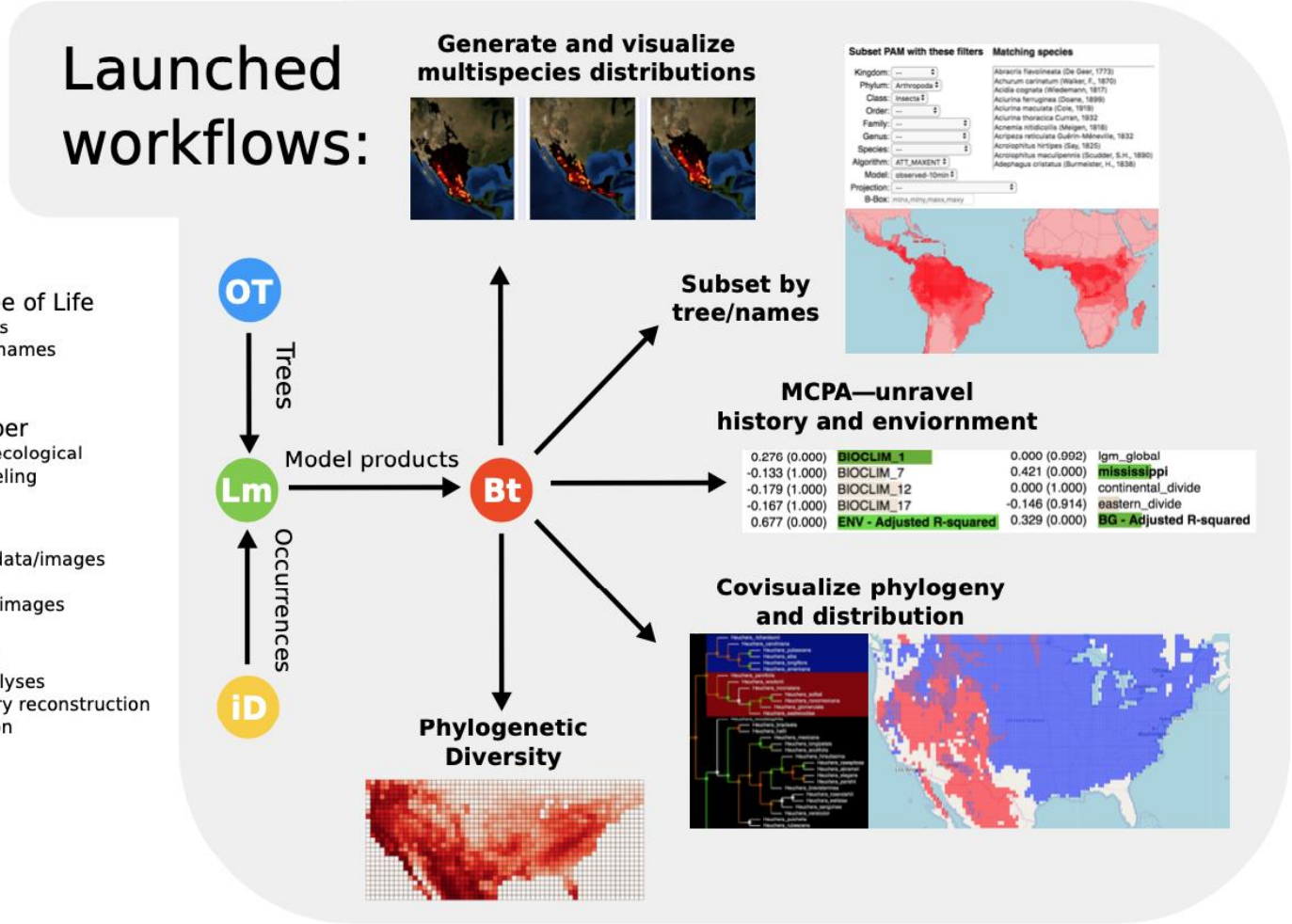
- ✓ **Launched**
- ✓ **Proposed (most have been implemented already!)**

OT Open Tree of Life
- phylogenies
- taxonomy/names

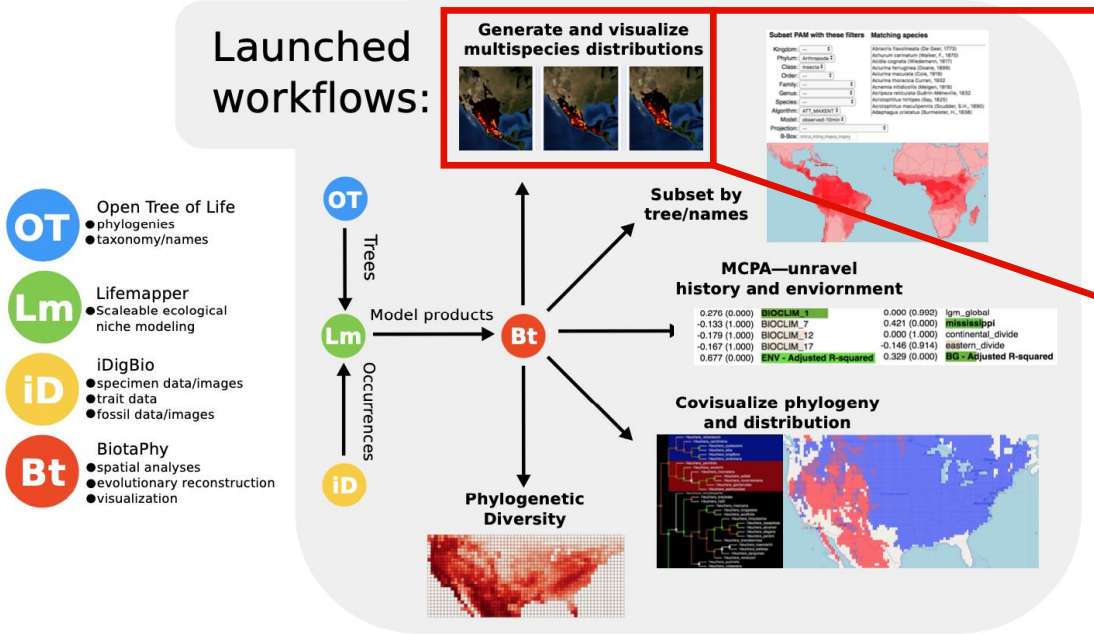
Lm Lifemapper
- Scaleable ecological niche modeling

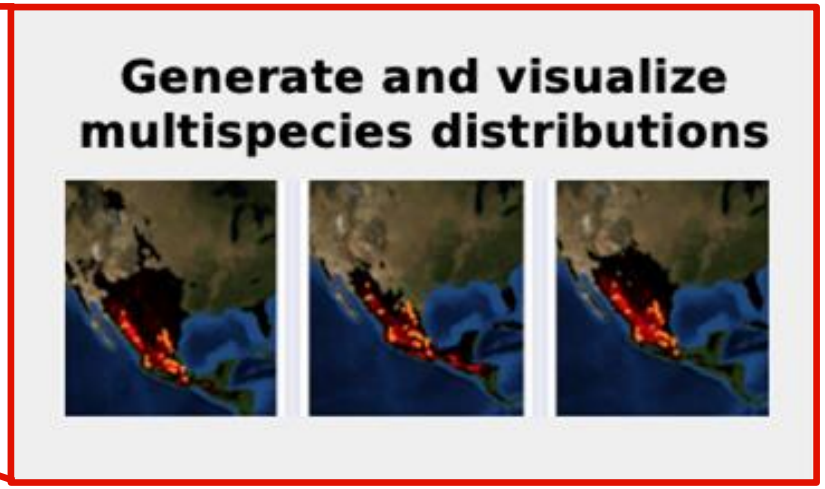
iD iDigBio
- specimen data/images
- trait data
- fossil data/images

