Title

From microbial community structure to metabolic inference using paprica.

Permalink

https://escholarship.org/uc/item/5ct4x8kw

Journal

STAR Protocols, 2(4)

Authors

Erazo, Natalia

Dutta, Avishek

Bowman, Jeff

Publication Date

2021-12-17

DOI

10.1016/j.xpro.2021.101005

Peer reviewed

STAR Protocols

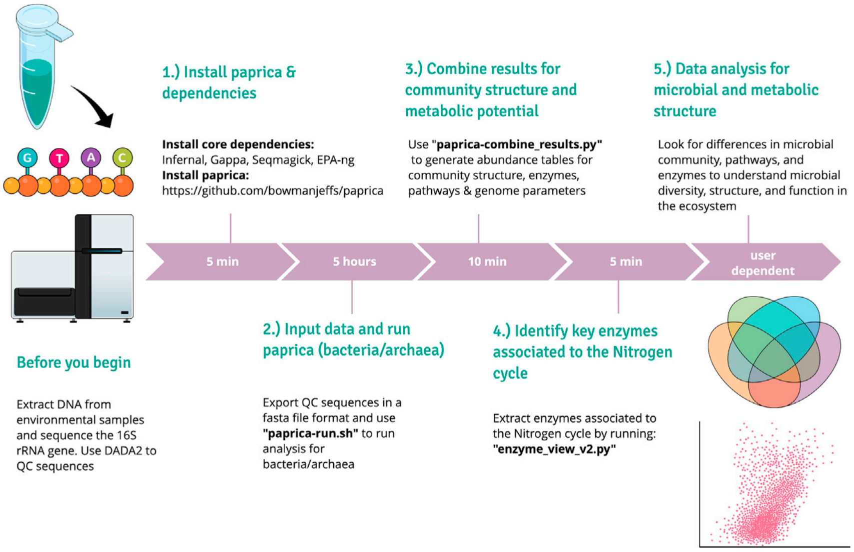
CellPress
OPEN ACCESS

Protocol

From microbial community structure to metabolic inference using paprica

Natalia G. Erazo, Avishek Dutta, Jeff S. Bowman

nerazo@ucsd.edu



Before you begin

Extract DNA from environmental samples and sequence the 16S rRNA gene. Use DADA2 to QC sequences

1.) Install paprica & dependencies

Install core dependencies:
Infernal, Gappa, Seqmagick, EPA-ng
Install paprica:
https://github.com/bowmanjeffs/paprica

2.) Input data and run paprica (bacteria/archaea)

Export QC sequences in a fasta file format and use **"paprica-run.sh"** to run analysis for bacteria/archaea

3.) Combine results for community structure and metabolic potential

Use **"paprica-combine_results.py"** to generate abundance tables for community structure, enzymes, pathways & genome parameters

4.) Identify key enzymes associated to the Nitrogen cycle

Extract enzymes associated to the Nitrogen cycle by running: **"enzyme_view_v2.py"**

5.) Data analysis for microbial and metabolic structure

Look for differences in microbial community, pathways, and enzymes to understand microbial diversity, structure, and function in the ecosystem

Microbial taxonomic marker gene studies using 16S rRNA gene amplicon sequencing provide an understanding of microbial community structure and diversity; however, it can be difficult to infer the functionality of microbes in the ecosystem from these data. Here, we show how to predict metabolism from phylogeny using the paprica pipeline. This approach allows resolution at the strain and species level for select regions on the prokaryotic phylogenetic tree and provides an estimate of gene and metabolic pathway abundance.

Highlights

Paprica determines microbial community structure and infers metabolic structure

Paprica uses a phylogenetic placement approach

Paprica generates an estimate of genome sizes, gene content, and metabolic pathways

This pipeline helps to understand the function of bacteria in complex ecosystems

Erazo et al., STAR Protocols 2, 101005
December 17, 2021 © 2021 The Authors.
https://doi.org/10.1016/j.xpro.2021.101005

STAR Protocols

CellPress
OPEN ACCESS

Protocol

From microbial community structure to metabolic inference using paprica

Natalia G. Erazo,[1,2,4,5,*] Avishek Dutta,[1] and Jeff S. Bowman[1,2,3,5]

[1]Scripps Institution of Oceanography, UC San Diego, La Jolla, CA, USA

[2]Center for Marine Biodiversity and Conservation, UC San Diego, La Jolla, CA, USA

[3]Center for Microbiome Innovation, UC San Diego, La Jolla, CA, USA

[4]Technical contact

[5]Lead contact

*Correspondence: nerazo@ucsd.edu

https://doi.org/10.1016/j.xpro.2021.101005

SUMMARY

Microbial taxonomic marker gene studies using 16S rRNA gene amplicon sequencing provide an understanding of microbial community structure and diversity; however, it can be difficult to infer the functionality of microbes in the ecosystem from these data. Here, we show how to predict metabolism from phylogeny using the paprica pipeline. This approach allows resolution at the strain and species level for select regions on the prokaryotic phylogenetic tree and provides an estimate of gene and metabolic pathway abundance.

