Springer Protocols

Terry J. McGenity
Kenneth N. Timmis
Balbina Nogales *Editors*

Hydrocarbon and Lipid Microbiology Protocols

Statistics, Data Analysis, Bioinformatics and Modelling

Springer

Springer Protocols Handbooks

Terry J. McGenity · Kenneth N. Timmis · Balbina Nogales
Editors

Hydrocarbon and Lipid Microbiology Protocols

Statistics, Data Analysis, Bioinformatics and Modelling

Springer

Editors
Terry J. McGenity
School of Biological Sciences
University of Essex
Colchester, Essex, UK

Kenneth N. Timmis
Institute of Microbiology
Technical University Braunschweig
Braunschweig, Germany

Balbina Nogales
Department of Biology
University of the Balearic Islands
and Mediterranean Institute
for Advanced Studies
(IMEDEA, UIB-CSIC)
Palma de Mallorca, Spain

ISSN 1949-2448 ISSN 1949-2456 (electronic)
Springer Protocols Handbooks
ISBN 978-3-662-49309-0 ISBN 978-3-662-49310-6 (eBook)
DOI 10.1007/978-3-662-49310-6

Library of Congress Control Number: 2016938230

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer-Verlag GmbH Berlin Heidelberg

This Volume is dedicated to the memory of Wilfred Röling, who played a major role in its conception and realisation.

While the Hydrocarbon and Lipid Microbiology Protocols series grew out of, and replaced, Volume 5 of the Handbook of Hydrocarbon and Lipid Microbiology that was published in 2010, it is greatly expanded in scope, detail and comprehensiveness. One major issue the editors confronted when developing the new concept for the series was whether or not to cover the topics of statistics, data analysis, bioinformatics and modelling. On one hand, excellent comprehensive treatises on these topics are available, and, furthermore, none of the editors had relevant expertise. On the other hand, these topics are absolutely central to experimentation in the field of hydrocarbon and lipid microbiology, especially for experimental design, sample workup, data handling and analysis and rigorous interpretation, and the whole idea of the Hydrocarbon and Lipid Microbiology Protocols series is to have in one collection essentially all methods and applications relevant to answer the question that is being posed. We solved our conundrum by consulting our friend and Scientific Advisory Board Member Wilfred Röling, who confirmed that a selection of key topics would seriously enhance the value of the series to potential users. This Volume is a result of that advice and Wilfred's suggestions of topics and authors.

Wilfred was intensively involved in this Volume and generously and enthusiastically gave his time, energy and intellectual input over the entire period of chapter recruitment and editing. The substantive role he played is *reflected in the fact that he wrote the Introduction. Tragically, he died on September 25, 2015, soon after writing the Introduction.*

Wilfred was a truly inspirational scholar, active in a range of topics within environmental microbiology and microbial biotechnology, and was one of only a handful of researchers worldwide able to effectively combine wet and in silico science. The loss to scholarly endeavour of his enormous potential is tragic. As a Scientific Advisory Board Member of this Hydrocarbon and Lipid Microbiology Protocols series, he provided key expertise, important council and an infectious enthusiasm that contributed significantly to the creation of this work, qualitatively and quantitatively. His reviews of submitted protocols were insightful and helped raise their quality and usefulness. He was one-of-a-kind and we shall miss him and his council in the future. We respectfully dedicate this Volume to his memory.

Preface to Hydrocarbon and Lipid Microbiology Protocols[1]

All active cellular systems require water as the principal medium and solvent for their metabolic and ecophysiological activities. Hydrophobic compounds and structures, which tend to exclude water, although providing *inter alia* excellent sources of energy and a means of biological compartmentalization, present problems of cellular handling, poor bioavailability and, in some cases, toxicity. Microbes both synthesize and exploit a vast range of hydrophobic organics, which includes biogenic lipids, oils and volatile compounds, geochemically transformed organics of biological origin (i.e. petroleum and other fossil hydrocarbons) and manufactured industrial organics. The underlying interactions between microbes and hydrophobic compounds have major consequences not only for the lifestyles of the microbes involved but also for biogeochemistry, climate change, environmental pollution, human health and a range of biotechnological applications. The significance of this "greasy microbiology" is reflected in both the scale and breadth of research on the various aspects of the topic. Despite this, there was, as far as we know, no treatise available that covers the subject. In an attempt to capture the essence of greasy microbiology, the *Handbook of Hydrocarbon and Lipid Microbiology* (http://www.springer.com/life+sciences/microbiology/book/978-3-540-77584-3) was published by Springer in 2010 (Timmis 2010). This five-volume handbook is, we believe, unique and of considerable service to the community and its research endeavours, as evidenced by the large number of chapter downloads. Volume 5 of the handbook, unlike volumes 1–4 which summarize current knowledge on hydrocarbon microbiology, consists of a collection of experimental protocols and appendices pertinent to research on the topic.

A second edition of the handbook is now in preparation and a decision was taken to split off the methods section and publish it separately as part of the Springer Protocols program (http://www.springerprotocols.com/). The multi-volume work *Hydrocarbon and Lipid Microbiology Protocols*, while rooted in Volume 5 of the Handbook, has evolved significantly, in terms of range of topics, conceptual structure and protocol format. Research methods, as well as instrumentation and strategic approaches to problems and analyses, are evolving at an unprecedented pace, which can be bewildering for newcomers to the field and to experienced researchers desiring to take new approaches to problems. In attempting to be comprehensive – a one-stop source of protocols for research in greasy microbiology – the protocol volumes inevitably contain both subject-specific and more generic protocols, including sampling in the field, chemical analyses, detection of specific functional groups of microorganisms and community composition, isolation and cultivation of such organisms, biochemical analyses and activity measurements, ultrastructure and imaging methods, genetic and genomic analyses, systems and synthetic biology tool usage, diverse applications, and

[1] Adapted in part from the Preface to *Handbook of Hydrocarbon and Lipid Microbiology*.

the exploitation of bioinformatic, statistical and modelling tools. Thus, while the work is aimed at researchers working on the microbiology of hydrocarbons, lipids and other hydrophobic organics, much of it will be equally applicable to research in environmental microbiology and, indeed, microbiology in general. This, we believe, is a significant strength of these volumes.

We are extremely grateful to the members of our Scientific Advisory Board, who have made invaluable suggestions of topics and authors, as well as contributing protocols themselves, and to generous *ad hoc* advisors like Wei Huang, Manfred Auer and Lars Blank. We also express our appreciation of Jutta Lindenborn of Springer who steered this work with professionalism, patience and good humour.

Colchester, Essex, UK	Terry J. McGenity
Braunschweig, Germany	Kenneth N. Timmis
Palma de Mallorca, Spain	Balbina Nogales

Reference

Timmis KN (ed) (2010) Handbook of hydrocarbon and lipid microbiology. Springer, Berlin, Heidelberg

Contents

About the Editors

Terry J. McGenity is a Reader at the University of Essex, UK. His Ph.D., investigating the microbial ecology of ancient salt deposits (University of Leicester), was followed by postdoctoral positions at the Japan Marine Science and Technology Centre (JAMSTEC, Yokosuka) and the Postgraduate Research Institute for Sedimentology (University of Reading). His overarching research interest is to understand how microbial communities function and interact to influence major biogeochemical processes. He worked as a postdoc with Ken Timmis at the University of Essex, where he was inspired to investigate microbial interactions with hydrocarbons at multiple scales, from communities to cells, and as both a source of food and stress. He has broad interests in microbial ecology and diversity, particularly with respect to carbon cycling (especially the second most abundantly produced hydrocarbon in the atmosphere, isoprene), and is driven to better understand how microbes cope with, or flourish in hypersaline, desiccated and poly-extreme environments.