Bt BiotaPhy
- spatial analyses
- evolutionary reconstruction
- visualization

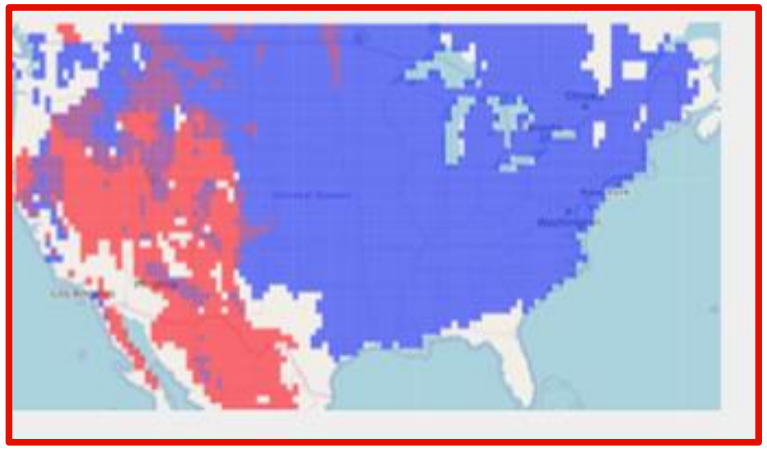


BiotaPhy Two: Launched Workflows

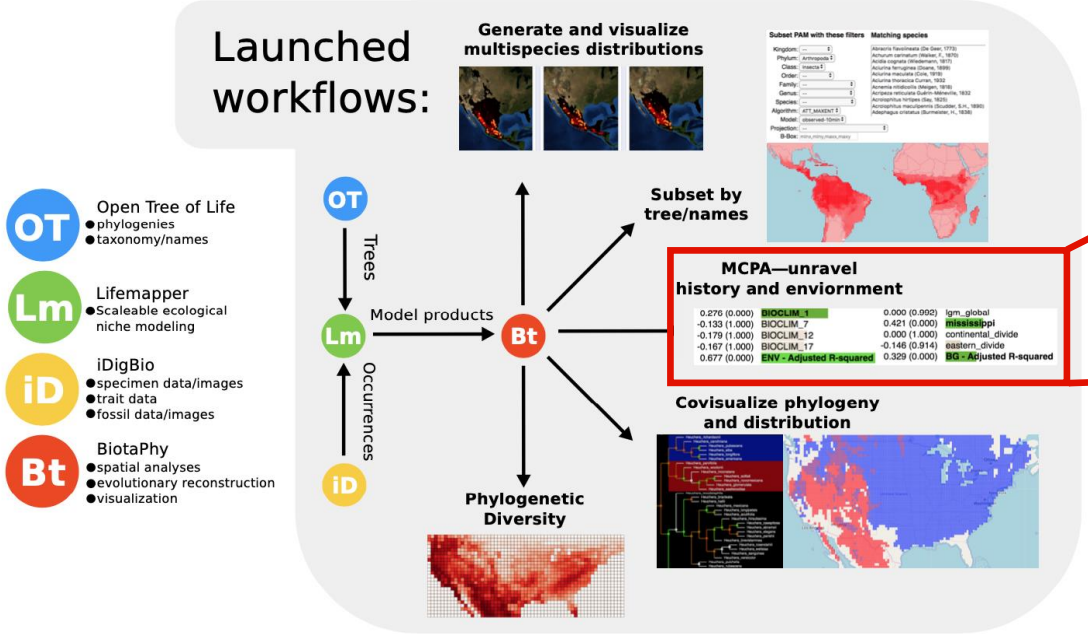


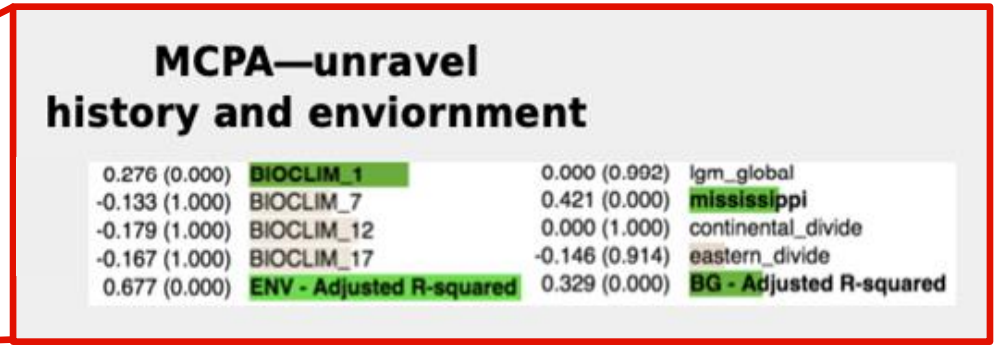


Saxifragaceae

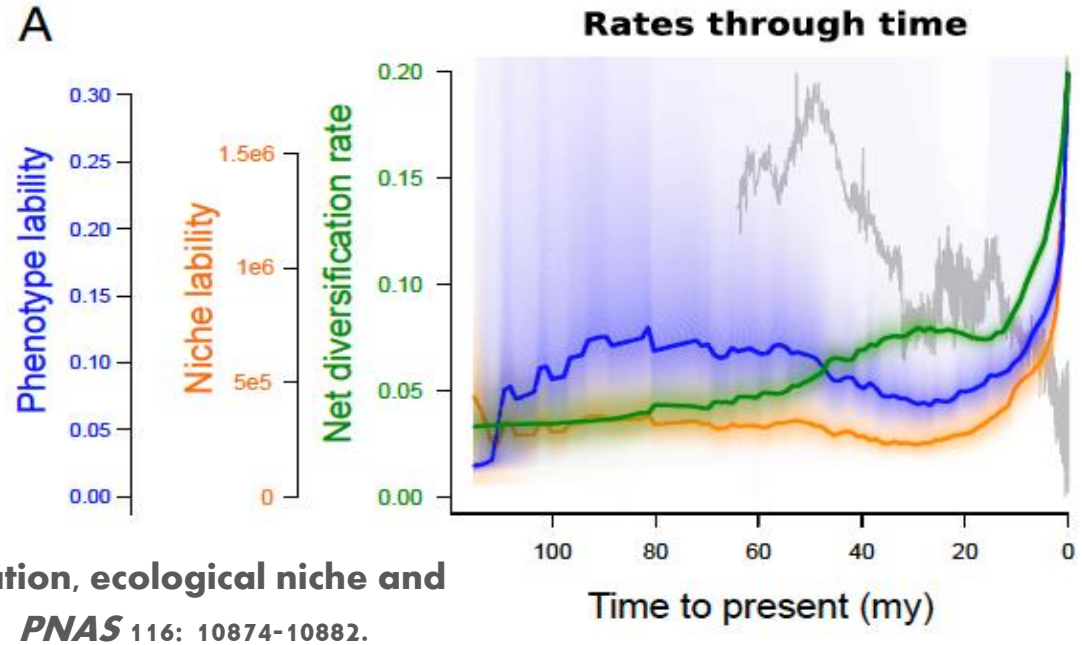


BiotaPhy Two: Launched Workflows



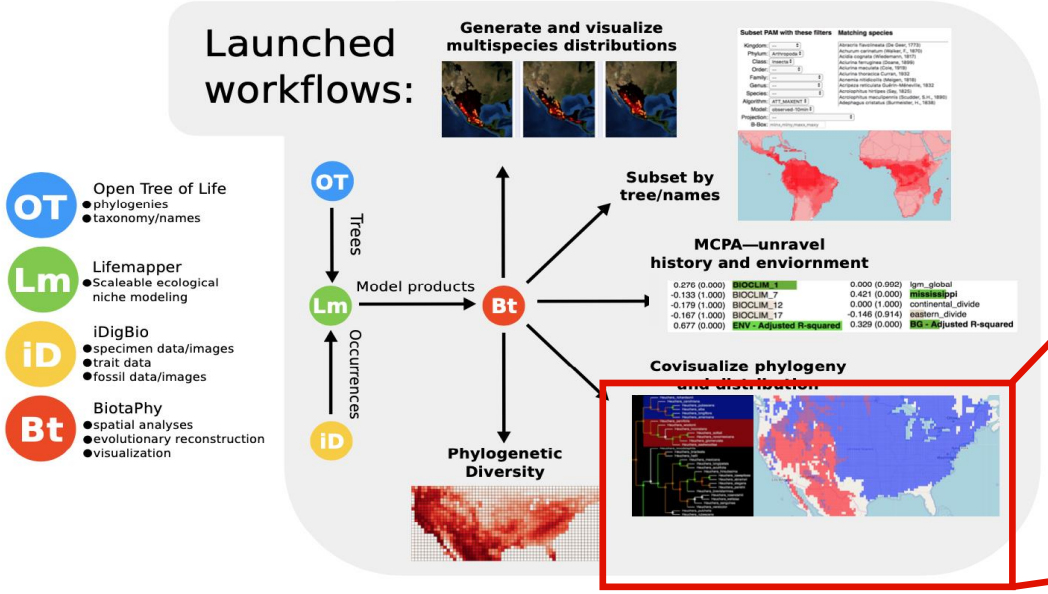


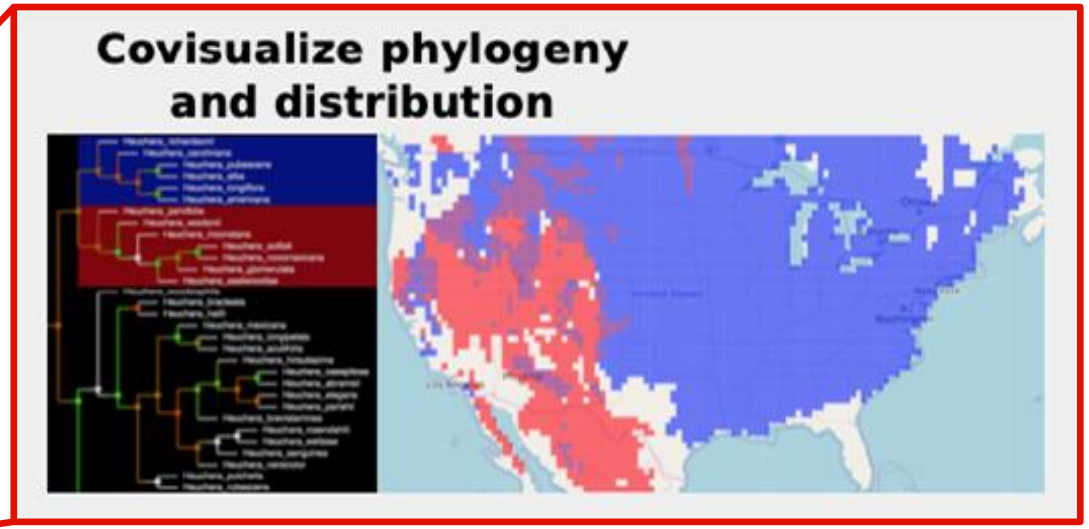
Meta-Community Phylogenetic Analysis
Test the role of biogeography and environmental niche in the diversity of Saxifragales

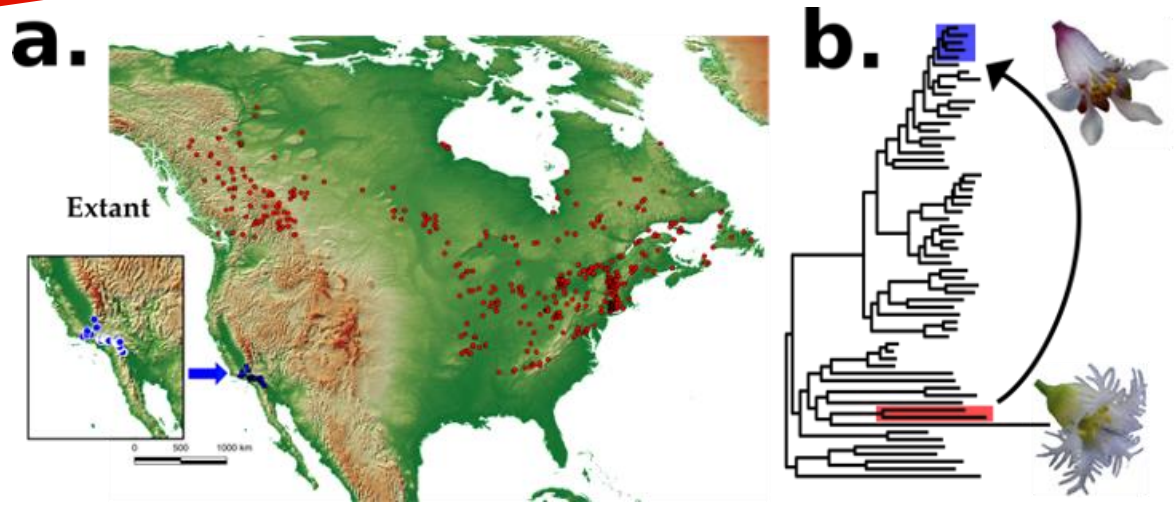


Folk et al.. 2019. Uncoupled diversification, ecological niche and phenotype in a temperate radiation. *PNAS* 116: 10874-10882.