For complete details on the use and execution of this protocol, please refer to Erazo and Bowman (2021).

BEFORE YOU BEGIN

Data collection

This protocol describes how to use the paprica pipeline to determine microbial community structure and predict metabolic pathways by phylogenetic placement approach (Bowman and Ducklow, 2015). This method differs from traditional OTU-based methods, such as UCLUST (Rideout et al., 2014), MOTHUR (Schloss et al., 2009), and USEARCH-UPARSE (Edgar, 2010), in that it relies on the placement of reads on a phylogenetic tree created from the 16S + 23S rRNA gene reads from all completed bacterial and archaeal genomes in the NCBI RefSeq database (Haft et al., 2018). Because the metabolic potential of each phylogenetic edge on the reference tree is known, paprica generates a reasonable estimate of genome sizes, gene content, and metabolic pathways for the genome of origin of each read. In Bowman and Ducklow (2015), paprica was used to infer microbial metabolism and results were comparable to other common metabolic inference pipelines such as PICRUSt (Langille et al., 2013). Paprica has been used to understand the temporal dynamics of bacterial and archaeal community structure in coastal ecosystems (Wilson et al., 2021), the microbial diversity of hypersaline lakes (Klempay et al., 2021), and the microbial dynamics in up-flow bioreactors (Dutta et al., 2020). Paprica offers a framework to bridge the gap between taxonomy and marker gene studies, which are economical but indirectly linked to community function, and potential metabolism, which is costly and can be labor intensive to analyze but is directly linked to function (Bowman and Ducklow, 2015).

The input to the paprica pipeline is 16S rRNA amplicon sequence reads. It is important that the sequences go through standard quality control and denoising procedures, as described below, and that the final format input to the pipeline is a fasta file. Reads are tallied across multiple samples at the level of unique reads (i.e., amplicon sequence variants or ASVs) providing maximum possible resolution of metabolic and community structure.

Key terminology in paprica:

Edge: An edge is a point of placement on a reference tree (e.g., branch of the reference tree).

Unique read: a read from a dataset that has been denoised using DADA2.

Closest completed genome (CCG) (edge type I): The most closely taxonomically related completed genome to a query read. This term applies to reads that place a terminal edge (branch tip) on the reference tree.

Closest estimated genome (CEG) (edge type II): The set of genes that are estimated to be found in all members of a clade originating at a branch point in the reference tree. This term applies to reads that place an internal edge on the reference tree.

Reference tree: The tree of representative 16S rRNA gene sequences from all completed bacterial genomes in NCBI RefSeq database. Here the topology of the tree defines what pathways are predicted for internal branch points.

DNA extraction

◷ Timing: DNA extraction: 24–96 samples/day

1. We extracted DNA from our environmental samples using the DNAeasy PowerWater DNA extraction kit (Qiagen).
2. Extracted DNA was quantified using the Qubit HS DNA quantification kit (Invitrogen) and then quality checked by gel electrophoresis and PCR amplification of the 16S rRNA gene using primers 515F and 806R (Walters et al., 2016) for bacteria and archaea.
3. High quality extracted DNA was submitted to the Argonne National Laboratory sequencing center for amplification and library preparation with the same primer set, followed by 2 × 151 paired-end sequenced on the Illumina MiSeq platform.

Sequence analysis

◷ Timing: 1–2 h

4. Illumina MiSeq reads were demultiplexed using the ''iu-demultiplex'' command in Illumina utils (Eren et al., 2013).
5. Demultiplexed reads were quality controlled and denoised using the ''FilterandTrim'' and ''dada'' commands within the R package DADA2 (Callahan et al., 2016), and assembled with the ''mergePairs'' command. The final merged reads had mean quality scores >30.
6. The following script can be used to QC sequences: ''dada2.r'' and it can be found here: https://github.com/bowmanjeffs/shared_scripts/blob/master/dada2.r
7. To export the data into fasta file format you can use ''deunique_dada2.py'' script and it can be found here: https://github.com/bowmanjeffs/shared_scripts/blob/master/deunique_dada2.py