Kenneth N. Timmis read microbiology and obtained his Ph.D. at Bristol University, where he became fascinated with the topics of environmental microbiology and microbial pathogenesis, and their interface pathogen ecology. He undertook postdoctoral training at the Ruhr-University Bochum with Uli Winkler, Yale with Don Marvin, and Stanford with Stan Cohen, at the latter two institutions as a Fellow of the Helen Hay Whitney Foundation, where he acquired the tools and strategies of genetic approaches to investigate mechanisms and causal relationships underlying microbial activities. He was subsequently appointed Head of an Independent Research Group at the Max Planck Institute for Molecular Genetics in Berlin, then Professor of Biochemistry in the University of Geneva Faculty of Medicine. Thereafter, he became Director of the Division of Microbiology at the National Research Centre for Biotechnology (GBF)/now the Helmholtz Centre for Infection Research (HZI) and Professor of Microbiology at the Technical University Braunschweig. His group has worked for many years, *inter alia*, on the biodegradation of oil hydrocarbons, especially the genetics and regulation of toluene degradation, pioneered the genetic design and experimental evolution of novel catabolic activities, discovered the new group of marine hydrocarbonoclastic bacteria, and conducted early genome sequencing of bacteria that became paradigms of microbes that degrade organic compounds (*Pseudomonas putida* and *Alcanivorax borkumensis*). He has had the privilege and pleasure of working with and learning from some of the most talented young scientists in environmental microbiology, a considerable number of which are contributing authors to this series, and in particular Balbina and Terry. He is Fellow of the Royal Society, Member of the EMBO, Recipient of the Erwin Schrödinger Prize, and Fellow of the American Academy of Microbiology and the European Academy of Microbiology. He founded the journals *Environmental Microbiology*, *Environmental Microbiology Reports* and *Microbial Biotechnology*. Kenneth Timmis is currently Emeritus Professor in the Institute of Microbiology at the Technical University of Braunschweig.

Balbina Nogales is a Lecturer at the University of the Balearic Islands, Spain. Her Ph.D. at the Autonomous University of Barcelona (Spain) investigated antagonistic relationships in anoxygenic sulphur photosynthetic bacteria. This was followed by postdoctoral positions in the research groups of Ken Timmis at the German National Biotechnology Institute (GBF, Braunschweig, Germany) and the University of Essex, where she joined Terry McGenity as postdoctoral scientist. During that time, she worked in different research projects on community diversity analysis of polluted environments. After moving to her current position, her research is focused on understanding microbial communities in chronically hydrocarbon-polluted marine environments, and elucidating the role in the degradation of hydrocarbons of certain groups of marine bacteria not recognized as typical degraders.

Introduction to Computer-Assisted Analysis in Hydrocarbon and Lipid Microbiology

Wilfred F.M. Röling[†]

Abstract

The large datasets generated in the immense complex field of hydrocarbon and lipid microbiology cannot be addressed and understood without computer-assisted analysis. Key components in computer-assisted analysis are bioinformatics, databases, statistics, and mathematical modeling. Computer-assisted analysis contributes to the entire scientific process: experimental design, data analysis, data storage, and utilization of these stored data. A wide range of relevant protocols are presented in this volume: protocols dealing with subjects that are also popular in other research areas and protocols addressing important topics specific for hydrocarbon and lipid microbiology. The basics of a major open-source programming language, Python, will be introduced.

Keywords: Bioinformatics, Computational biology, Mathematical biology, Statistics, Systems biology

1 The Need for Computer-Assisted Analysis in Hydrocarbon and Lipid Microbiology

Hydrocarbon and lipid microbiology comprises a very complex subject. There are at least ten thousand different hydrocarbon and lipid molecules. Crude oil spilled on a beach consists of over 17,000 different components, including many hydrocarbons which are toxic to living beings [1]. Thus, it is important that these hydrocarbons are removed, in which microorganisms come to the rescue, with or without some human help [1]. The capacity to degrade one or more hydrocarbons has been identified in more than 100 genera of microorganisms, especially bacteria and fungi [2]. The genomes of these microorganisms typically contain thousands of genes (e.g., Schneiker et al. [3]), and among microorganisms degrading the same compound, sometimes a large number of different pathways to degrade this compound have been observed. For example, five different pathways to initiate the aerobic degradation of toluene are

[†]Deceased, September 25, 2015

known [4]. Actively hydrocarbon-degrading microorganisms can be encountered in many environmental settings, including the very warm, the very cold and high salt. But even on a small spatial scale, heterogeneity in environmental conditions can be substantial and affect the occurrence of hydrocarbon-degrading microorganisms and their activities [5]. Uncontrollable factors, like currents, underground flows, and winds, may have different effects on different parts of the same crude oil-polluted beach, and assessing bioremediation treatments requires an experimental design with replicates [5, 6].

The above-sketched situation, while not even attempting to cover the whole field of hydrocarbon and lipid microbiology, is already mind-bogglingly complex. Thus, it is evident that computer-assisted analyses are essential to enhance insight into hydrocarbon and lipid microbiology, in order to pinpoint which factors regulate and control the occurrence and activities of microorganisms that consume or produce hydrocarbons and lipids. The importance of computer-assisted analysis has only increased, and will further increase, in the current (meta-)omics era, in which tremendous amounts of experimental data are being generated. Nowadays, we can easily determine the presence and expression of thousands of genes at once in a single species or in a hydrocarbon-degrading microbial community [7–9]. In microbial ecology, high-throughput sequencing of ribosomal RNA genes to establish community composition has become standard, with millions of sequences generated in a single run [7]. It becomes also increasingly possible to determine single-cell properties [7].

2 The Components of Computer-Assisted Analysis

Computer-based analysis can assist in every step of the scientific process: it can contribute to experimental design, data analysis, data storage, and utilization of these stored data. Key components in computer-assisted analysis are bioinformatics, databases, data analysis, statistics, and mathematical modeling: the subjects of this volume. It is not simple, nor truly needed, to distinguish these fields from each other: in the current large data set era, their integration is increasingly needed, and many bioinformatics tools contain components of several of these fields.

Bioinformatics draws relationships between data by combining computer science, statistics, mathematics, and engineering to study and process biological data. Thus, bioinformaticians develop methods and software tools to increase the understanding of biological processes. The field of bioinformatics is very diverse, from data mining, pattern recognition, to data visualization and network analysis. Major components are the analysis of sequence data, the use and management of various types of information, and the development of new algorithms and statistical measures that reveal relationships among variables in large data sets. Bioinformatics has similar approaches and aims as computational biology, but whereas bioinformatics organizes and analyzes basic biological data, computational biology builds theoretical models of biological systems.

The use of data stored in online, publicly accessible databases is central to many bioinformatics applications. Current databases cover almost everything from DNA and protein sequences, molecular structures, to phenotypes and biodiversities. Databases vary in their format, type, and detail of content and way of accession. Yet a database is in essence simply a collection of information, organized such that it can easily be managed, accessed, and updated. Data management systems are central to this goal, and these software applications enable interaction with the user, other applications, and the database itself to capture and analyze data.

Statistical data analysis is often incorporated in bioinformatics tools. One can say that while bioinformatics operates on data, statistics infers from data. To be more precise, statistics comprises collection, analysis, interpretation, presentation, and organization of data. It deals with all aspects of data including the planning of data collection: the experimental design. Two main statistical methodologies can be distinguished. Firstly, descriptive statistics summarizes data, for instance, its mean and standard deviation. Secondly, inferential statistics draws conclusions from data that are subject to random variation, by testing a null hypothesis, a general statement that there is no relationship between two quantities. Statistical analysis can act on a single variable (univariate statistics, e.g., analysis of variance) or on multiple variables at once (multivariate analysis, e.g., principal component analysis). The experimental data one would like to statistically analyze should truly represent the overall population it is derived from, such that inferences and conclusions can be extended to the population as a whole. Measurements are also subject to error, and statistics offers methods to estimate and correct for random variation within sampling and data collection procedures. Statistical methods for experimental design can help to reduce these issues at the onset of a study. Valid use of any statistical method requires that the data and their collection satisfy the underlying assumptions of the method; thus, it is important that the user familiarizes himself or herself with the basics of the statistical techniques he or she wants to use. Neglect or intentional misuse of the underlying assumptions can led to errors in description and interpretation, as well as the deliberate interpretation of only those data or use of only those statistical approaches that led to conclusions fitting a certain purpose. These misuses may lead to wrong decisions with regard to, for instance, environmental policies on oil pollution or biotechnological production of lipids.

At the heart of many statistical tools is often some kind of mathematical model. In general terms, a mathematical model is a description of a system using mathematical concepts and language, and the process of its development is called mathematical modeling. Mathematical models can take many forms; besides statistical models, they also include dynamical systems, differential equations, and game theoretic models and combinations of these models. They are usually composed of relationships and variables, in which relationships are described by operators (e.g., functions) while variables are quantifiable abstractions of parameters of the system of interest. The system might range from a single enzyme to a whole ecosystem. Mathematical models help to understand these systems, to study the effects of different components in them, and to make predictions about behavior. The lack of agreement between such theoretical models and experimental measurements often leads to important advances as better theories, and models, are developed. Mathematical models and their analysis and parameterization are an important part of the emerging field of systems biology (see Volumes Hydrocarbon and Lipid Microbiology: Synthetic and Systems Biology - Tools, and Hydrocarbon and Lipid Microbiology: Synthetic and Systems Biology - Applications, in this series).