BiotaPhy Two: Launched Workflows

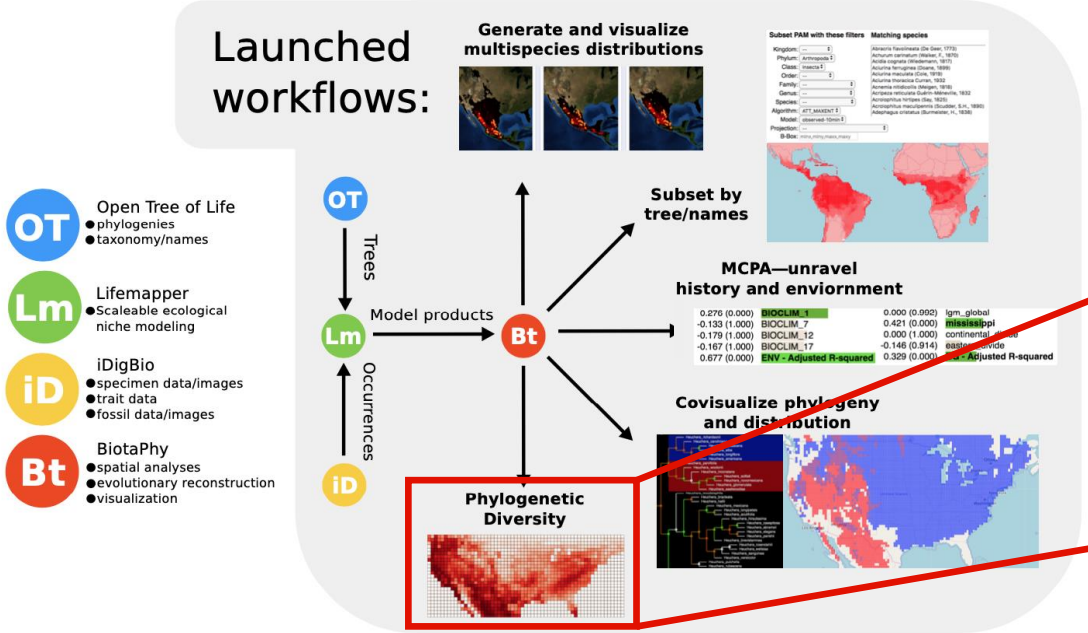


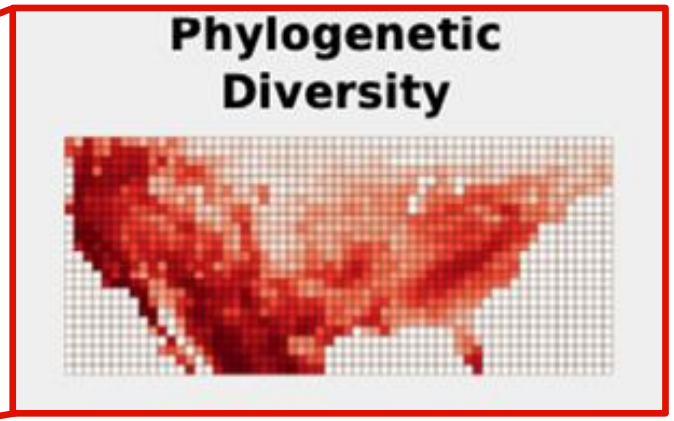


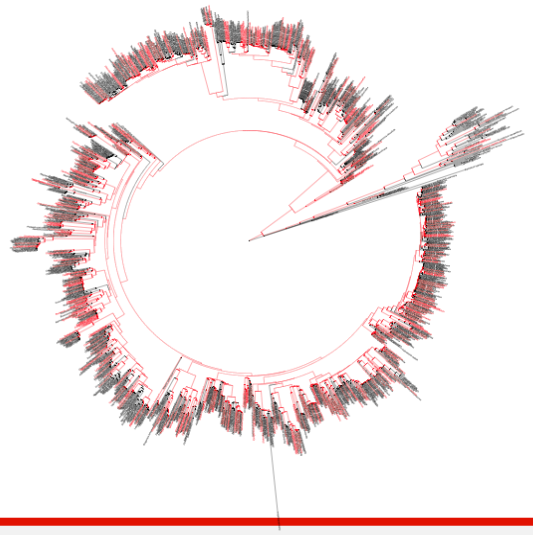


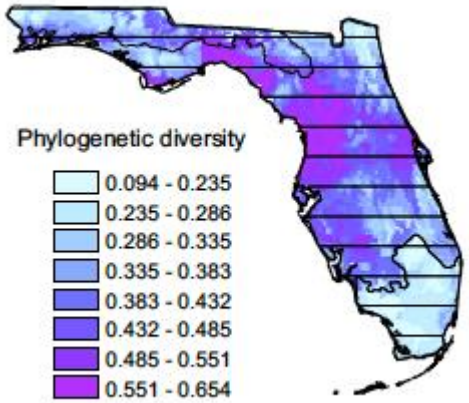
Folk et al.. 2018. Assessing ancestral niche suitability and geographic range dynamics as drivers of hybridization in *Heuchera* (Saxifragaceae). *American Naturalist* 192: 171-187.

BiotaPhy Two: Launched Workflows



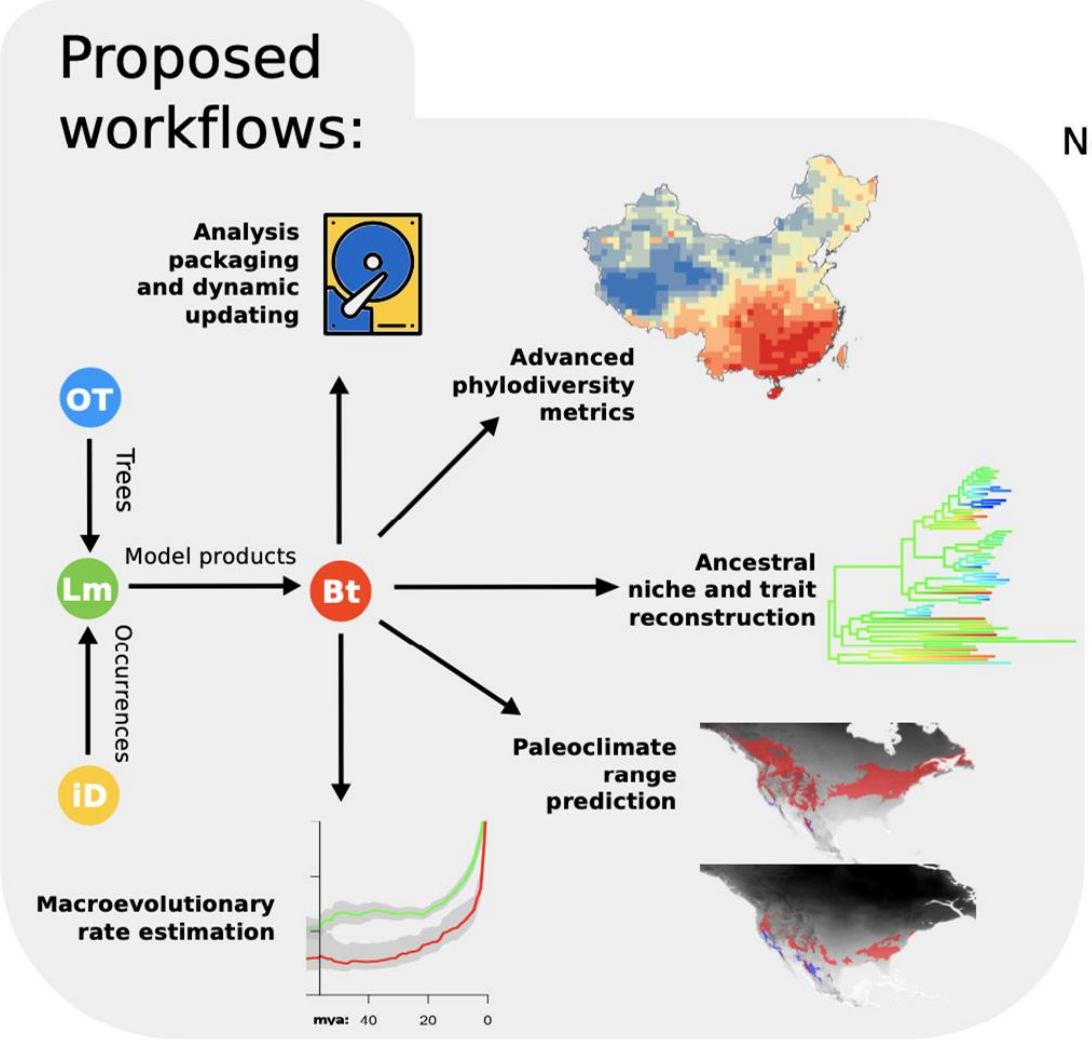






Allen, J., et al. 2019. Spatial phylogenetics of Florida vascular plants: The effects of tree uncertainty and ultrametricity. *iScience* 11: 57–70 https://doi.org/10.1016/j.isci.2018.12.002

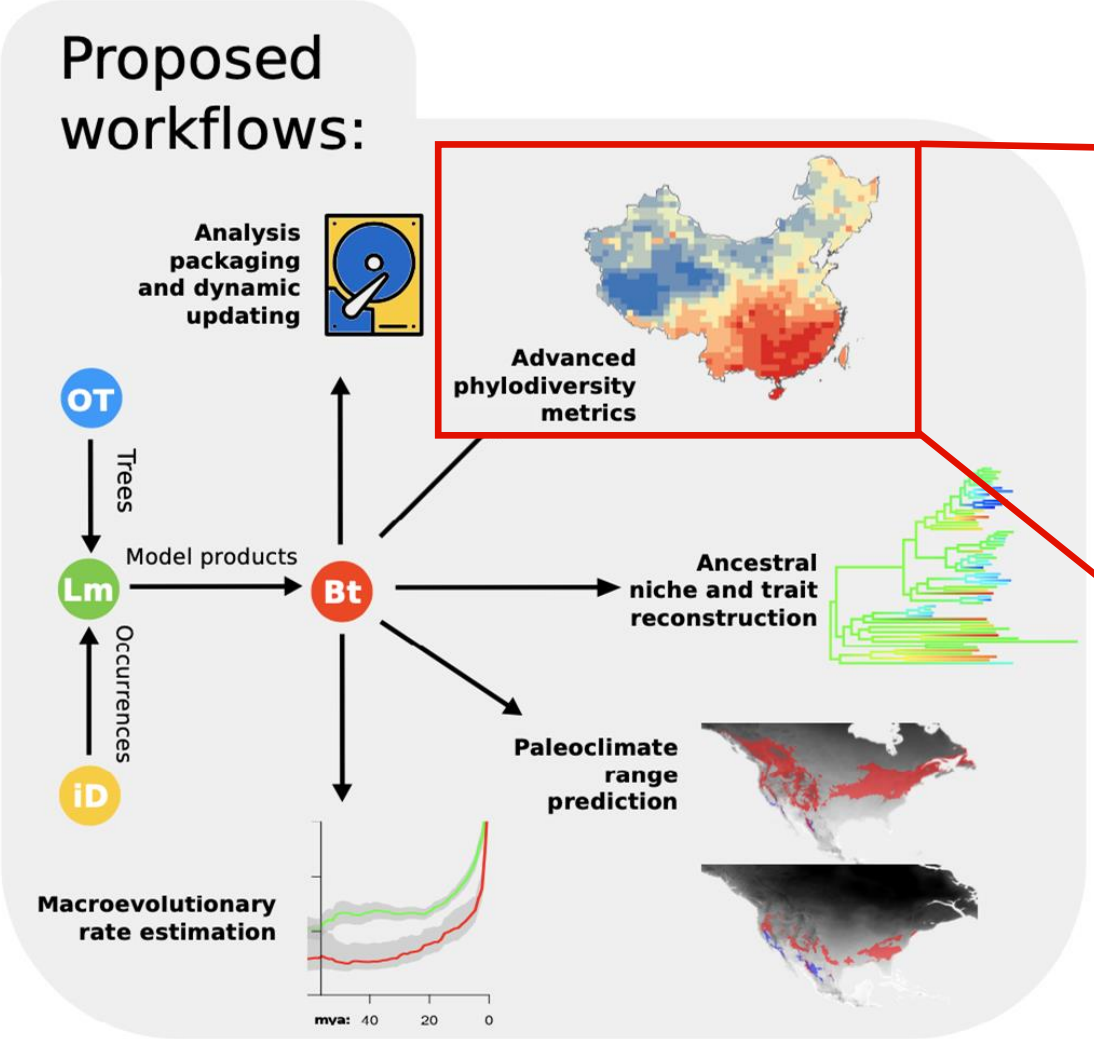
BiotaPhy Two: Proposed Workflows

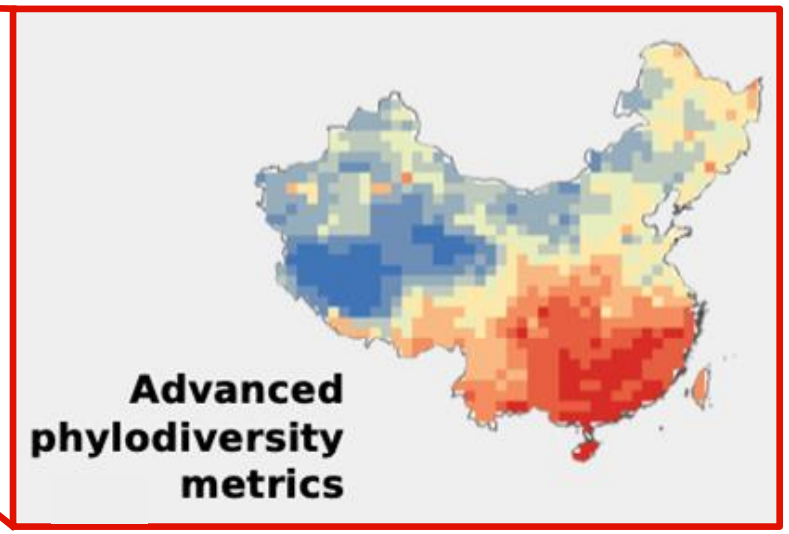


New Approaches

- Trait reconstructions, correlations with ecological factors
- Ancestral niche comparisons
- Conservation of ecological niche through time
- Predictions of ancestral suitable habitat under paleoclimate scenarios
- Community divergence times and other advanced community metrics
- Species diversification rates, trait-associated diversification
- Niche evolutionary rates