⚠ CRITICAL: It is very important to QC sequences before running paprica. Here we used the DADA2 pipeline following standard filtering parameters. We recommend checking whether the default parameters work for your sequences. You can find more information and tutorials on the DADA2 pipeline here: https://benjjneb.github.io/dada2/index.html

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and algorithms		
DADA2	Callahan et al. (2016)	https://github.com/benjjneb/dada2
Paprica	Bowman and Ducklow (2015)	https://github.com/bowmanjeffs/paprica
EPA-ng	Barbera et al. (2019)	https://github.com/Pbdas/epa-ng
Infernal	Nawrocki and Eddy (2013)	https://github.com/EddyRivasLab/infernal
Gappa	Czech et al. (2020)	https://github.com/lczech/gappa
Seqmagick	Matsen Group	https://fhcrc.github.io/seqmagick/

MATERIALS AND EQUIPMENT

- Data (fasta file 16s rRNA sequences – see data collection in before you begin)
- paprica installation and required dependencies:

Python (v.3.6 or higher)

Python modules: pandas, Biopython, numpy, termcolor

Infernal

Gappa

Seqmagick

EPA-ng

For installation of the required dependencies, follow these instructions for a Linux operating system:

https://github.com/bowmanjeffs/paprica/blob/master/linux_install.sh

or for a Mac OSX system:

https://www.polarmicrobes.org/installing-paprica-on-mac-osx/

You can also get paprica with the required dependencies using Docker:

https://hub.docker.com/r/jsbowman/paprica (see below for installation).

> *Note:* paprica requires a bash equipped, Linux-like operating environment with at least 8 Gb RAM.

STEP-BY-STEP METHOD DETAILS

Step 1: Installing paprica

> Timing: 5 min

Users familiar with Docker containers can use the paprica Docker image at https://hub.docker.com/repository/docker/jsbowman/paprica instead of installing the dependencies:

```
> docker pull jsbowman/paprica:latest

> docker run -it jsbowman/paprica

> cd /paprica
```

If you do not use Docker and install the required dependencies, different issues could arise (Troubleshooting 1 and 2).

The full installation of paprica includes downloading the paprica repository from GitHub. An example of how to run paprica using real data is available at: https://www.polarmicrobes.org/analysis-with-paprica/

1. Install paprica and activate scripts by running the following code:

```
> git clonehttps://github.com/bowmanjeffs/paprica.git

> cd paprica

> chmod a+x *py

> chmod a+x *sh
```

2. You can test paprica by:

```
>./paprica-run.sh test bacteria
```

Note: If you want to create an environmental variable to run paprica in docker, you can learn more about it here: https://docs.docker.com/compose/environment-variables/. It is not required to create an environmental variable to run paprica, if the paprica dependencies are up to date.

Step 2: Preparing input for paprica

Timing: 3 min

3. Once the data is QC'd and in a fasta file format, we need to construct a loop to run paprica on multiple fasta files. Because many of the programs executed by paprica are intrinsically parallelized we do not recommend executing multiple iterations of paprica in parallel. Doing so will result in only a small increase in efficiency. To execute our loop, we need to create a text file with a list of the samples to run. This file contains the name of samples without the file extension (.fasta). If you execute the "deunique_dada2.py" script, all of your fasta files will end with .exp.fasta. You can create the file of sample names by running this bash loop in the same directory as your samples:

```
> for f in *.exp.fasta; do printf ’%s\n’ "${f%.fasta}" >> samples.txt;

done
```

Step 3: Run paprica for bacteria

Timing: ~ 5 h for 120 samples for Bacteria and Archaea (CPU cores: 18)

The first time you run paprica on a 16S rRNA gene library it will determine which reads belong to the domains Bacteria and Archaea. There are separate reference trees and databases for these domains so they should be handled separately, but because these reads are derived from the same library they can be recombined in post-processing. For a multi-sample analysis, you will need to execute two scripts:

"paprica-run.sh": This is the core analysis in paprica that will place reads on a 16S + 23S rRNA gene tree compromised of all completed genomes in Refseq. It will map the enzymes, pathways, and genome parameters at each point of placement to the query reads, and it will use the points of placement to provide a consensus taxonomy for each query read. To do this paprica-run.sh will internally execute three scripts:

"paprica-pick_domain.py": It identifies the domain of your input reads (archaea, bacteria).

"paprica-place_it.py": It does a phylogenetic placement of query reads.

"paprica-tally_pathways.py": It finds pathways and other information associated with each point of placement.

> *Note:* If you want to run paprica with default settings you don't need to do any changes here (see limitations for further information). If you encounter issues, see Troubleshooting 3 and 4.