Computer-based analysis tools come in many flavors, and especially those coded in free, open-source programming languages are of high interest and utility to scientists. The demand for new algorithms for the analysis of biological data derived from novel experimental approaches, the potential for innovative in silico experiments and data integration, and freely available open codes have enabled the development of an active, scientific community of interacting users and programmers of bioinformatics tools, as well as led to the release of a large range of widely used open-source software tools. A key example of such open-source programming languages is Python, which focuses on bioinformatics and systems biology. Learning the basics of such languages gives easy access to a wide range of bioinformatics and statistical tools and enables one to modify these tools or create new tools.

The above indicates that computer-assisted analysis covers a broad area, often combining a number of disciplines. Many new tools are being developed and published. On the one hand, this makes it often difficult to see the forest from the trees, as many of these tools appear to aim for the same but with different approaches and formats. Also, while some bioinformatics software and databases are maintained on a regular basis, others are not kept well up-to-date. On the other hand, new bioinformatics tools are still needed, in particular tools that integrate multilevel experimental data analysis with interrogation of various databases, to predict key functional properties of species and microbial communities [10].

Novel developments are also still needed and expected for network analysis, nonlinear approaches to biological processes, and decision maps or neural networks, to name just a few topics that are now popular.

3 The Content of This Volume

It is impossible to cover in this volume all statistics, bioinformatics, databases, and modeling approaches relevant to hydrocarbon and lipid microbiology, let alone treat their theoretical backgrounds. For instance, for statistics there are many good textbooks around, from statistics at university level [11] to advanced statistics [12]. The chapter by Gibbons in this volume provides a detailed example on how to analyze a multivariate data set. Two major, yet often overlooked issues in statistics are experimental design and correcting for multiple testing. If an experimental design is not adequate, then the resulting experimental data will have limiting utility and their collection may have been a waste of effort and money. This subject is dealt with by Larsen. Qiime (protocol by Peña) is an example of a multifaceted bioinformatics tool currently widely used by microbial ecologists to analyze high-throughput sequence data; it includes database searches and statistical data analysis. MG-RAST (Meyer) is a metagenomics service for the analysis of microbial community structure and function. Qiime and MG-RAST are very popular with experimental microbial ecologists. New approaches to data analyses are continuously appearing, also inspired by technical advances in data generation, such as high-throughput sequencing of many samples in parallel. Emerging fields are, for instance, network biology and protein-protein interactions, for which Ma and Pazos, respectively, provide protocols. Many of the above-mentioned protocols employ open-source computer codes; the chapter by Knights and Meulemans introduces the basics of coding in Python.

The above-mentioned protocols are not specific for hydrocarbon and lipid microbiology, yet a number of tools in this volume are. These protocols explain how to predict reaction kinetics and thermodynamics of biodegradation (chapter by Dolfing), biodegradation pathways and networks (Pazos), and the environmental fate of hydrocarbons (Wu and Coulon).

All the protocols together cover several major topics with respect to hydrocarbon and lipid microbiology.

References

1. Head IM, Jones DM, Röling WFM (2006) Marine microorganisms make a meal of oil. Nat Rev Microbiol 4:173–182
2. Prince RC (2005) The microbiology of marine oil spill bioremediation. In: Ollivier B, Magot M (eds) Petroleum microbiology. American Society for Microbiology Press, Washington DC, pp 317–336
3. Schneiker S, dos Santos V, Bartels D, Bekel T, Brecht M, Buhrmester J, Chernikova TN, Denaro R, Ferrer M, Gertler C, Goesmann A, Golyshina OV, Kaminski F, Khachane AN, Lang S, Linke B, McHardy AC, Meyer F, Nechitaylo T, Puhler A, Regenhardt D, Rupp O, Sabirova JS, Selbitschka W, Yakimov MM, Timmis KN, Vorholter FJ, Weidner S, Kaiser O, Golyshin PN (2006) Genome sequence of the ubiquitous hydrocarbon-degrading marine bacterium *Alcanivorax borkumensis.* Nat Biotechnol 24:997–1004
4. Parales RE, Ditty JL, Harwood CS (2000) Toluene-degrading bacteria are chemotactic towards the environmental pollutants benzene, toluene, and trichloroethylene. Appl Environ Microbiol 66:4098–4104
5. Röling WFM, Milner MG, Jones DM, Fratepietro F, Swannell RPJ, Daniel F, Head IM (2004) Bacterial community dynamics and hydrocarbon degradation during a field-scale evaluation of bioremediation on a mudflat beach contaminated with buried oil. Appl Environ Microbiol 70:2603–2613
6. Prosser JI (2010) Replicate or lie. Environ Microbiol 12:1806–1810
7. Mason OU, Hazen TC, Borglin S, Chain PSG, Dubinsky EA, Fortney JL, Han J, Holman HYN, Hultman J, Lamendella R, Mackelprang R, Malfatti S, Tom LM, Tringe SG, Woyke T, Zhou JH, Rubin EM, Jansson JK (2012) Metagenome, metatranscriptome and single-cell sequencing reveal microbial response to deepwater horizon oil spill. ISME J 6:1715–1727
8. Lu ZM, Deng Y, Van Nostrand JD, He ZL, Voordeckers J, Zhou AF, Lee YJ, Mason OU, Dubinsky EA, Chavarria KL, Tom LM, Fortney JL, Lamendella R, Jansson JK, D'Haeseleer P, Hazen TC, Zhou JZ (2012) Microbial gene functions enriched in the deepwater horizon deep-sea oil plume. ISME J 6:451–460
9. Hazen TC, Dubinsky EA, DeSantis TZ, Andersen GL, Piceno YM, Singh N, Jansson JK, Probst A, Borglin SE, Fortney JL, Stringfellow WT, Bill M, Conrad ME, Tom LM, Chavarria KL, Alusi TR, Lamendella R, Joyner DC, Spier C, Baelum J, Auer M, Zemla ML, Chakraborty R, Sonnenthal EL, D'Haeseleer P, Holman HYN, Osman S, Lu ZM, Van Nostrand JD, Deng Y, Zhou JZ, Mason OU (2010) Deep-sea oil plume enriches indigenous oil-degrading bacteria. Science 330:204–208
10. Röling WFM, van Bodegom PM (2014) Toward quantitative understanding on microbial community structure and functioning: a modeling-centered approach using degradation of marine oil spills as example. Front Microbiol 5
11. Quinn G, Keough M (2002) Experimental design and data analysis for biologists. Cambridge University Press, Cambridge
12. Legendre P, Legendre L (1998) Numerical ecology. Elsevier Science, Amsterdam

A Brief Review on the Ecological Network Analysis with Applications in the Emerging Medical Ecology

Zhanshan (Sam) Ma, Chengchen Zhang, Qingpeng Zhang, Jie Li, Lianwei Li, Linyi Qi, and Xianghong Yang

Abstract

Ecology studies the relationship between organisms and their environment, and a well-accepted paradigm of ecological science is to divide the subject into molecular ecology, autecology, population ecology, community ecology, ecosystem ecology, and landscape ecology, according to the scale of investigation. Network analysis can arguably be a powerful tool for any scale of the ecological research because it is particularly suitable for investigating the complex relationships in multivariate and multidimensional settings. In this article, we present a glimpse of the state-of-the-art research in ecological network analysis, focusing on the species interaction network (SIN) in the human microbiome. Our choice of the focus takes advantage of the recent revolutionary metagenomic technology that has open unprecedented opportunities to the study of microbial communities, especially the human microbiome. We also present a case study to demonstrate the analysis of SINs in both healthy and diseased oral microbiomes. The analysis reveals striking characteristic changes in the human oral microbiome associated with the transition from healthy regime to periodontitis regime.