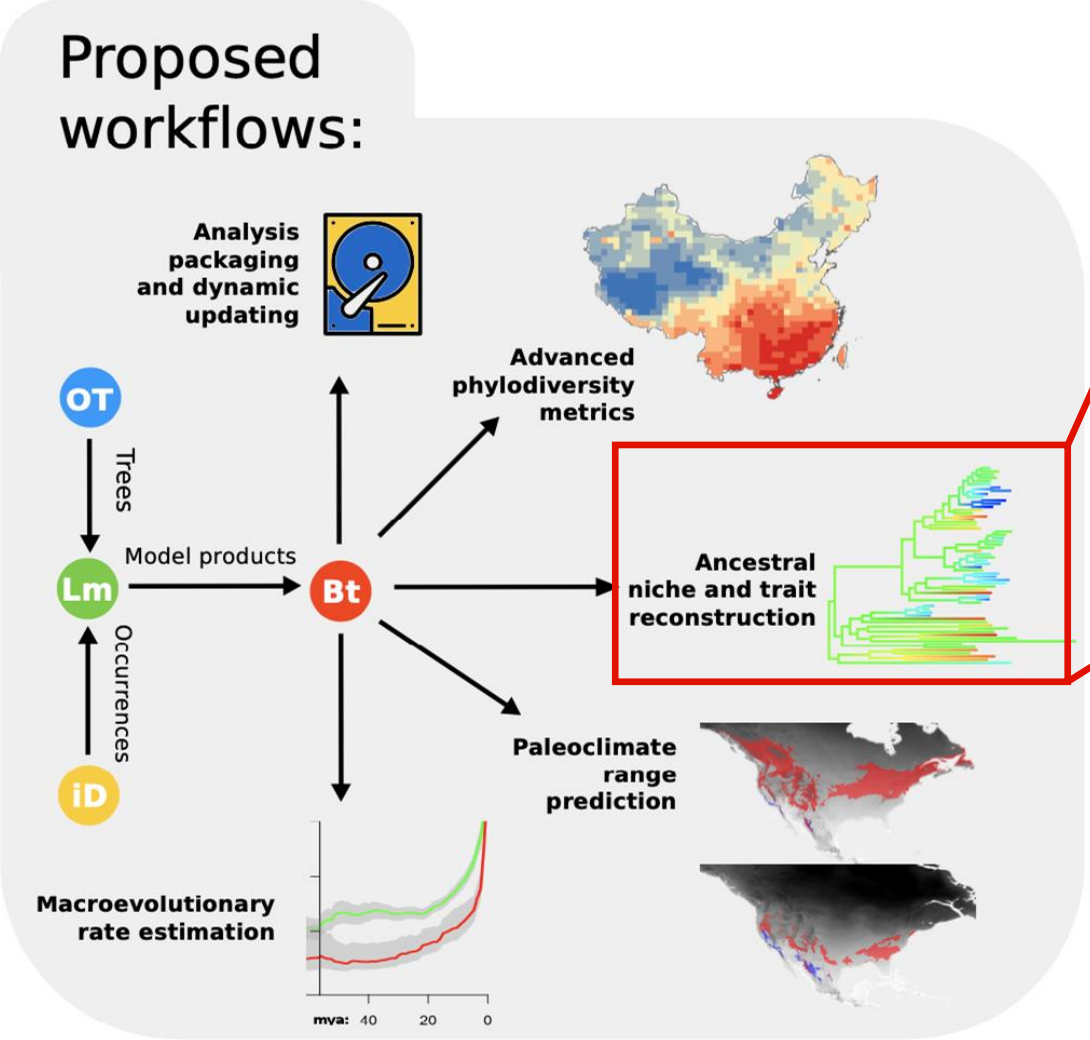
BiotaPhy Two: Proposed Workflows





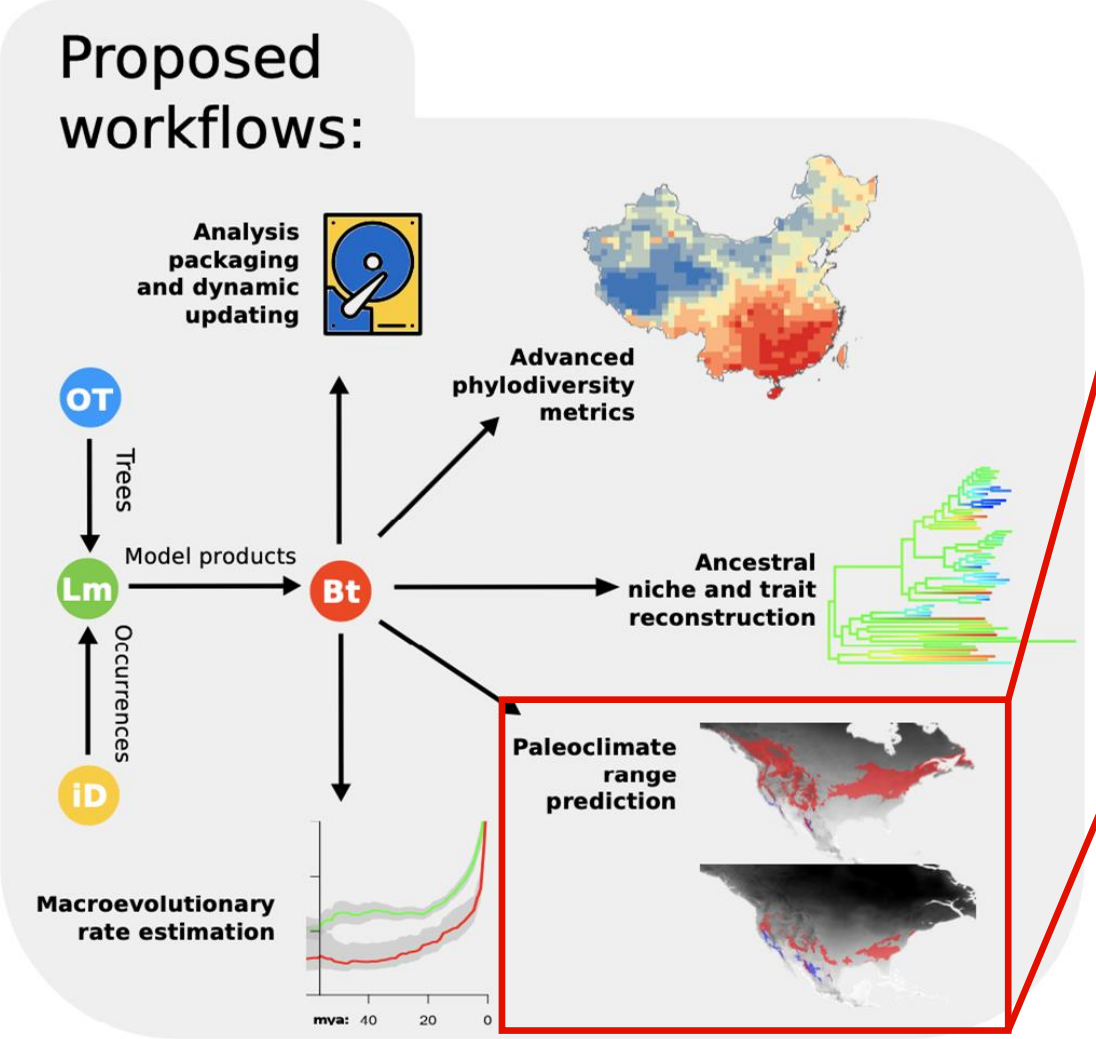
Lu, L. M., Mao L, Yang T, Ye JF, Liu B, Li HL, Sun M, Miller JT, Mathews S, Hu HH, Niu YT, Peng DX, Chen YH, Smith SA, Chen M, Xiang KL, Le CT, Dang VC, Lu AM, Soltis PS, Soltis DE, Li JH, Chen ZD. 2018. Evolutionary history of the angiosperm flora of China. Nature 554:234–238. doi:10.1038/nature25485.

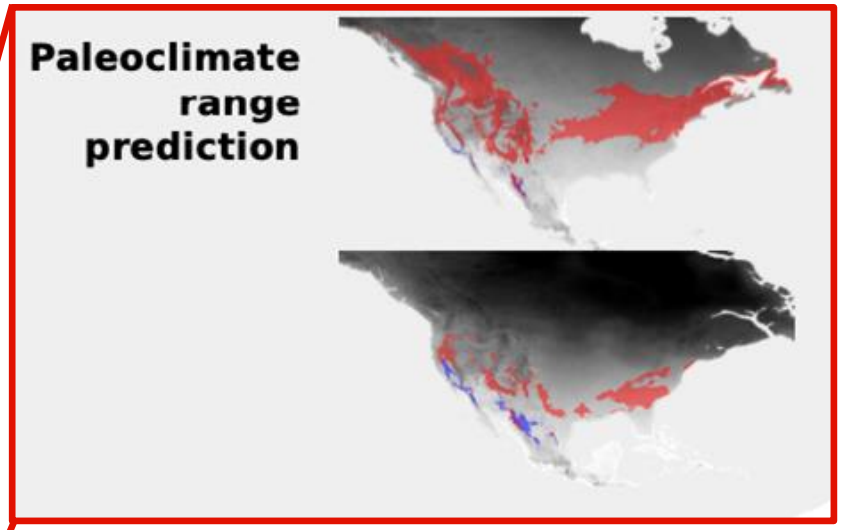
BiotaPhy Two: Proposed Workflows



Folk et al.. 2018. Assessing ancestral niche suitability and geographic range dynamics as drivers of hybridization in *Heuchera* (Saxifragaceae). *American Naturalist* 192: 171-187.