"paprica-combine_results.py": This will create abundance tables for edges, unique reads, enzymes, and pathways, and a table of predicted genome parameters.

4. Copy the "paprica-run.sh" script into the current directory where we have the sample files. You can run this code to run the analysis for Bacteria (Troubleshooting 5):

```
> cp ~/paprica/paprica-run.sh paprica-run.sh

> while read f; do ./paprica-run.sh $f bacteria; done < samples.txt
```

> *Note:* At this point we have individual analysis files for all the samples. We want to combine these results into abundance tables for edges, unique reads, enzymes, and pathways. Here we use another core script of paprica to generate these tables.

> ⏸ **Pause point:** This is a good pausing point. Depending on the CPU cores of your computer, this analysis should run for ~2.5 h.

5. The "paprica-combine_results.py" script can now be used to combine your files into tables of abundance by running this:

```
> paprica-combine_results.py -domain bacteria -o 2021_mangrove_study
```

> *Note:* The prefix that you'd like to give to the output files is set by the -o flag in the command above.

6. After running the script, this will produce abundance tables for community structure and predicted metabolic pathways, enzymes, and genome parameters.

Step 4: Run paprica for Archaea

7. To run paprica for Archaea we'll modify the code in step 4 to specify this domain:

```
> while read f; do ./paprica-run.sh $f archaea; done < samples.txt
```

8. After running paprica we'll combine the files as before using the following code:

```
> paprica-combine_results.py -domain archaea -o 2021_mangrove_study
```

9. Same as in step 6 this will produce an abundance of matrix tables for community structure, pathways, enzymes, and genome parameters for Archaea.

The "paprica-combine_results.py" script will generate the following files:

[bacteria or archaea].edge_tally.csv: This is an abundance table, giving the 16S rRNA gene copy number-corrected abundances for each edge in each sample. Columns are edges and rows are samples. **[bacteria or archaea].unique_tally.csv**: This is an abundance and copy number-normalized abundance table of unique sequences or ASVs (amplicon sequence variants).

[bacteria or archaea].taxon_map.csv: This is the taxonomic file that maps edge numbers to the name of the lowest consensus taxonomy (CEGs) or strain name (CCGs), and using this file in combination with many_tests.bacteria.seq_edge_map.csv will give you the taxonomy of the ASVs.

[bacteria or archaea].seq_edge_map.csv: This file maps ASVs to edge_number and gives the proportion that is placed to that edge number (in case the ASV is placed to multiple edges and you need to correct for this).

[bacteria or archaea].edge_data.csv: This file contains the mean genome parameters for each sample.

The above files will be useful for analysis of the community structure of the samples. Paprica also generates files of pathways and enzymes that can be helpful in trying to understand metabolism:

bacteria.ec_tally.csv: This file contains enzyme abundance for each sample.

bacteria.path_tally.csv: This file contains metabolic pathway abundance for each sample.

Step 5: Identify and extract key enzymes for the nitrogen cycle

⏱ **Timing: ∼ 5 min**

10. Clone the downstream_paprica repository: https://github.com/avishekdutta14/downstream_paprica

```
> git clonehttps://github.com/avishekdutta14/downstream_paprica.git

> cd downstream_paprica

> chmod a+x *py
```

11. We'll use the script "enzyme_view_v2.py", this script helps to name E.C. (Enzyme Commission) numbers and extracts the genes related to the sulfur and nitrogen cycles. The input for this script

is the file generated in the "paprica-combine_results.py" script (bacteria.ec_tally.csv) and enzyme_final.csv (present in the repository). Run the following:

```
> python3 enzyme_view_v2.py
```

Files generated running "python3 enzyme_view_v2.py":

enzyme_view.csv: This file is the enzyme abundance for each sample with E.C. numbers for enzyme names.

nitrogen_cycle.csv: This file is the enzyme abundance associated with the nitrogen cycle.

sulfur_cycle.csv: This file is the enzyme abundance associated with the sulfur cycle.

EXPECTED OUTCOMES

Paprica will produce several output files that can be helpful in downstream analysis. Running "paprica-run.sh" will generate intermediate analysis files for Bacteria and Archaea. For information on the intermediate analysis files visit: https://github.com/bowmanjeffs/paprica/wiki/3.-Output. The most useful files are the ones generated using the "paprica-combine_results.py" script. In Erazo and Bowman (2021), we used "bacteria[and archaea].unique_tally. csv" to understand microbial community structure and "bacteria.edge.data.csv" to determine key genome signatures such as genome size and number of 16S copies. We used "bacteria.path_tally.csv" to determine what pathways were significantly associated with levels of disturbance of shrimp aquaculture effluent (Figure 1). We additionally examined differences in enzymes associated to the nitrogen cycle by using the output form "enzyme_view_v2.py" and the "nitrogen_cycle.csv file", and identified key differences associated to nitrogen fixation and denitrification in mangrove-estuarine with low and high levels of disturbance (Figure 1).