Keywords: Ecological networks, Human microbiome, Metagenomics, Microbial networks, Species interaction networks

1 Introduction

We live in an era of networking. The Internet of information highway is extending to the Internet of Things (IoT). Our life is dependent on networks, from transportation network to mobile phone network, from power grid network to bank payment network, from the Internet to virtual social networks. Our economy is dependent on critical national infrastructures that are heavily dependent on the computer networks. A *Science of Networks* was born at the beginning of the new century thanks to the efforts of generations of scientists and technologists. Network science or using a less ambitious term, network analysis, offers powerful tools to study complex multidimensional, multivariate, and dynamic relationships in nature and society. The interactions between network science and ecological science seem particularly fruitful. *Ecological networks* have been constructed with an increasing frequency, and ecological network analysis has become a crucial tool in ecological research (e.g., [1–14]). In fact, ecologists are among the earlier pioneers of network analysis with their food web research as early as 1920s (e.g., [15–20]).

Microbial ecology has been experiencing rapid advances thanks to the metagenomic revolution that originated in environmental microbiology, expanded to the human microbiome research with the launch of the human microbiome project (HMP) around 2007 [21–23], a natural follow-up of the millennium HGP (Human Genome Project), and more recently to the earth microbiome project (EMP) [24, 25]. Human metagenome is often referred to as the *second* human genome emphasizing its importance to humans. Indeed, ecological theory can be a unifying driving force and test-bed for this revolution (e.g., [26–28]), and network science may play a critical role in the expansion of the metagenomic technology into the research on the human microbiome and microbial ecology at large (e.g., [6, 7, 9, 10, 13, 14]). In this article, we present a brief review on the network analysis and its applications to the ecological research, with a particular reference to the human microbiome. After a concise review of the basics of network analysis as well as major ecological networks, we present a case study with the human oral microbiome dataset to demonstrate the potential of ecological network analysis in investigating the health and disease implications of the human microbiome, specifically a shift from healthy oral microbiome to the regime associated with periodontitis.

2 Fundamental of Network Analysis

To introduce some recent advances in the network analysis of food webs, we first explain a few necessary concepts and models. Our explanation can only touch the very surface of those concepts and models due to the scope of this article, and interested readers should refer to comprehensive monographs and reviews such as Girvan and Newman [81], Newman [82–84], Junker and Schreiber [64], Barabási et al. [85], and Barabási [86]. The mathematical approaches involved in current network analysis are quite diverse, but graph theory, especially random graph theory, should be one of the most fundamental areas supporting the new science of network. Graph theory is a branch of mathematics with a history of at least tracing back to Euler's study of "Konigsberg bridge problem" in 1736. The problem was to determine if it is possible to walk through seven bridges that were built over the river Pregel in the town of Konigsberg with the condition that the walker would cross

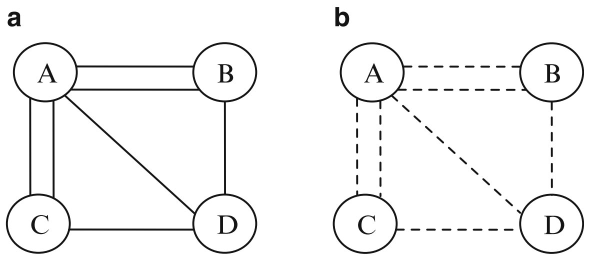


Fig. 1 (**a**) Graph representation of Euler's 1736 Konigsberg seven-bridge problem. (**b**) A random graph version of the seven-bridge problem: *dash line* indicates the existence of bridges (edges) is probabilistic. Note that this is not a rigorous interpretation of random graph concept

each bridge exactly once. Euler formulated the problem as a graph model (Fig. 1a), $G = (V, E)$, where $V = (A, B, C, D)$, representing the four regions of the town Konigsberg connected by the seven bridges built on the river Pregel, $E = (AC, AB, AD, BA, BD, CA, CD)$. Euler proved it is impossible to walk through all four regions (nodes in V) without crossing some bridges more than once.

More than 200 years after Euler's work, Erdos and Renyi [29, 30] introduced a revolutionary change to graph theory by introducing probability theory to the study of graphs. Their pioneering work set the foundation for a new branch of graph theory – random graph theory [29–32]. What they were excited initially was that certain properties of graphs are very simple to prove by introducing probability to describe the existence of edges in the graph. With an intuitive, but not rigorous mathematically, explanation, traditional graph theory assumes that the existence of an edge in a graph is a certainty or deterministic matter: either existence or non-existence. Formally, Erdos and Renyi [30] random graph model is a graph $G = (V, E)$ (Fig. 1b), V consisting of N_V nodes, and E consisting of N_E edges. Since there are $N_V(N_V - 1)/2$ possible edges that may connect the N_V nodes, the probability that there is an edge between any two nodes in the graph is:

$$p = 2N_E/N_V(N_V - 1). \tag{1}$$

An alternative formulation for Erdos–Renyi is to assume that the probability to connect any two nodes in the N_V nodes is equal with a value p, and then the number of edges in the graph is a random variable with the mathematical expectation $E[N_E]$:

$$E[N_E] = pN_V(N_V - 1)/2. \tag{2}$$

With random graph theory, several new concepts are particularly important, and some of them are unique for random graphs. One of the most basic properties of vertex V_i is its degree k_i, i.e., the number of edges adjacent (directly connected) to the vertex. The degree distribution $p(k)$ of a graph is the probability distribution that a randomly picked vertex in the graph has a degree of k. The degree distribution of previously mentioned Erdos–Renyi random graph is binomial and can be approximated with Poisson distribution for large networks (when $N_V \rightarrow \infty$). That is, the probability of a vertex to have degree k is:

$$p(k) = e^{-\lambda}\lambda^k/k! \tag{3}$$

where $\lambda = pN_V$ denotes the average degree, or the expectation of the Poisson distribution. Since Poisson distribution has the same mean and variance, a typical Erdos–Renyi random graph is rather homogenous, and most vertices have a similar degree, distributed near symmetrically around the average degree (λ). In contrast, many real-world networks such as social networks and the Internet show highly skewed or heterogeneous degree distribution. In those networks, the degree distribution follows the so-called power law distribution:

$$p(k) \propto k^{-\gamma} \tag{4}$$

where γ is the degree exponent. With power law degree distribution, most vertices only have a small number of edges (the so-termed fat tail), but a small number of vertices (the so-termed hub nodes) possess a disproportionably large number of vertices connected to them. Barabási and Albert [33] termed this kind of network scale-free power distribution networks, or simply *scale-free networks*. Scale-free networks are very robust against the failure of randomly chosen nodes, but can be very vulnerable to targeted attack due to the cascade failures induced by the removal of critical hub nodes [34].

Besides degree distribution, another fundamental metric of a graph is the *distance* d_{ij} between any two vertices V_i and V_j, which is defined as the shortest path length between the pair of vertices, i.e., the minimum number of edges that must be traversed to travel from vertex V_i to V_j. After defining the distance between any pair of nodes, the diameter of a network can be defined as the maximal distance of any pair of vertices, i.e., $d_m = \max(d_{ij})$. Similarly, the *average* or *characteristic path length* of a network, denoted as $d = \langle d_{ij} \rangle$, can be defined as the average distance between all pairs of vertices. Strikingly, the average path lengths of many real-world and empirical networks are usually very small, as compared to the sizes of these networks. The network with this kind of surprisingly small average path length is termed "small-world networks." For example, it is found that the average shortest path length between any two metabolites within a cellular metabolism network of metabolites (vertices) linked by biochemical reactions (edges) is only about three, and this characteristic distance seems largely independent of specific organism. Unlike the degree distribution property, the small-world property is not truly a network property because of the lack of objective criterion for the "smallness." For example, the most well-known "six degrees of separation" in small-world networks originated from the assertion that all people on earth are separated from each other by just six intermediate acquaintances. However, the number *six* should not be taken literally. To deal with this lack of objective metrics for small-world networks, it is suggested that small networks refer to networks whose average path length d increases slower or equal than the logarithm of the network size, i.e., $d < \log(N_V)$ when $N_V \to \infty$.

Although Erdos–Renyi random graph model does not follow the scale-free power law degree distribution with a highly homogenous degree distribution, it does display small-world property when $p \geq \log(N_V)/N_V$. Erdos–Renyi can explain percolation theory that is primarily concerned with the percolation threshold when the whole network becomes connected as a result of phase transition crossing the threshold. When $p \geq \log(N_V)/N_V$, the whole network is connected for almost all realizations of the Erdos–Renyi network.