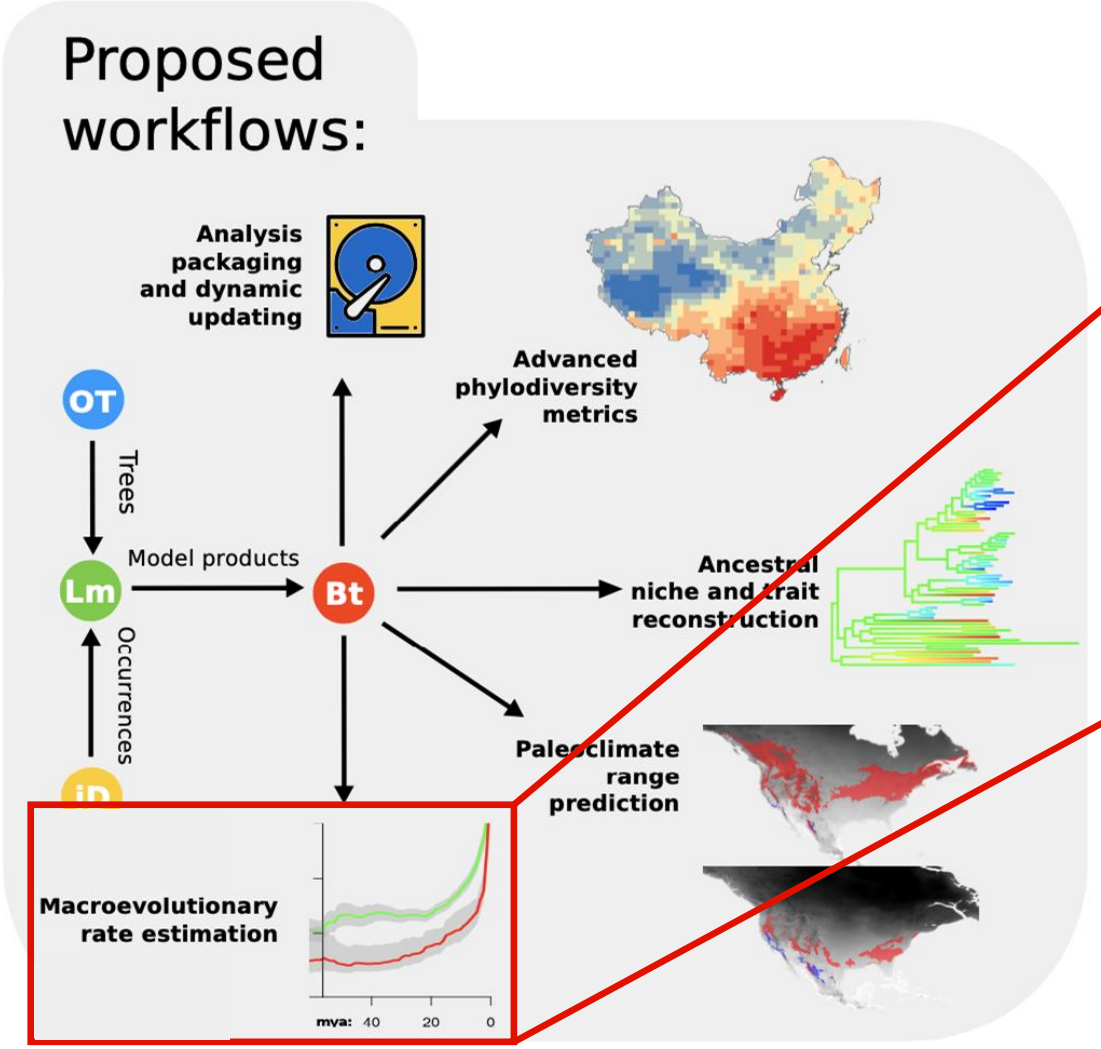
BiotaPhy Two: Proposed Workflows

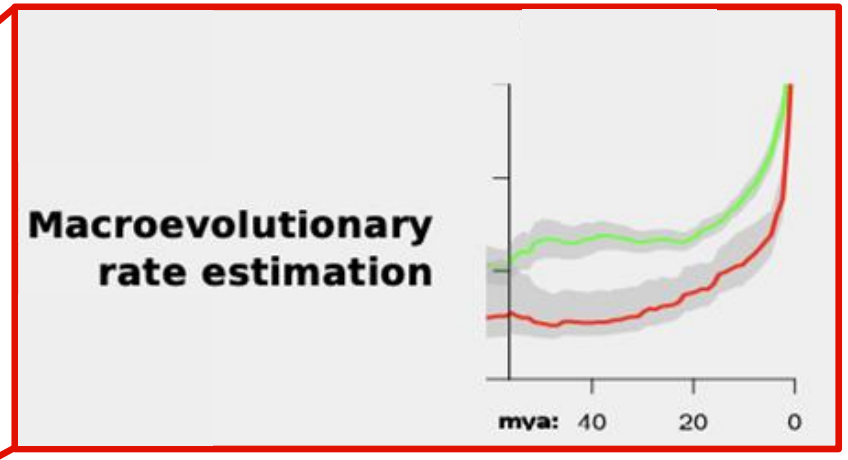




Folk et al.. 2018. Assessing ancestral niche suitability and geographic range dynamics as drivers of hybridization in *Heuchera* (Saxifragaceae). *American Naturalist* 192: 171-187.

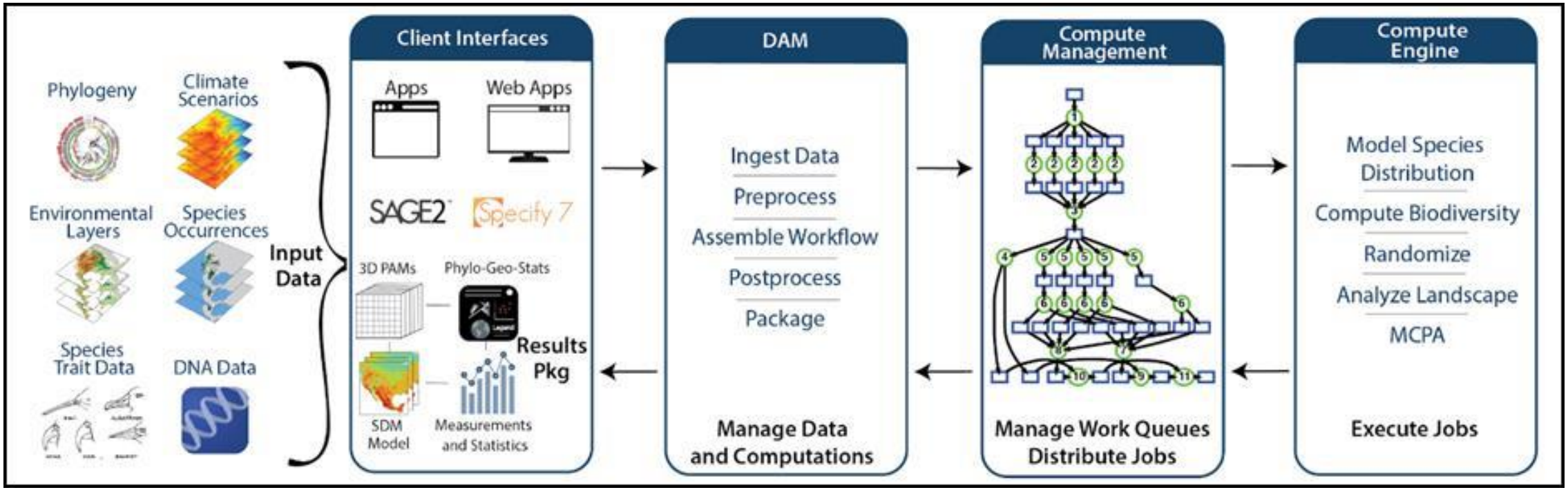
BiotaPhy Two: Proposed Workflows





Folk et al.. 2019. Uncoupled diversification, ecological niche and phenotype in a temperate radiation. *PNAS* 116: 10874-10882.

Technical Scope: BiotaPhy's architecture



Technical Scope: BiotaPhy/Lifemapper

Technology Context and Goals:

- ✓ Scaling: geographically, taxonomically, phylogenetically
- ✓ Community gateway
- ✓ High throughput networks
- ✓ Parallelization by species
- ✓ Algorithms for creating and analyzing matrices, p/a and other matrices

Technical Scope: BiotaPhy's Architecture

- ✓ Dedicated workflow environments, VisTrails and Kepler in earlier projects
- ✓ Monolithic engine with some support for APIS, Web client, CC Tools
- ✓ Modules with stronger emphasis on APIs and libraries
- ✓ Docker modules connected by simple scripts

Technical Scope: BiotaPhy's Outreach

Present and Future Work:

- ✓ Projects, recent and ongoing with the BiotaPhy biologists
- ✓ Specify Software integration
- ✓ Fomenting new collaborations

Technical Scope: BiotaPhy's Computing Approaches

Present and Future Approaches for Computing with Species Occurrence Data:

- ✓ Software tools here
- ✓ Future of big data integrative processing
- ✓ Digital Object Architecture
- ✓ Invitation for feedback and collaboration
- ✓ Full documentation at: biotaphy.github.io/tutorials

Technical Scope: Tutorial Overview

Webinar 1: Tutorial Overview

biotaphy.github.io/tutorials/tutorial/w1_overview.html

» Webinar 1: Tutorial Overview

View page source

Webinar 1: Tutorial Overview

Introduction

These tutorials step through the process of running a Biotaphy tool (or command) with example data. There are several entry points to the tools. These tutorials cover running them through a command line script, which creates a Docker container, installs the tools, executes the processes, and returns the outputs to the local machine.

Preparation

Download the github repository for this tutorial, including scripts, data, and configuration files. Go to the https://github.com/biotaphy/tutorials page, and click the green button marked “Code”. In the resulting drop-down menu, choose “Download ZIP”. When the file has downloaded, move the file (or the directory resulting from auto-decompress) to your chosen directory. If the file has not been uncompressed, do that. In Windows, right-click the zip file and choose “Extract All...”. In OSX or Linux, run unzip with the filename.

To run Biotaphy tools on any OS, locally through Docker, first download and install Docker.

Show desktop

This is what the tutorial page looks like! Access: https://biotaphy.github.io/tutorials/tutorial/w1_overview.html

Technical Scope: Tutorial Overview

Webinar 1: Tutorial Overview

Introduction

These tutorials step through the process of running a Biotaphy tool (or command) with example data. There are several entry points to the tools. These tutorials cover running them through a command line script, which creates a Docker container, installs the tools, executes the processes, and returns the outputs to the local machine.

✓ **Several entry points**

Technical Scope: Tutorial Overview

Webinar 1: Tutorial Overview

Introduction

These tutorials step through the process of running a Biotaphy tool (or command) with example data. There are several entry points to the tools. These tutorials cover running them through a command line script, which creates a Docker container, installs the tools, executes the processes, and returns the outputs to the local machine.

- ✓ **Several entry points**
- ✓ **Command line based**

```
[mariacortez@login6 mariacortez]$ ls
ancient_angiosperms            campos_rupestres.sqlite  Ocbil
campos_rupestres               favorite_plant_project   Open_Tree_9.1
campos_rupestres.33862038.out  MAFFT                    OpenTree_SSB2020
[mariacortez@login6 mariacortez]$ module load python
[mariacortez@login6 mariacortez]$ cd campos_rupestres/
[mariacortez@login6 campos_rupestres]$ ls
mata_atlantica.R  OpenTree  pathto  SQLite  SQLite_2  SQLite_Jan_2022  SQL
[mariacortez@login6 campos_rupestres]$ cd SQLite_Jan_2022/
[mariacortez@login6 SQLite_Jan_2022]$ la
-bash: la: command not found
[mariacortez@login6 SQLite_Jan_2022]$ ls
campos_rupestres_2.sqlite  campos_rupestres_4.py           campos_rupestre
campos_rupestres_3.py      campos_rupestres_4_utf.txt      campos_rupestre
campos_rupestres_3.sqlite  campos_rupestres_jan_22.sqlite  distribution_me
[mariacortez@login6 SQLite_Jan_2022]$ cd campos_rupestres_SQLite/
[mariacortez@login6 campos_rupestres_SQLite]$ ls
```

Technical Scope: Tutorial Overview

Webinar 1: Tutorial Overview

Introduction

These tutorials step through the process of running a Biotaphy tool (or command) with example data. There are several entry points to the tools. These tutorials cover running them through a command line script, which creates a Docker container, installs the tools, executes the processes, and returns the outputs to the local machine.

- ✓ **Several entry points**
- ✓ **Command line based**
- ✓ **Uses Docker**

Technical Scope: Tutorial Overview

Webinar 1: Tutorial Overview

Introduction

These tutorials step through the process of running a Biotaphy tool (or command) with example data. There are several entry points to the tools. These tutorials cover running them through a command line script, which creates a Docker container, installs the tools, executes the processes, and returns the outputs to the local machine.