LIMITATIONS

The paprica pipeline connects 16S rRNA gene sequences to likely metabolic functions and genomic characteristics. This (and similar) approaches are best thought of as models for genomic composition based on community structure (Langille et al., 2013). As such they are akin to the data reanalysis products common in other fields (e.g., meteorology and oceanography). It is important to note that like all reanalysis products paprica has limitations and there are a few aspects that need to be considered.

The paprica pipeline uses 16S rRNA data to predict metabolic pathways. This means that any eukaryotic or viral contributions will not be predicted; therefore, paprica is only able to help predict the portion of the metabolic potential contributed by the genomes targeted by the primers used. As with all amplicon-based approaches, biased primers can result in inaccurate predictions.

Paprica's metabolic inference is based on RefSeq (Haft et al., 2018) and pathway prediction with pathway-tools (Karp et al., 2021). Gaps or inaccuracies in pathway annotation or assignment of gene function can propagate to the prediction. Paprica uses a cutoff = 0.50 for metabolic assignment by default. This means that paprica will assign an enzyme or pathway to a CEG if it appears in 50 % or more CCGs in that clade. Depending on the objective of the study, the cutoff can be changed in "paprica-run.sh" by modifying the **-cutoff** flag for the "paprica-tally_pathways.py" command (mentioned in step 3: run paprica for bacteria).

It is important to consider that the edge numbers assigned to the reference tree by EPA-ng during each new build of the database are not comparable across database versions. This means that the edge number representing a genome in one version will be different in the next. This is important if your analysis extends across multiple versions of the paprica database (i.e., you reanalyze some data