Yet another important network metric is the *clustering coefficient.* It addresses the local cohesiveness phenomenon and measures the probability that two vertices with a common neighbor are connected. With undirected networks (the edge does not have a direction, AB and BA represent the same edge), the clustering coefficient C_i of the vertex V_i is computed as:

$$C_i = \frac{E_i}{k_i(k_i - 1)/2}, \tag{5}$$

where E_i is the realized number of edges between the neighbors, and the denominator is the maximally possible number $E_{\max} = k_i(k_i - 1)$. C_i is a measure of the cohesiveness of vertex V_i's local neighbors. There is also the global or average clustering coefficient of the network, $C = E[C_i]$. There is an alternative but equivalent definition for C, defined in terms of the number of *triads*, which are triples of vertices where all three vertex are pairwisely connected with each other. Triples are the three-vertices structure where at least one vertex is connected to both others. The global or average clustering coefficient is calculated as:

$$C = \frac{3 \times \text{number of triads}}{\text{number of connected triples}}. \tag{6}$$

The high clustering coefficients, as displayed by many empirical networks, indicate a local cohesiveness and a tendency of vertices to form clusters or groups. However, statistical testing and interpretation of clustering is not a trivial task and an estimated clustering coefficient C should be compared with an appropriate null model to verify that the estimated C is indeed significant.

Beyond the clustering coefficient, which is a rough estimate of the heterogeneity of network, an important aspect of network analysis is to detect clusters in the network. The network-clustering problem can be defined as follows: given a graph $G^0 = (V^0, E^0)$, one tries to find subsets (disjoint of the subsets is not required) $\{V_1{}^0, \ldots, V_r{}^0\}$ of V_0 such that $V^0 = \sum_{i=1}^{r} V_i^0$. Each of the subsets is a *cluster* defined by structures such as cliques or other distance and diameter-based models. The defining criteria for the structures represent the cohesiveness required for the cluster. The clustering problem is often approaches with one of the two following optimization problems: (1) minimizing the number of clusters with the constraint that every cluster found possess cohesiveness over a prescribed threshold; (2) maximizing the cohesiveness for each cluster found with the constraint that the number of clusters resulted does not exceed a prescribed number K.

The last concept in network analysis to introduce in this chapter is the *network motif*. The term network motif refers to particular sub-graphs representing patterns of local interconnections between network elements. Motifs are building blocks and design patterns of complex networks. One example of motifs that is particularly important in gene regulation networks is the feed-forward loop (Fig. 2a), which can filter particular information. Another example of motifs is the so-termed multi-input motif (Fig. 2c). The multi-input motif can be decomposed as single-input motif (Fig. 2b). Furthermore, the topology of multi-input motif is isomorphic with the well-known *utility graph* in graph theory when the direction of the edge is ignored. In a utility graph, each household (A, B, C; the top vertices in Fig. 2c) is pair-wisely connected to all utility companies [such as electricity (E), phone (F), and gas (G), the bottom vertices], but there is connection neither between the households nor between the utility companies. The properties of utility graph should be familiar to computer scientists and can be readily applied to revealing the significance of multi-input motif in biological networks.

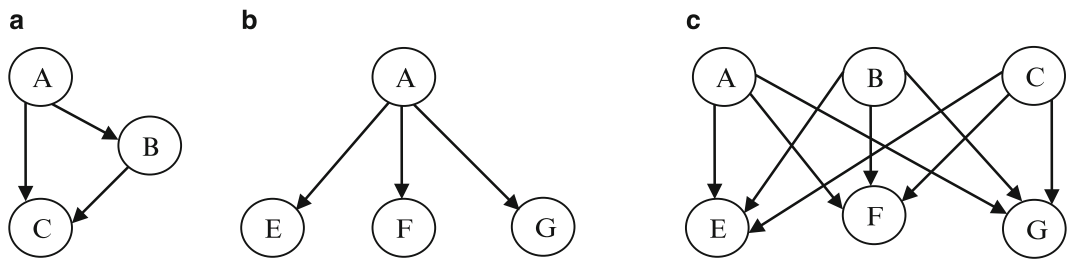


Fig. 2 Three motifs with functionality relevance in biological networks (redrawn from Schwobbermeyer [35]): (**a**) feed-forward loop motif, (**b**) single-input motif, (**c**) multi-input motif

Network motifs were identified as statistically significant over-represented patterns of local interconnections in complex networks. It is argued that in the case of biological networks, the motifs have been shaped by natural selection during the evolution processes.

Erdos and Renyi [26] seminal research spawned extensive studies on random graphs, mostly by pure mathematicians and physicists studying percolation theory. Random graph theory is well recognized as one of the most exciting advances in graph theory in the twentieth century [25, 26, 31, 32], and has become intimately interwoven with several fields of mathematics and physics including percolation theory, random cluster, random matrix, expanded graphs, interacting particle systems (IPS), stochastic cellular automata, etc. Most of these theories have found wide applications in the study of complex systems. Given that networks including food webs are essentially more complex systems, the application of random graph theory and its extensions to network analysis should be a natural development. Somewhat surprisingly, the application of random graph theory to biological and ecological systems has been limited to a few cases, notably the cascade model in community ecology until the new century. The avalanche of network analysis, of which random graph theory forms the theoretic foundation, was not triggered until the publication of two independent seminal papers published by Watts and Strogatz [36] on the dynamics of "small-world" networks in the journal *Nature*, and by Barabasi and Albert [33] on the emergence of scaling in random networks in the journal *Science*, respectively. A commonality of both the papers is the extension of basic random graph models so that they can better fit to the patterns exhibited by some empirically observed networks in social, technological, and natural networks. In the paper published by Watts and Strogatz [36], the authors found that the neural networks of a nematode (*Caenorhabditis elgans*), the power grid of the western US, and the collaboration graph of film actors share similar properties: all three networks are highly clustered (with densely connected clusters), yet they have small characteristic (average) path length. This category of networks described by Watts and Strogatz [36] is the previously mentioned "small-world networks," apparently mirroring the small-world phenomenon previously discovered in social network analysis. In another paper published by Barabasi and Albert [33] one year later, the authors discovered that the degree-distributions of those networks follow a scale-free power law distribution, and they coined the term scale-free networks. As mentioned previously, in a scale-free network, many vertices have relatively few connections, but some (hubs) are highly connected. A consequence of the scale-free connectivity distribution is that the network is very robust against random removal of nodes, but can be very fragile when the hub nodes are knocked out, especially when cascading failure is triggered by the removal of hub.

The snowball of network analysis literature was quickly formed when many researchers across various disciplines simultaneously looked for evidence of small-world and/or scale-free networks. In some disciplines such as social network analysis and food web research in ecology, where similar approaches had already been relatively widely adopted prior to the avalanche triggered by the 1998–1999 Nature and Science papers, the avalanche indeed stimulated the renewed interests to those earlier approaches and further analysis with the approaches from the "new science of networks," as referred to the discipline established by the avalanche in some recent literature. Generally, the responses and expectations from these disciplines are moderate and quite rational, perhaps due to the early experience. Some social network analysts contested that they have been performing similar network analysis routinely for decades. Ecologists seem to have been refrained from the similar claims, but the concepts and models of power laws, fat-tail scale-free distributions appears in ecological literature at least back to 1961 when Taylor [37] published a seminal paper in journal *Nature*, in which he demonstrated that the abundance distribution of biological populations in space follows a simple power law. Taylor's power law has been verified by numerous follow-up studies, especially on insects. Theoretically, the power law was extended to explain both temporal and spatial distributions of insect populations, as well as the regulation of population dynamics. Of course, the previously discussed cascade food web model by Cohen et al. [19] in 1980s is even closer to the random network approach.

In some other fields, notably bioinformatics and computational biology, gigantic data sets generated by the DNA sequence and various "–omics" techniques offer fertile soils for testing new network theory. In addition, biologists have been eagerly searching for analytic, statistic, and computational methods that can be helpful to their endeavor to discover biological insights and patterns from those hopelessly large datasets since the 1990s. This supply-demand synchrony obviously is responsible for the wide adoption of network analysis in bioinformatics and computational biology. Numerous software packages for analyzing biological networks have been released in recent years [38–44]. The available software further accelerates the application of network analysis in biology.