- ✓ **Several entry points**
- ✓ **Command line based**
- ✓ **Uses Docker**
- ✓ **Outputs Saved in local machine**

main tutorials / data / easy_bake / heuchera_accepted1.txt

zzeppozz clarity ✓

1 contributor

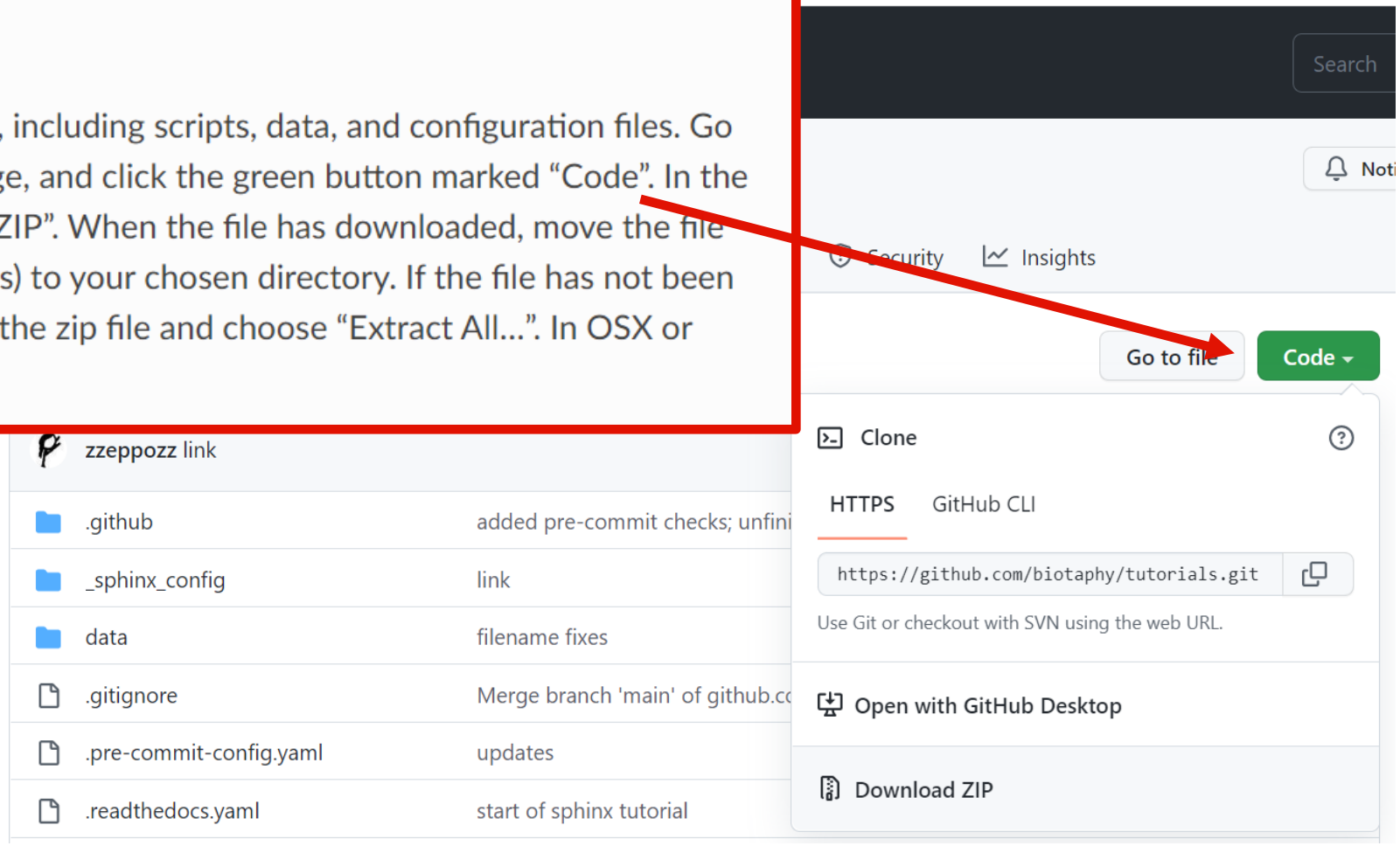
124 lines (124 sloc) | 2.32 KB

```
Heuchera puberula
Heuchera scapigera
Heuchera lucida
Heuchera hirtiflora
Heuchera barbarossa
Heuchera richardsonii
Heuchera hirsuticaulis
Heuchera wootonii
Heuchera foliosa
Heuchera duranii
```

Technical Scope: Tutorial Overview

Preparation

Download the github repository for this tutorial, including scripts, data, and configuration files. Go to the https://github.com/biotaphy/tutorials page, and click the green button marked “Code”. In the resulting drop-down menu, choose “Download ZIP”. When the file has downloaded, move the file (or the directory resulting from auto-decompress) to your chosen directory. If the file has not been uncompressed, do that. In Windows, right-click the zip file and choose “Extract All...”. In OSX or Linux, run unzip with the filename.

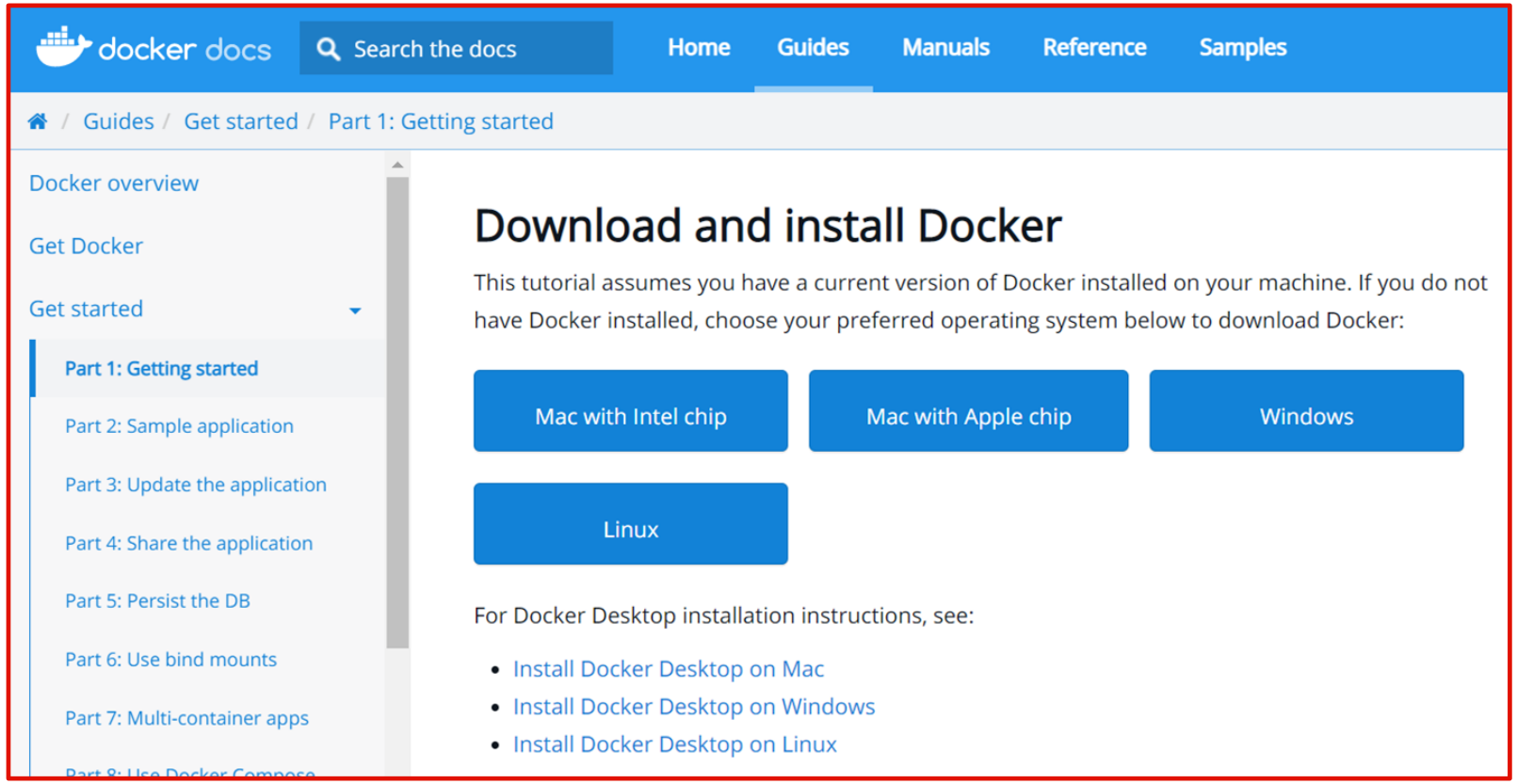


Technical Scope: Tutorial Overview

To run Biotaphy tools on any OS, locally through Docker, first download and install Docker.

Docker works similarly to a virtual machine, but more efficiently!!!!

Google 'Docker', access the website and download the appropriate version!



Technical Scope: Tutorial Overview

Host and Container input, output, log directories

A data Docker Volume, is created on the Host machine, and the tutorials/data/input, tutorials/data/config, and tutorials/data/wranglers directories in this repository are copied to it. These directories are then made available on the Docker Container under the /volumes/data volume (directory). If modified, the docker "data" volume must be re-created to propagate those changes to the containers.

Another Docker volume, *output*, is created on the Host machine and mounted at /volumes/output in the Docker Container. Changes in this directory are saved in the volume, and copied back to the host machine, under the data directory.

Host machine

2 different volumes

data

output

File system

Technical Scope: Tutorial Overview

Host and Container input, output, log directories

A data Docker Volume is created on the Host machine, and the tutorials/data/input, tutorials/data/config, and tutorials/data/wranglers directories in this repository are copied to it. These directories are then made available on the Docker Container under the /volumes/data volume (directory). If modified, the docker "data" volume must be re-created to propagate those changes to the containers.

Another Docker volume, *output*, is created on the Host machine and mounted at /volumes/output in the Docker Container. Changes in this directory are saved in the volume, and copied back to the host machine, under the data directory.

Host machine

2 different volumes

- data
  - input
  - configurations
  - wranglers
- output

File system

Technical Scope: Tutorial Overview

Host and Container input, output, log directories

A data Docker Volume, is created on the Host machine, and the tutorials/data/input, tutorials/data/config, and tutorials/data/wranglers directories in this repository are copied to it. These directories are then made available on the Docker Container under the /volumes/data volume (directory). If modified, the docker "data" volume must be re-created to propagate those changes to the containers.

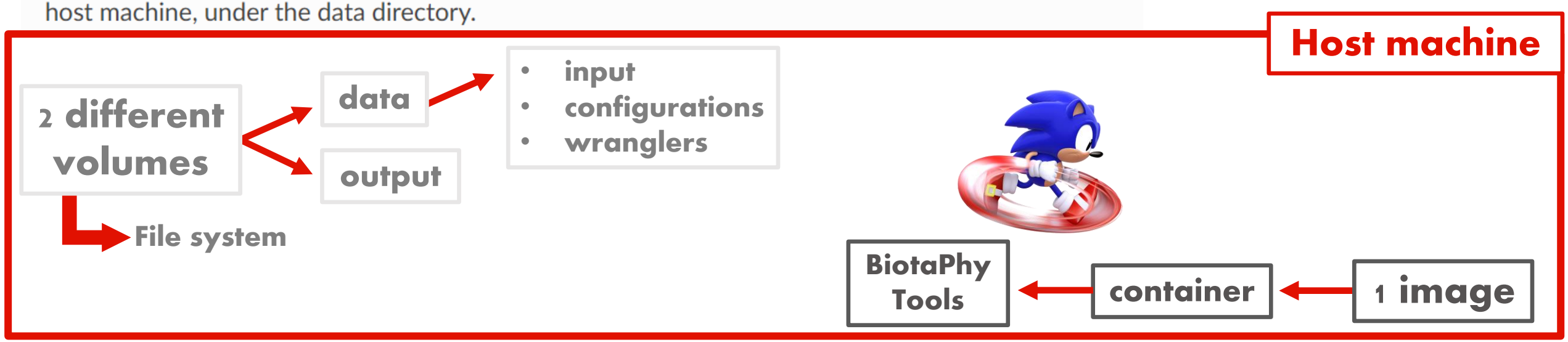
Another Docker volume, *output*, is created on the Host machine and mounted at /volumes/output in the Docker Container. Changes in this directory are saved in the volume, and copied back to the host machine, under the data directory.