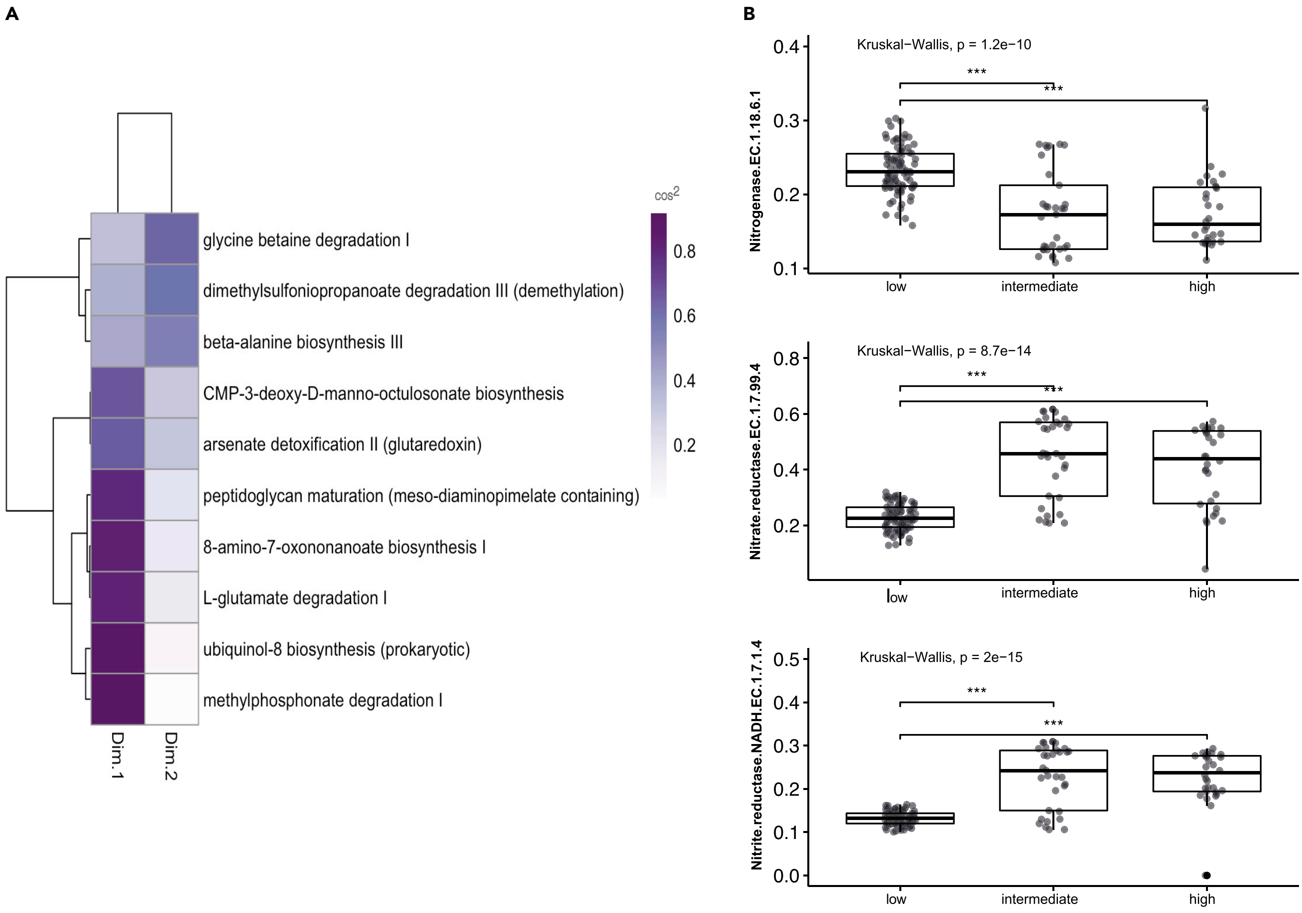


Figure 1. Metabolic pathways

(A) Contribution of top taxa from CCA ordination analysis and cos2 values.

(B) Nitrogenase EC 1.18.6.1, Nitrate reductase EC 1.7.99.4 and Nitrite reductase NADH EC 1.7.1.4 normalized (Hellinger transformation) abundance. Kruskal–Wallis test and p values with Dunn post-test, ***denotes p value < 0.001. This figure was published in Sensitivity of the mangrove-estuarine microbial community to aquaculture effluent. Iscience 24.3 (2021):102204.

months later after the release of a new database). You can keep track of which database was used for an analysis by referring to the database_created_at line in the *.[bacteria or archaea].sample_data.txt file produced by paprica.

For paprica to do phylogenetic placement, it depends on a good reference phylogenetic tree and a high-quality alignment. To maximize the quality of the reference trees paprica relies on a concatenated 16S and 23S rRNA gene alignment and subtrees for each phylum. Inevitably there are reference strains that are not well described by the phylogenetic model. It is recommended that (as with any classification scheme) you validate the identity of key taxa in your analysis. We've noticed that this is particularly important for endosymbionts and other bacteria with highly modified and streamlined genomes. A useful tool to validate taxa for paprica output is ROPE that relies on RDP classification (Wang et al., 2007).

You can find information on installation and code here: https://github.com/avishekdutta14/ROPE

TROUBLESHOOTING

There're few core bioinformatics programs that need to be installed for paprica to work (unless you're using the Docker image). Issues could arise during these installations. If you are a Linux user, the installation of the dependencies should be straightforward (tutorial): https://github.com/bowmanjeffs/paprica/blob/master/linux_install.sh

For a Mac OSX user the installation might need additional steps, you can look at this tutorial:

https://www.polarmicrobes.org/installing-paprica-on-mac-osx/

Problem 1

Installing dependencies such as gappa and EPA-ng will require installation of additional packages. It's important to review the documentation of each dependency by following the links provided in the key resources table of the GitHub repositories and make sure you satisfy all the requirements **(step 1).**

Installing gappa:

```
> git clone --recursivehttps://github.com/lczech/gappa.git

> cd gappa

> make
```

Potential solution

If you get an error associated with the command *make* is likely due to a missing library or missing requirements, make sure you install and follow instructions of missing libraries. Make sure you satisfy the following requirements:

Make and CMake 2.8.7 or higher

Up-to-date C++11 compiler, e.g., clang++ 3.6 or GCC 4.9 or higher.

Problem 2

If you encounter issues when running paprica, it is important to check that EPA-ng was properly installed **(step 1).** Here we describe how to build it from source, but you can also install it using Homebrew or Conda (see here: https://github.com/Pbdas/epa-ng#installation):

```
> sudo apt-get install autotools-dev libtool flex bison cmake automake autoconf

> git clonehttps://github.com/Pbdas/epa-ng.git

> cd epa-ng

> make
```

Potential solution

If the installation fails, it's probably due to a missing library. Check for the error message to determine what library you need to install. If you get an error associated with missing *zlib*, you will need to have *zlib1g-dev* installed, as well as zlib1g.

```
> sudo apt-get install zlib1g-dev
```

> *Note:* It's important to have an up-to-date C++11 compiler as mentioned in Problem 1.