In still other applications, the insights from network analysis may be marginal, but many less rigorously scrutinized applications were performed. Even in studies where the potential insights from network analysis can be very significant, the collection, organization, and analysis of the datasets must follow proper procedures and be tested with appropriate null models. Despite huge influences and popularity generated by the new science of networks, the new science is not free from critiques. Besides critics on some implausible applications, some more serious critics on the new science of networks come from mathematicians and graph theoreticians (e.g., [45]). For example, in some cases, analytical solutions to some random graph problems have been worked out by mathematicians, but simulations were used to find solutions by practitioners. The situation is not unlike what happened with the application of chaos theory in the 1970s and 1980s. A practical difficulty is that the theoretical foundation of the new network science is beyond the reach of most biologists and perhaps many computer scientists given that random graph theory (the foundation) is a specialized and advanced field of pure mathematics. It is therefore the responsibility of the theoreticians of the new network science to safeguard the proper applications and to keep the theory of the new science grounded with its mathematical foundation, i.e., mostly random graph theory.

It should be noted that the quality of data used to conduct network analysis is arguably the most critical factor that influences the success or failure of the operation. Nevertheless, in practice, the datasets available for many ecological network analyses are far from ideal. For example, the so-termed *species interaction network* (SIN) is often built with the species (or OTU: Operational Taxonomic Unit) abundance data. Indeed, in most cases, the network links (edges) of SIN were established by computing the correlation coefficients among species abundances. Hence, strictly speaking, species correlation network is a more appropriate term for networks built with such kind of datasets. In order to follow convention in ecological studies, we preserve the usage of *SIN*. In other words, we preserve the usage of interaction because at present correlation data are often the only available data type for building ecological networks. Perhaps even looser usages are the somewhat analogical usages of the terms such as cooperation vs. competition, facilitation vs. inhibition, which actually boil down to positive vs. negative correlations. Finally, although it is well known that correlation is not equivalent with causation, researchers sometimes could not help from crossing the fine line. Despite these issues and pitfalls, we still believe that network analysis is one of the most powerful apparatuses to investigate ecological interactions (relationships) as long as proper cautions are taken.

3 Major Ecological Networks Studied in Microbial Ecology

3.1 Food Web Networks

The study of the food web, or the feeding relationship among species in the community, is one of the oldest and also the most important research topics in community ecology. Generally, there are two major categories food web models [46]: population-level models and phenomenological models. The former was formulated based on species interactions models such as Lotka–Volterra differential equation system models and emphasizes the interactions among individuals and resulting emergent properties at food web level. The most well-known work in this category models beyond Lotka–Volterra [47, 48] model should be Robert May's [17] monograph "Stability and complexity in model ecosystems." The latter was built upon a set of heuristic rules with the objective to explain some empirical observations [46]. In the category of phenomenological models, network models (often represented as graphs) are the most widely adopted. In these models, the structure of food webs or ecological communities in general is represented with a graph model, $G = (V, L)$, where V is the set of vertices representing species in the community, and L is the set of links representing predation in the food web or representing species interactions (in more general SINs of the community). In addition, the vertices of a food web network can be the so-termed *trophic guild*, which consists of those species that share the same predators and preys [49]. The phenomenological observation and modeling of food web should have similarly long history with the population-level modeling, but arguably the most interesting research has been performed in the last few decades, starting with Cohen et al. [19] cascade model of food web and further stimulated by recent resurgence of network analysis [50–52]. Of course, the distinction between the two types of modeling approaches is mostly artificial, and both approaches are intended to reveal the nature of food web. Nevertheless, it is highly possible that both approaches just touch different parts of the food web 'elephant.'

Alternative classifications for food webs exist. One particularly interesting scheme is the distinction of three kinds of food webs, primarily from a network perspective. Those three kinds of food web networks are [49]: binary food web, quantitative trophic food web, and ecological information food webs. The first kind is the traditional "who eat whom" food web, which suffers from the ignorance of relative importance of the links. Binary network is effective for detecting qualitative scaling patterns of food webs, and the results can be compared with other complex networks across disciplines such as social and technological networks (e.g., friendship network and the Internet). It is inadequate for analyzing the functionalities of ecosystems due to its ignorance of the link importance. The second kind is built upon the quantification of "who eats whom how much; who recycles how much," and it can overcome some limitation of binary food webs by incorporating individual properties such as body size or biomass. Quantitative food web models can be utilized to describe fluxes and cycling of carbon (energy) and nutrients in a community, and may be mass-balanced and assessed by network analysis when data are sufficient to support the analysis. This allows relatively deep understanding of the ecosystem functionalities. The third kind food web network – ecological information network – can be built to overcome some limitations of the quantitative food webs. For example, quantitative food webs are still static given that they are snapshots of the spatial-temporal dynamics of the community; furthermore, the interaction strengths between network nodes are influenced not only by the fluxes, but also by the environment and information communications. The dynamic information exchanges in the community, although they may consume little energy and materials to transport, may modulate the strengths of competition, predation, and mutualism and exert profound influence on the community dynamics. The examples of information communication in biological community abounds. Gaedke [49] cited the role of quantitative minor components such as polyunsaturated fatty acids, vitamins and sterol, which may determine the growth rate of the consumer. Other examples he cited for the information modulation role include allelochemicals such as kairomones and pheromones. In bacterial communities, quorum sensing is another obvious example of information modulation of the community dynamics. To consider the dynamic information exchanges in a community, the third kind of food web – ecological information networks – becomes necessary [49].

The data requirements for building ecological information networks are much more demanding than those for building the first two kinds of food web networks. Gaedke [49] listed two types of mathematical models for modeling information networks: (1) system of differential equations for 'homogenous' populations such as unicellular algae population in an open water body, where all individuals behave similarly and (2) individual-based and spatially explicit models for populations where individual life stages and neighborhood relationships may influence the interaction strengths and other ecological processes, such as tree growth in plant ecology. Here, we suggest that a third kind of mathematical models should be particularly effective in building ecological information networks, that is, evolutionary game theory models (e.g., [53]). Since at present, relatively few studies have been conducted to build quantitative and ecological information food web networks, we only discuss binary food webs in this chapter.

Ecologists have proposed and extensively tested some major food web models including cascade model, niche model, nested hierarchy model since 1980s. Although the usage of the term network in the food web literature is relatively rare, their resemblance to the recent surge of network analysis since late 1990s, pioneered by Watts and Strogatz [36] and Barabasi and Albert [33] is obvious. Firstly, both cascade model (including its modifications) and network models are essentially the extended random graph models, in which probability theory is applied to study the properties of graphs. Secondly, although the term network was not often used in food web research, food web is nothing but a network, perhaps except that cycles are rare in food web networks. To some extent, food web ecologists were among the pioneers of the current new science of networks. Nevertheless, there is indeed considerable amount of differences between the approaches adopted by food web ecologists and those developed by the *new network scientists.*

Although there are repeated claims and suggestions in literature that the food web shares common properties, such as 'small-world' and 'scale-free' properties, with many large-scale social, technological and biological networks such as acquaintance network, the Internet, and metabolic pathways, the evidence accumulated so far does not allow the close of the case. Although the contention on this issue so far is not intense, a somewhat 'pessimistic' prediction we make is that the contention may mirror the traditional debates in food web studies spawned by MacArthur [54], May [17], and Cohen et al. [19]. This pessimistic opinion of course does not imply that network analysis is less useful for food web analysis. In contrary, we are optimistic that network analysis can help us to deepen our understanding on the nature of food webs. The pessimistic aspect of our opinion comes from two observations: (1) Like previous approaches developed in food web analysis, the network analysis approach may be effective in touching some aspects of the 'elephant.' The existing food web models such as the cascade and niche models, which are essentially very similar to the approaches from the net network analysis, may have already demonstrated the advantages and limitations of the network analysis in general. (2) As argued in the perspective section, the idea of food web paradigm may be flawed because food web in strict sense is an entity that excludes other potentially very important species interactions such as mutualism and perhaps also competition. In other words, food web only captures the 'bloody' side of the nature – predation, ignores the cooperation which appears as ubiquitous as competition in the natural world. With an analogy of the 'elephant,' we have been observing the behavior of an elephant in the zoo, rather than in its natural habitat.