Technical Scope: Tutorial Overview

Host and Container input, output, log directories

A data Docker Volume, is created on the Host machine, and the tutorials/data/input, tutorials/data/config, and tutorials/data/wranglers directories in this repository are copied to it. These directories are then made available on the Docker Container under the /volumes/data volume (directory). If modified, the docker "data" volume must be re-created to propagate those changes to the containers.

Another Docker volume, *output*, is created on the Host machine and mounted at /volumes/output in the Docker Container. Changes in this directory are saved in the volume, and copied back to the host machine, under the data directory.



Technical Scope: Tutorial Overview

Host and Container input, output, log directories

A data Docker Volume, is created on the Host machine, and the tutorials/data/input, tutorials/data/config, and tutorials/data/wranglers directories in this repository are copied to it. These directories are then made available on the Docker Container under the /volumes/data volume (directory). If modified, the docker "data" volume must be re-created to propagate those changes to the containers.

Another Docker volume, *output*, is created on the Host machine and mounted at /volumes/output in the Docker Container. Changes in this directory are saved in the volume, and copied back to the host machine, under the data directory.

Host machine

2 different volumes

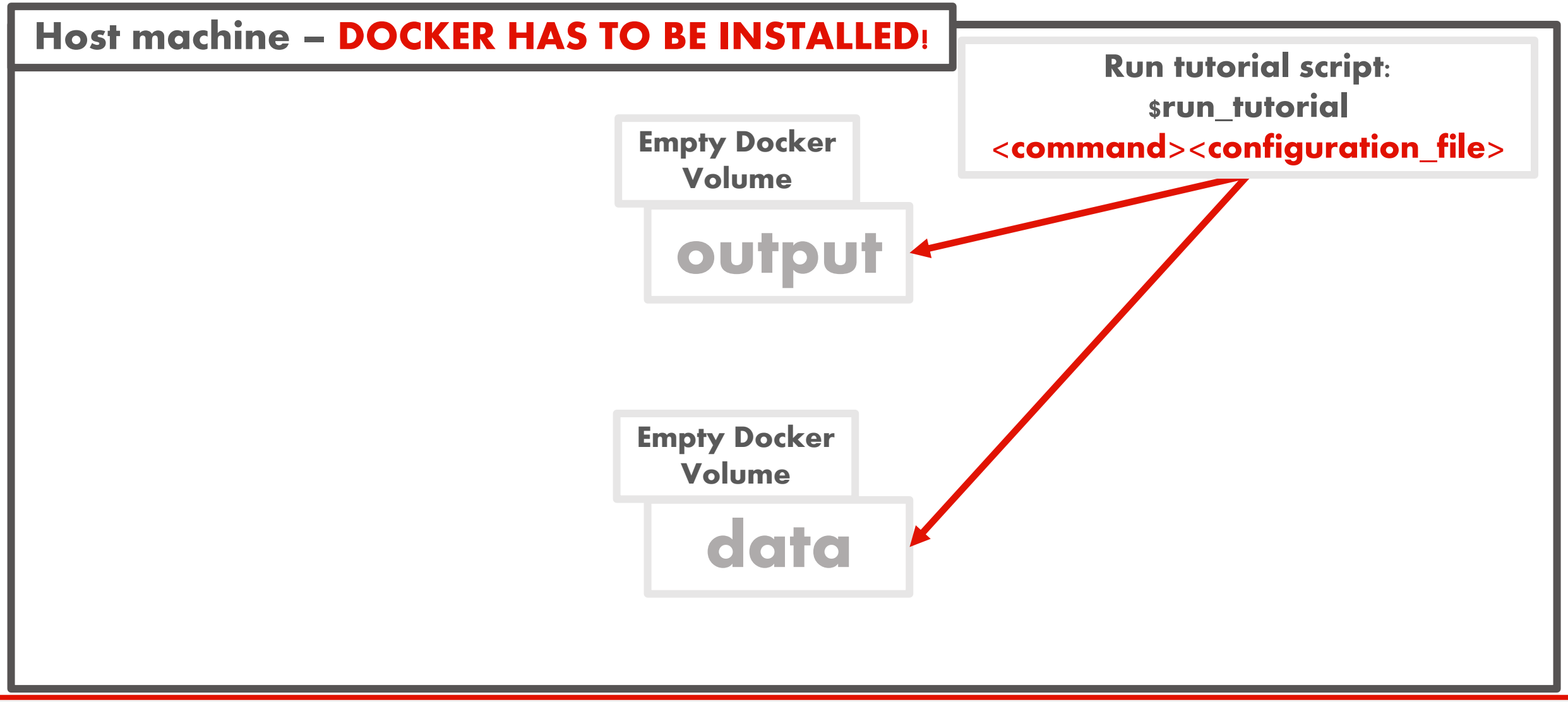
- data
  - input
  - configurations
  - wranglers
- output

File system

- ✓ Static
- ✓ Doesn't run by itself
- ✓ Contains scripts

1 image → container → BiotaPhy Tools

Technical Scope: Tutorial Overview



Technical Scope: Tutorial Overview

Host machine – **DOCKER HAS TO BE INSTALLED!**

Run tutorial script:
$run_tutorial
<command><configuration_file>

Empty Docker Volume

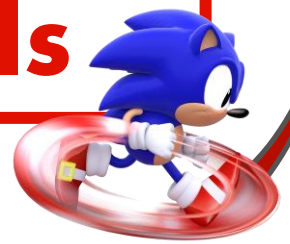
output

Empty Docker Volume

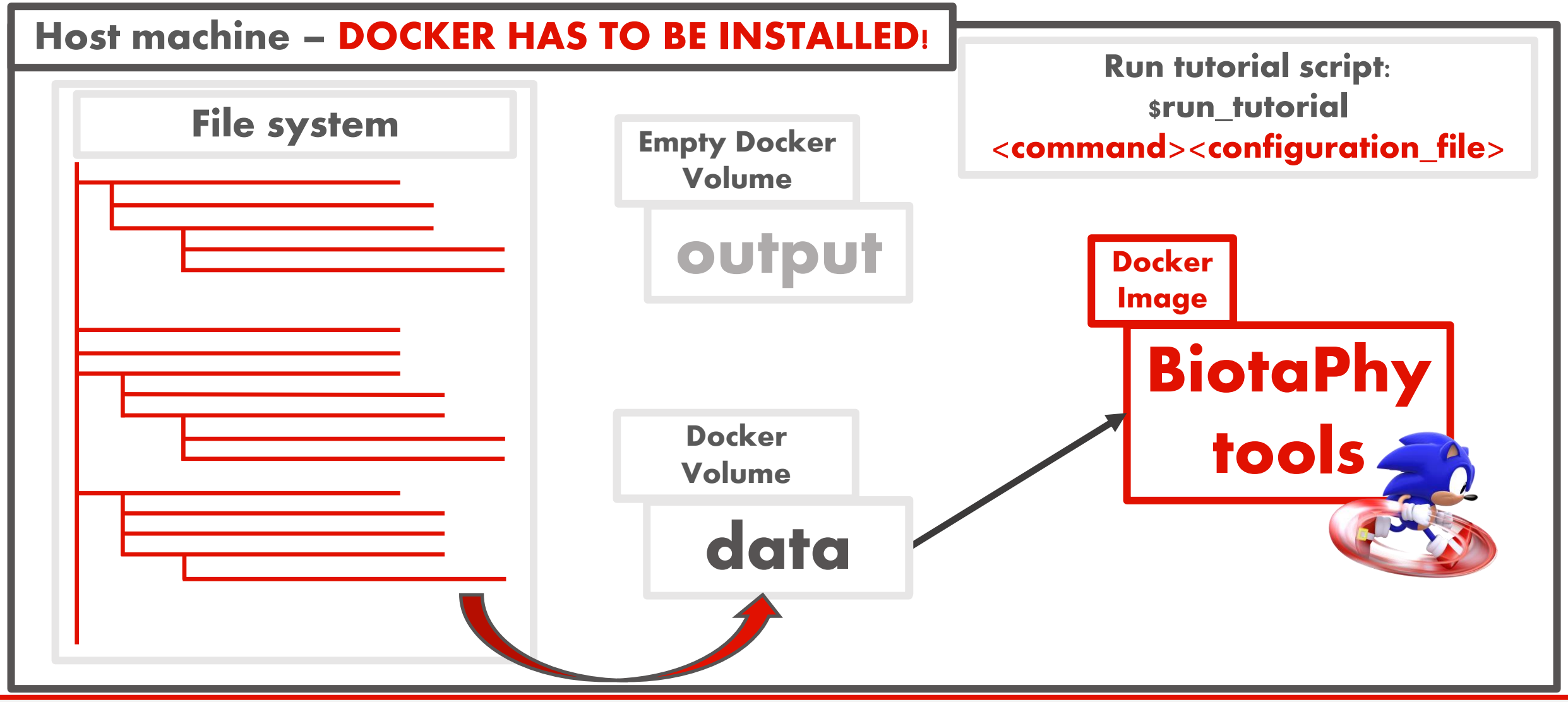
data

Docker Image

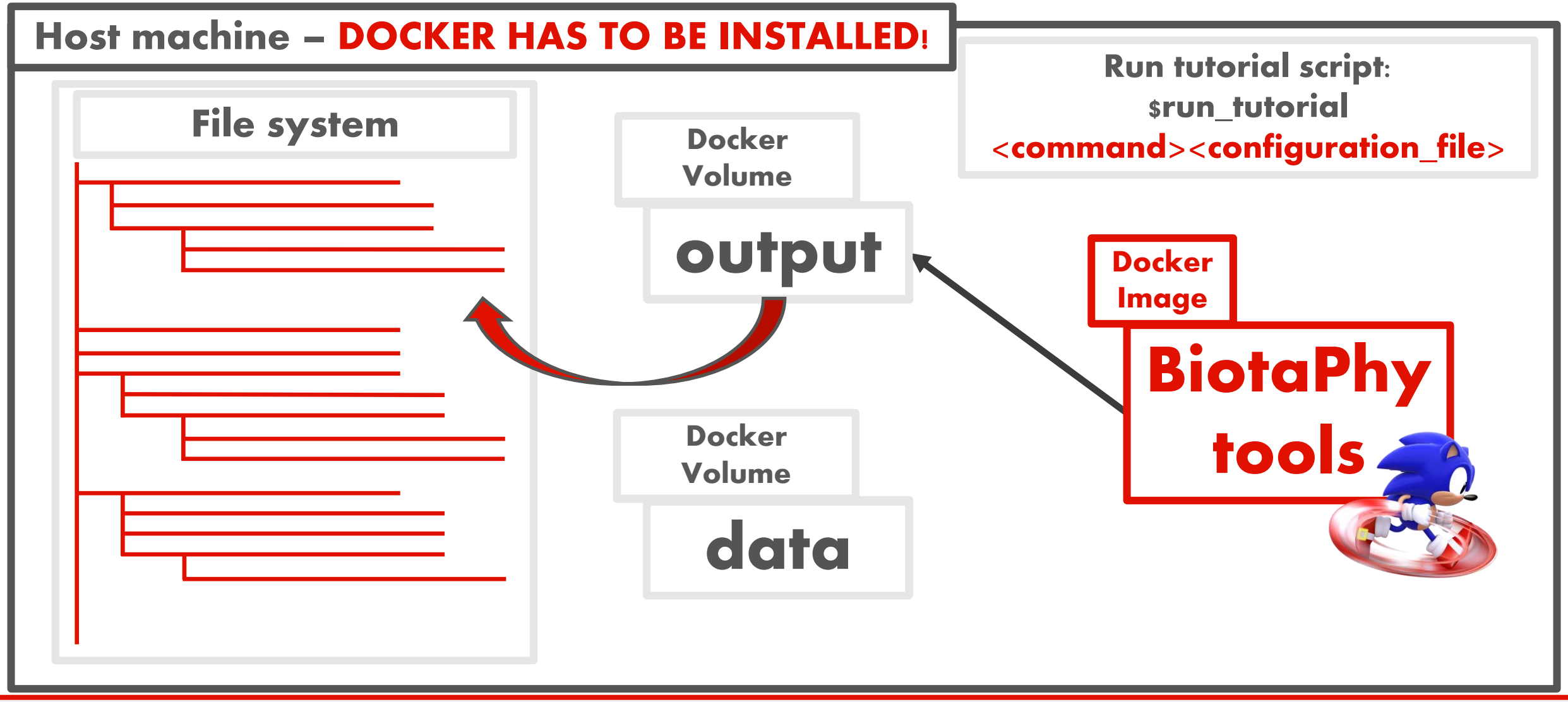
BiotaPhy tools



Technical Scope: Tutorial Overview



Technical Scope: Tutorial Overview



Technical Scope: Tutorial Overview

Data preparation: Script parameters File

All commands require a parameters file with tool-specific parameters. The file must be in JSON format. More information is [here](script_params.rst).