Problem 3

If paprica fails to create files needed for the analysis, it's possible that one of the dependencies was not properly installed. For example, you run paprica and get an error associated to a missing file **(step 3):**

File not found error: [Err no2] No such file or directory: "/User/../paprica/test.bacteria/test.bacteria.phylum_resps.list.csv"

Potential solution

Make sure that the correct versions of dependencies are installed, and you are using Python 3. The code below shows how to install the Python modules and test them:

```
> pip3 install numpy

> pip3 install biopython

> pip3 install joblib

> pip3 install pandas

> pip3 install seqmagick

> pip3 install termcolor
```

Open Python:

```
> python3
```

Test the dependencies you installed:

```
> import numpy

> import Bio

> import joblib

> import pandas

> import termcolor
```

Seqmagick is not a module, so to test it run:

```
> seqmagick
```

Problem 4

If all the dependencies were properly installed and you still face issues running paprica, it is important to make sure that the dependencies are in your PATH (**step 3**).

Potential solution

Open your .bash_profile or .bashrc:

```
> nano .bashrc
```

Navigate to the end of the file and copy this code (make sure you modify your-user-name to its actual one):

```
> export PATH=/Users/your-user-name/infernal/binaries:${PATH}

> export PATH=/Users/your-user-name/infernal/easel:${PATH}

> export PATH=/Users/your-user-name/epa-ng/bin:${PATH}
> export PATH=/Users/your-user-name/paprica:${PATH}

> export PATH=/Users/your-user-name/gappa/bin:${PATH}
```

Problem 5

It's important that when running paprica, you create a text file with all the sample names and remove the extension ".fasta". If you follow step 2, the code should create the file with sample names without ".fasta". If you run paprica and get an error associated to "no such file" **(step 3).** Check the text file to confirm that the extensions were removed:

```
> nano samples.txt
```

Potential solution

Make sure your sample names have unique identifiers, and rerun code mentioned in step 4 to create a text file with all sample names.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Natalia Erazo (nerazo@ucsd.edu) or Jeff Bowman (jsbowman@ucsd.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The data that support the findings of this study and sequences were submitted to the NCBI sequence read archive (SRA) under BioProject ID: PRJNA633714. Paprica output is available on the author's GitHub repository: https://github.com/galud27/Microbial-community-in-mangrove-forest. The primary paprica repository is https://github.com/bowmanjeffs/paprica. Downstream paprica analysis code is available here: https://github.com/avishekdutta14/downstream_paprica.

ACKNOWLEDGMENTS

N.G.E. was supported by Organization of American States and SENESCYT fellowships. J.S.B. was supported by a grant from the Simons Foundation Early Career Investigator in Marine Microbiology program. We would like to thank the platform (mindthegraph.com) used here to create the graphical abstract.

AUTHOR CONTRIBUTIONS

Conceptualization & Development of Bioinformatic tool, J.S.B.; Methodology, N.G.E., A.D, and J.S.B; Writing – Original Draft, N.G.E.; Writing – Review & Editing, N.G.E, A.D, J.S.B.; Funding Acquisition and Supervision, J.S.B.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

Barbera, P., Kozlov, A.M., Czech, L., Morel, B., Darriba, D., Flouri, T., and Stamatakis, A. (2019). EPA-ng: massively parallel evolutionary placement of genetic sequences. Syst. Biol. *68*, 365–369. Edited by D. Posada. https://doi.org/10.1093/sysbio/syy054.

Bowman, J.S., and Ducklow, H.W. (2015). Microbial communities can be described by metabolic structure: a general framework and application to a seasonally variable, depth-stratified microbial community from the coastal west Antarctic Peninsula. PLoS One *10*, e0135868. Edited by C. Moissl-Eichinger. https://doi.org/10.1371/journal.pone.0135868.

Callahan, B.J., McMurdie, P.J., Rosen, M.J., Han, A.W., Johnson, A.J.A., and Holmes, S.P. (2016). DADA2: high-resolution sample inference from

Illumina amplicon data. Nat. Methods *13*, 581–583. https://doi.org/10.1038/nmeth.3869.

Czech, L., Barbera, P., and Stamatakis, A. (2020). Genesis and Gappa: processing, analyzing and visualizing phylogenetic (placement) data. Bioinformatics *36*, 3263–3265. Edited By R. Schwartz. https://doi.org/10.1093/bioinformatics/btaa070.