In summary, the food web (network) is a simplified community model, and the model captures some of the most fundamental aspects of community structure. Ecologists invented the food web paradigm, perhaps with the hope that its dynamics can represent the dynamics or stability of the community as a whole. The reality is that even with the simplified paradigm of food web, the efforts to understand the relationship between the community structure (measured with diversity) and stability have frustrated many ecologists to such an extent that some of them openly call for the abandon of the ship and call the diversity-stability a lost paradigm. While abandon of the ship may not be the option, we must recognize that there should be some rational components in those complaints and frustration with food web paradigm. Otherwise, why can we be so often to reach diametrically opposite conclusions by using different models with the same data sets? Furthermore, attempts to reconcile the assumptions of various models have also failed to make significant advances. We believe that one possible strategy to escape from the current dilemma can be to think out of the food web 'box.' Throughout the chapter, we used the analogy of elephant and blind man. What leads to the current impasses in diversity-stability, or community dynamics at large may be the combination of both factors (issues): (1) Food web is not the whole elephant (community), and some parts of the elephant have been blocked by the fence in the zoo; (2) With each of our current approaches, we can only touch a specific part of the elephant. To deal with the first issue, we need to enrich the paradigm of food webs, such as considering other relationship. Nature is not limited to bloody predation, and cooperation and communication are everywhere. From this perspective, perhaps replacing food web with a new concept such as SINs or simply community network may not be a bad idea. To address the second issue, we need to continue our efforts in developing more effective approaches for analyzing community dynamics. The combinative efforts from both fronts should help us to deepen our understanding of the community stability and dynamics.

3.2 Other Major Ecological Networks

Beyond food webs, ecologists have also performed extensive studies on other ecological networks, most notably, mutualistic networks and host-parasitoid networks (HPNs) (see Ings et al. [3] for a comprehensive review). Mutualistic networks or webs have been studied to reveal the nexus of ecosystem services such as pollination and seed dispersal, and their studies are less concerned with population dynamics or energy fluxes per se. Three mutualistic systems received most attention: pollination networks (plants-pollinators), frugivore networks (plants-seed-dispersers), and *ant–plant networks.* Microbes certainly play a role in the three mutualistic webs. For example, fungi are known to be an important party in many ant–plant networks. Nevertheless, currently, very few studies of mutualistic networks implicate microbes. This status, of course, by no means implies that mutualism is insignificant in microbial world. Contrarily, mutualism is obviously extremely important, not only between microbial species, but also between microbe–plant, microbes–animal interactions. Although pathogenic microbes undoubtedly have been the most vicious enemies of human beings and the focus of modern medical research, most microbes are neutral or beneficial to humans. There is no doubt that pathogens are in minority in the human microbiome, and most microbes inhabited in our bodies get along well with us. Commensalism, including mutualism should be equally ubiquitous and important in human–microbes interactions, if not more, than human–pathogen interactions. The on-going NIH (National Institute of Health) HMP in the USA and similar projects in EU initially focus on *cataloguing* of human microbes species via metagenomics technology. The US HIH has set the study of *microbial community dynamics* as an important research agenda in post-HMP era. Obviously, what may lead to significant breakthroughs in biomedicine should be the research on the interactions between the microbial community and human body, in which mutualism and co-evolution between human and microbes probably are the dominant mechanisms/forces. In particular, the paradigm of *super-microorganism* (microbial community) vs. *host* (human body) is not unlike the interactions between ants-colony (known as *superorganism* in insect sociobiology) and plant. As mentioned previously, the latter has been one of the three major types of mutualistic networks in current research on ecological networks. Therefore, we believe that future biomedical studies of human microbiome vs. human body interactions, or *super-microorganism vs. host*, can draw methods and insights from existing studies of ant–plant networks. At present, HMP has not generated sufficient datasets to construct microbiome network involving the host; nonetheless, the extensive datasets from HMP research do offer us an unprecedented opportunity to construct SINs within the human microbiome (HM). In the next section, we present a case study to demonstrate the construction and analysis of the human microbiome (HM) SIN, or HM-SIN, as well as their biomedical implications.

HPN analysis studies a special type of predator–prey relationship (upon which traditional food web networks are formulated), between parasitoids and their hosts. HPN focuses on distinct guilds of terrestrial insects (hosts) and traces the links from hosts to their parasitoids. On surface, the HPN appears has little to do with microbial networks. Nevertheless, we speculate that plasmids-bacteria should form networks similar to the HPN in insect world. Furthermore, hyper-parasitism or super-parasitism exists in HPN, where parasitoids are the hosts of other parasitoids, is not unlike the interactions between human (host), bacterial pathogens, and plasmids of the pathogens. From this perspective, the study of HPN networks may also be of important inspiration to biomedical research on host–pathogen interactions.

A recent trend in ecological network analysis is the cross-comparisons between different network types as well as a more mechanistic approach, rather than a phenomenological approach [3]. For instance, the identification of motifs and compartments may allow researchers to explore the similar role of co-evolution in shaping mutualistic networks and HPNs, as well as the role of body size in food webs. In addition, ecologists increasingly recognize the significance of individual traits and behaviors in ecological network analysis, because the ecological interactions ultimately happen between individuals. The new generation of networks with built-in individual traits and behaviors overcomes the shortage of networks constructed with species-averaged data, and opens a new paradigm of ecological network analysis, which is likely to mirror the success of the individual-based modeling (IBM) paradigm emerged in the 1990s. Still, recent network analysis also sheds some new light on pursuing one of the "lost" paradigms in ecology–diversity – stability relationships. A new postulation supported by recent network analysis suggests that it is the distribution of interaction strengths and the configuration of complexity, rather than just its magnitude that controls network organization and stability [3].

3.3 Network Analysis as a Powerful Tool for the Emerging Medical Ecology

According to the website source www.medicalecology.org, maintained by Dr. Dickson Despommier and Dr. Steven Chen of Columbia University, the term *Medical Ecology* was first coined by eminent microbiologist Rene Dubos, who believed that natural world could offer many of our needs if explored fully. Dubos was apparently inspired by events such as the discoveries of penicillin and gramicidin (his own discovery), in which soil microbes played a critical role, as well as the treatment of malaria with quinine. Recent redefinition of medical ecology has much broader meanings. For example, www.medicalecology.org defined "Medical Ecology is an emerging science that defines those aspects of the environment that have a direct bearing on human health. The concept of ecosystem functions and services helps to describe global processes that contribute to our well-being, helping to cleanse the air we breathe, the water we drink, and the food we eat. Environmental degradation often leads to alterations in these aspects, leading to various states of ill health." In the post-HGP (Human Genomic Project) and HMP era, Pepper and Rosenfeld [55] and Ma [56] were among the earliest who called for the expansion of the scope of medical ecology. Pepper and Rosenfeld [55] noted that several striking new empirical results from the study of human gut microbiome are puzzling within the standard conceptual framework of biomedicine, which highlights the need for new perspectives from ecology to explore the link between medicine and ecology. Ma [56] proposed a new framework for medical ecology centered on the human microbiome, and suggested a list of focal fields for the research in medical ecology. Figures 3 and 4 show the framework diagrams for a new medical ecology and its focal research fields originally developed in Ma [56], in which he argued that the study of the relationship between human microbiome and human host as well as its implications to human health and diseases should be a core component of newly expanded *medical ecology*. He further suggested that the emerging medical ecology is at the stage when *medical genetics* was emerging in the 1960s, and it should ultimately assume a similar role in medicine as today's medical genetics assumes.

Figure 3 shows the relationship between redefined medical ecology and its core parent fields, as well as its supporting and supported fields. Obviously, the topics of traditional medical ecology were omitted fully. Instead, medical ecology is depicted as a