Each tutorial contains one or more example parameters files in the tutorials/data/config directory. These parameters files reference example data and parameters reasonable for that command. All required and optional parameters are described in individual tutorial pages.

Some tools will require an additional JSON format configuration file. In these cases, the additional JSON file will be named in the parameters file.

- ✓ **Parameter files are essential to properly run the commands!**
- ✓ **Stored in the data/config directory**

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes ("), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and
5. Output filenames must be a full path to the fil

```
{
    "max_open_writers": 100,
    "key_field_ignore": ["genus","scientficName"],
    "dwca":  [
        ["/volumes/data/input/occ_heuchera_gbif.zip",
         "/volumes/data/wranglers/no_wranglers.json"
        ]
    ],
    "out_dir": "/volumes/output/heuchera_dwca"
}
```

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes ("), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and
5. Output filenames must be a full path to the fil

```
{
    "max_open_writers": 100,
    "key_field_ignore": ["genus","scientficName"],
    "dwca":  [
        ["/volumes/data/input/occ_heuchera_gbif.zip",
         "/volumes/data/wranglers/no_wranglers.json"
        ]
    ],
    "out_dir": "/volumes/output/heuchera_dwca"
}
```

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes ("), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and
5. Output filenames must be a full path to the fil

```
{
    "max_open_writers": 100,
    "key_field_ignore": ["genus","scientficName"],
    "dwca":  [
        ["/volumes/data/input/occ_heuchera_gbif.zip",
         "/volumes/data/wranglers/no_wranglers.json"
        ]
    ],
    "out_dir": "/volumes/output/heuchera_dwca"
}
```

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes ("), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and
5. Output filenames must be a full path to the fil

```
{
    "max_open_writers": 100,
    "key_field_ignore": ["genus","scientficName"],
    "dwca":  [
        ["/volumes/data/input/occ_heuchera_gbif.zip",
         "/volumes/data/wranglers/no_wranglers.json"
        ]
    ],
    "out_dir": "/volumes/output/heuchera_dwca"
}
```

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes (“), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and enclosed in square brackets ([]).
5. Output filenames must be a full path to the fil

```
    "max_open_writers": 100,
    "key_field_ignore": ["genus","scientficName"],
    "dwca":  [
        ["/volumes/data/input/occ_heuchera_gbif.zip",
         "/volumes/data/wranglers/no_wranglers.json"
        ]
    ],
    "out_dir": "/volumes/output/heuchera_dwca"
}
```

Technical Scope: Tutorial Overview

Script parameters file

1. Each command has required and/or optional parameters.
2. The group of options are enclosed in curly braces ({}).
3. Each option keyword is quoted with double quotes (“), is followed by a colon (:) and a value.
4. Each value may be a double-quoted string, a number, a boolean (true or false, not quoted), or a list of these items, separated by commas and enclosed in square brackets ([]).
5. Output filenames must be a full path to the file (or directory containing files)

```
        ],
        "out_dir": "/volumes/output/heuchera_dwca"
    }
```

Technical Scope: Tutorial Overview

- log_console: this is a boolean value. The value *true*, causes the command to print logging lines in the command line window, to show the processes and progress.
- log_filename: a full path to the output log file to be created. The file is an *Output filename* as described above. It contains all logging output from the process, and may be useful for identifying what processes were executed and their outcomes.
- report_filename: a full path to the report log file to be created. The file is an *Output filename* as described above. It contains a summary of the modifications made to the output data, and may be useful for quantifying or comparing them.

8 lines (8 sloc) | 386 Bytes

```
1  {
2    "log_filename": "/volumes/output/wrangle_species_list1.log",
3    "log_console": true,
4    "report_filename": "/volumes/output/wrangle_species_list1_rpt.json",
5    "in_species_list_filename": "/volumes/data/input/heuchera.txt",
6    "wrangler_configuration_file": "/volumes/data/wranglers/splist_wranglers_gbif.json",
7    "out_species_list_filename": "/volumes/output/heuchera_accepted1.txt"
8  }
```

Continuing with Script parameters file! Optional parameters

Technical Scope: Tutorial Overview

- log_console: this is a boolean value. The value *true*, causes the command to print logging lines in the command line window, to show the processes and progress.
- log_filename: a full path to the output log file to be created. The file is an *Output filename* as described above. It contains all logging output from the process, and may be useful for identifying what processes were executed and their outcomes.
- report_filename: a full path to the report log file to be created. The file is an *Output filename* as described above. It contains a summary of the modifications made to the output data, and may be useful for quantifying or comparing them.

Continuing with Script parameters file! Optional parameters

8 lines (8 sloc) | 386 bytes

```
{
  "log_filename": "/volumes/output/wrangle_species_list1.log",
  "log_console": true,
  "report_filename": "/volumes/output/wrangle_species_list1_rpt.json",
  "in_species_list_filename": "/volumes/data/input/heuchera.txt",
  "wrangler_configuration_file": "/volumes/data/wranglers/splist_wranglers_gbif.json",
  "out_species_list_filename": "/volumes/output/heuchera_accepted1.txt"
}
```

Technical Scope: Tutorial Overview

- log_console: this is a boolean value. The value *true*, causes the command to print logging lines in the command line window, to show the processes and progress.
- log_filename: a full path to the output log file to be created. The file is an *Output filename* as described above. It contains all logging output from the process, and may be useful for identifying what processes were executed and their outcomes.
- report_filename: a full path to the report log file to be created. The file is an *Output filename* as described above. It contains a summary of the modifications made to the output data, and may be useful for quantifying or comparing them.

Continuing with Script parameters file! Optional parameters

8 lines (8 sloc) | 386 Bytes

```
{
  "log_filename": "/volumes/output/wrangle_species_list1.log",
  "log_console": true,
  "report_filename": "/volumes/output/wrangle_species_list1_rpt.json",
  "in_species_list_filename": "/volumes/data/input/heuchera.txt",
  "wrangler_configuration_file": "/volumes/data/wranglers/splist_wranglers_gbif.json",
  "out_species_list_filename": "/volumes/output/heuchera_accepted1.txt"
}
```

Technical Scope: Tutorial Overview

- log_console: this is a boolean value. The value *true*, causes the command to print logging lines in the command line window, to show the processes and progress.
- log_filename: a full path to the output log file to be created. The file is an *Output filename* as described above. It contains all logging output from the process, and may be useful for identifying what processes were executed and their outcomes.
- report_filename: a full path to the report log file to be created. The file is an *Output filename* as described above. It contains a summary of the modifications made to the output data, and may be useful for quantifying or comparing them.

We will learn more about wranglers later in the series!

8 lines (8 sloc) | 386 Bytes

```
{
  "log_filename": "/volumes/output/wrangle_species_list1.log",
  "log_console": true,
  "report_filename": "/volumes/output/wrangle_species_list1_rpt.json",
  "in_species_list_filename": "/volumes/data/input/heuchera.txt",
  "wrangler_configuration_file": "/volumes/data/wranglers/splist_wranglers_gbif.json",
  "out_species_list_filename": "/volumes/output/heuchera_accepted1.txt"
}
```

Technical Scope: Tutorial Overview

Run tool tutorials

The "run_tutorial" script will run each tutorial with two arguments, the 1) command name and 2) parameters file. The parameters file will be a path on the local machine, in the tutorials/data/config directory. The script will translate that to the container path, and execute the command in the container with the container's copy of the file. For example, the wrangle_species_list tutorial can be initiated with:

```
./run_tutorial.sh wrangle_species_list ./data/config/wrangle_species_list_gbif.json
```

Command name

Technical Scope: Tutorial Overview

Run tool tutorials

The "run_tutorial" script will run each tutorial with two arguments, the 1) command name and 2) parameters file. The parameters file will be a path on the local machine, in the tutorials/data/config directory. The script will translate that to the container path, and execute the command in the container with the container's copy of the file. For example, the wrangle_species_list tutorial can be initiated with:

```
./run_tutorial.sh wrangle_species_list ./data/config/wrangle_species_list_gbif.json
```

Parameters file

Technical Scope: Tutorial Overview

Run tool tutorials

The “run_tutorial” script will run each tutorial with two arguments, the 1) command name and 2) parameters file. The parameters file will be a path on the local machine, in the tutorials/data/config directory. The script will translate that to the container path, and execute the command in the container with the container’s copy of the file. For example, the wrangle_species_list tutorial can be initiated with:

```
./run_tutorial.sh wrangle_species_list ./data/config/wrangle_species_list_gbif.json
```

Command name

Parameters file

Technical Scope: Tutorial Overview

Run tool tutorials

The “run_tutorial” script will run each tutorial with two arguments, the 1) command name and 2) parameters file. The parameters file will be a path on the local machine, in the tutorials/data/config directory. The script will translate that to the container path, and execute the command in the container with the container’s copy of the file. For example, the wrangle_species_list tutorial can be initiated with:

```
./run_tutorial.sh wrangle_species_list ./data/config/wrangle_species_list_gbif.json
```

Outputs

All outputs are specified in the Tool Configuration File provided to the command, and will be copied to the data/outputs directory on completion.

Technical Scope: Tutorial Overview

Behind the scenes

The "run_tutorial" script will execute the following functions, unless their outputs have already been created:

1. Create several Docker Volumes to share data between the host and Docker container.
2. Build a Docker image.
3. Start a Docker Container from the image, with volumes attached. A Container is similar to a fully functioning computer with data and applications.
4. Execute the specified command with the parameters in the specified configuration file. The process will execute using a parameter file and data in the *data* data volume and write outputs to the output volume, executing code in the Docker container.
5. Copy the container **/volumes/output** directory back to the local data directory.
6. Stop and delete the container. All outputs in the docker volume are preserved and accessible the next time it is attached to a container.

Session Summary, Q&A and Discussion

- Advances in biodiversity science, combined with emerging technologies and the ability to handle "big data" have greatly improved and expanded our capability to explore biodiversity in an unprecedented fashion. We can now link data from growing repositories (trees, occurrences, traits) and computational tools/approaches to integrate evolution and ecology at broad scale. This new synthesis is reshaping views of ecology/evolution with important conservation implications.

- Scaling biogeographic analyses from a small number of species or genera to explore patterns of evolution and diversity for thousands or tens of thousands of taxa on continental and global scales requires software tools that automate and parallelize computational tasks to make analyses practically feasible, efficient, with reasonable timelines.

Allen et al.. 2019. Biodiversity synthesis across the green branches of the tree of life. *Nature Plants* 5:11-13.

Session Summary, Q&A and Discussion

Any questions??

Please use the link to the Jamboard to write your question!