Dutta, A., Smith, B., Goldman, T., Walker, L., Streets, M., Eden, B., Dirmeier, R., and Bowman, J.S. (2020). Understanding microbial community dynamics in up-flow bioreactors to improve mitigation strategies for oil souring. Front. Microbiol. *0*, 3046. https://doi.org/10.3389/FMICB.2020.585943.

Edgar, R.C. (2010). Search and clustering orders of magnitude faster than BLAST. Bioinformatics *26*, 2460–2461. https://doi.org/10.1093/BIOINFORMATICS/BTQ461.

Erazo, N.G., and Bowman, J.S. (2021). Sensitivity of the mangrove-estuarine microbial community to aquaculture effluent. iScience *24*, 102204. https://doi.org/10.1016/j.isci.2021.102204.

Eren, A.M., Vineis, J.H., Morrison, H.G., and Sogin, M.L. (2013). A filtering method to generate high quality short reads using Illumina paired-end technology. PLoS One *8*, e66643. https://doi.org/10.1371/journal.pone.0066643.

Haft, D.H., DiCuccio, M., Badretdin, A., Brover, V., Chetvernin, V., O'Neill, K., Li, W., Chitsaz, F., Derbyshire, M.K., Gonzales, N.R., et al. (2018). RefSeq: an update on prokaryotic genome annotation and curation. Nucleic Acids Res. *46*, D851–D860. https://doi.org/10.1093/nar/gkx1068.

Karp, P.D., Midford, P.E., Billington, R., Kothari, A., Krummenacker, M., Latendresse, M., Ong, W.K., Subhraveti, P., Caspi, R., Fulcher, C., et al. (2021). Pathway Tools version 23.0 update: software for pathway/genome informatics and systems biology. Brief. Bioinform. *22* (1), 109–126. https://doi.org/10.1093/bib/bbz104.

Klempay, B., Arandia-Gorostidi, N., Dekas, A.E., Bartlett, D.H., Carr, C.E., Doran, P.T., Dutta, A., Erazo, N., Fisher, L.A., Glass, J.B., et al. (2021). Microbial diversity and activity in Southern California salterns and bitterns: analogues for remnant ocean worlds. Environ. Microbiol. *23*, 3825–3839. https://doi.org/10.1111/1462-2920.15440.

Langille, M.G., Zaneveld, J., Caporaso, J.G., McDonald, D., Knights, D., Reyes, J.A., Clemente, J.C., Burkepile, D.E., Thurber, R.L.V., Knight, R., et al. (2013). Predictive functional profiling of microbial communities using 16S rRNA marker gene sequences. Nat. Biotechnol. *31*, 814–821. https://doi.org/10.1038/nbt.2676.

Nawrocki, E.P., and Eddy, S.R. (2013). Infernal 1.1: 100-fold faster RNA homology searches. Bioinformatics *29*, 2933–2935. https://doi.org/10.1093/bioinformatics/btt509.

Rideout, J.R., He, Y., Navas-Molina, J.A., Walters, W.A., Ursell, L.K., Gibbons, S.M., Chase, J., McDonald, D., Gonzalez, A., Robbins-Pianka, A., et al. (2014). Subsampled open-reference clustering creates consistent, comprehensive OTU definitions and scales to billions of sequences. PeerJ *2*, e545. https://doi.org/10.7717/PEERJ.545.

Schloss, P.D., Westcott, S.L., Ryabin, T., Hall, J.R., Hartmann, M., Hollister, E.B., Lesniewski, R.A., Oakley, B.B., Parks, D.H., Robinson, C.J., et al. (2009). Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl. Environ. Microbiol. *75*, 7537–7541. https://doi.org/10.1128/AEM.01541-09.

Walters, W., Hyde, E.R., Berg-Lyons, D., Ackermann, G., Humphrey, G., Parada, A., Gilbert, J.A., Jansson, J.K., Caporaso, J.G., Fuhrman, J.A., et al. (2016). Improved bacterial 16S rRNA gene (V4 and V4-5) and fungal internal transcribed spacer marker gene primers for microbial community surveys. mSystems *1*, e0009–e0015. https://doi.org/10.1128/mSystems.00009-15.

Wang, Q., Garrity, G.M., Tiedje, J.M., and Cole, J.R. (2007). Naïve Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy. Appl. Environ. Microbiol. *73*, 5261–5267. https://doi.org/10.1128/AEM.00062-07.

Wilson, J.M., Chamberlain, E.J., Erazo, N., Carter, M.L., and Bowman, J.S. (2021). Recurrent microbial community types driven by nearshore and seasonal processes in coastal Southern California. Environ. Microbiol. *23*, 3225–3239. https://doi.org/10.1111/1462-2920.15548.