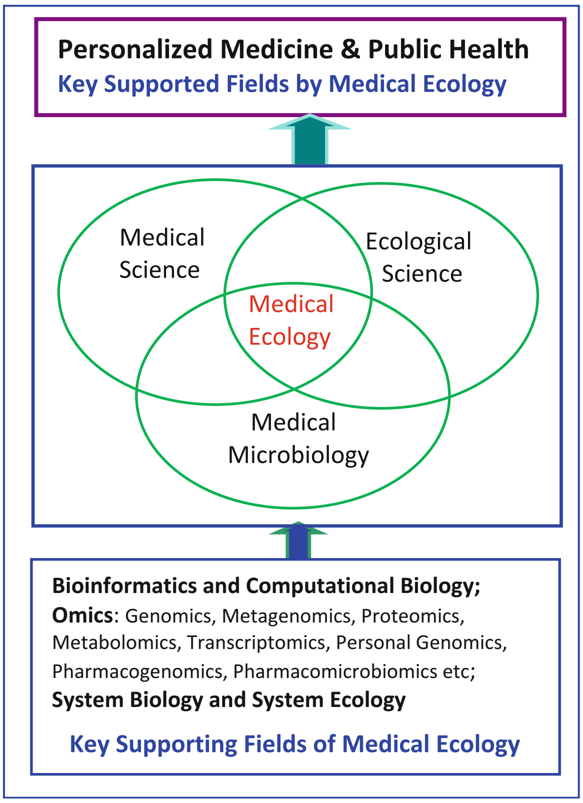
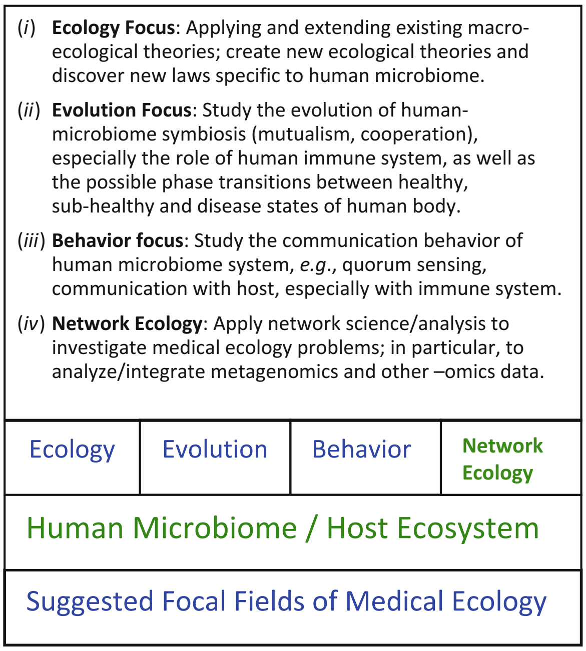

Personalized Medicine & Public Health
Key Supported Fields by Medical Ecology
Medical Science
Ecological Science
Medical Ecology
Medical Microbiology
Bioinformatics and Computational Biology;
Omics: Genomics, Metagenomics, Proteomics, Metabolomics, Transcriptomics, Personal Genomics, Pharmacogenomics, Pharmacomicrobiomics etc;
System Biology and System Ecology
Key Supporting Fields of Medical Ecology

Fig. 3 Medical ecology of the human microbiome as well as its parents, supporting and supported fields [56]

cross-disciplinary and trans-disciplinary subject with "native parents" of medicine, ecology, and microbiology. Its advances are directly dependent on various -*omics* technologies (in particular metagenomics), computational biology, bioinformatics, system biology (ecology), etc. It was also suggested that medical ecology should become a foundation of personalized medicine, and that personalized medicine is likely to be the biggest beneficiary of research in medical ecology. In Fig. 4, based on the tradition in macro-ecology, where ecology, evolution, and behavior are considered as three interwoven disciplines, a similar array of research fields for the emerging medical ecology was proposed. Those fields are: (1) applications and extensions of ecological theories from macro-ecology; (2) evolution of symbiosis, cooperation and mutualism of "human microbiome/host" relationships, especially the role of human immune system; (3) communication (signaling) behavior within the human microbiome and the communication (signaling) between microbiome and host; (4) network ecology. We emphasize *network ecology* because the potential applications of network science and/or analysis to medical ecology can be rather broad, from analyzing metagenomics data, integrating with metatranscriptome



(*i*) **Ecology Focus**: Applying and extending existing macro-ecological theories; create new ecological theories and discover new laws specific to human microbiome.

(*ii*) **Evolution Focus**: Study the evolution of human-microbiome symbiosis (mutualism, cooperation), especially the role of human immune system, as well as the possible phase transitions between healthy, sub-healthy and disease states of human body.

(*iii*) **Behavior focus**: Study the communication behavior of human microbiome system, *e.g*., quorum sensing, communication with host, especially with immune system.

(*iv*) **Network Ecology**: Apply network science/analysis to investigate medical ecology problems; in particular, to analyze/integrate metagenomics and other –omics data.

Ecology
Evolution
Behavior
Network Ecology
Human Microbiome / Host Ecosystem
Suggested Focal Fields of Medical Ecology

Fig. 4 Suggested focal fields of medical ecology [56]

and/or metaproteome data, to system biology/ecology modeling. Of course, the domain of medical ecology far exceeds what were listed in Fig. 4; nonetheless, we believe those are the most pressing but currently largely ignored topics for medical ecology, in particular from the perspective of personalized medicine.

4 A Case Study on the SIN of the Human Microbiome

4.1 Species Interaction Networks

In nature, biological species exist in the form of populations, which in turn form communities. A community and its habitat (including the environment) where the populations of member species inhabit is termed ecosystem. These are not rigorous definitions, but convenient for introducing the relationships (interactions) among species in a community. A simple illustration of the interactions among species can be made in pair-wise fashion with positive (+), negative (−), and neutral (0) representing three possible effects (outcomes) for each other. There are six possibilities in total, but the '0-0' interaction pattern is obviously of little interest to us because it indicates that both species are independent with each other or no interaction between them. Those other five interactions types are referred to as: mutualism (+ +), competition (− −), parasitism or predation (+ −), commensalism (+0), and amensalism (− 0) [9]. To construct a SIN, one needs species abundance data. In the case of microbial community, one can get the data with a range of technologies, ranging from cytometry by microarrays to rRNA marker gene sequencing and metagenomic sequencing [9, 57–61]. Specialized bioinformatics pipelines that depend on sequence clustering and reference databases to assign sequence reads to microbial OTU (Operational Taxonomic Unit) or taxa have been developed as open source software packages [62, 63]. The most common data types are the so-termed OTU tables from 16S-rRNA sequencing for bacterial, with each row representing the OTU (species or other taxa) abundances from one community sample, computed with Mothur [63] or QIIME [62].

After obtaining the OTU tables, which represent the species abundances (16S-rRNA reads) in the community sample, the next step is to infer species interaction relationships by performing *network inferences*, which include two general approaches. The first approach is to infer pair-wise correlation (similarity) relationship by computing various correlation coefficients, and the most frequently used are *Pearson product moment* correlation and *Spearman's rank-order* correlation. Two additional commonly used correlation and association measures are *mutual information*, which is derived from information theory and provides a general measure of dependencies, and *partial correlation coefficients*, which can measure the correlation between any pair of variables when a third (or more) specified variables are assumed to be constant [64]. The second network inference approach tries to decipher relationships involving multiple species, and the most commonly used approaches are the *regression analysis* and *rule-based data mining* techniques. The latter approach involves searching significant association roles from all logically possible rules. For example, a rule could be something like "in the absence of OTU *x* and the presence of OTU *y*, OTU *z* is also present" [9]. During the network inference step, one may also incorporate environmental factors or metadata in the case of human microbiome research by treating environmental variables as 'equivalents' of special OTUs. This allows us to investigate the influence of environmental factors on individual species as well as the overall network structure [9, 64].

The species interaction relationships inferred from the above step, in the form of correlation (similarity) matrix or association rules, can be fed into network visualization software packages such as Cytoscape [38], Osprey [65], VisANT [66], iGraph [42, 67], Gephi [68], Pajek (e.g., [69]), Tulip [94], or ORA [70], to produce network graphs visualizing the SINs. Most of these network analysis and visualization tools, besides producing network graphs, can also compute *network properties*, detect *network modules* (*communities*, or *motifs*), and perform *network model testing* against some important null models (such as scale-free network, small-world network, Erdos–Renyi random network, etc.). Examples of *network properties* include the previously defined local and global clustering coefficients [Eqs. (5) and (6)], but there are more including *network density*, *network modularity*, *number of communities*, *number of connected components*, *average path length*, *network diameter*, etc. Detecting network modules (communities, motifs) is usually a major functionality of network analysis software. There are many module (motif) detection tools, either standalone or as plug-ins for the above-mentioned network analysis tools, such as Mfinder [71], MAVisto [72], FANMOD [73], and Pajek [69]. Testing against standard network null models such as E-R random network [Eqs. (1) and (2)] and scale-free networks [Eqs. (3) and (4)] is important because the properties and evolution mechanisms of those null network models are usually well understood, and any deviation or congruence with the null models can reveal valuable insights for understanding the real-world network. An alternative way to test a network against a standard network model is to use the so-termed network alignment software, which compares two or more networks with some predefined criteria. Unfortunately, the network alignment problem is NP-hard in general. Existing software packages such as pathBLAST, NetworkBlast, NetAlign, IsoRankN, Craemlin, GRAAL, C-GRAAL (web links for these programs are indexed in the reference [87–93]), and more recently RESQUE [74], usually can only handle the comparisons of relatively small networks, and the number of networks that can be simultaneously aligned is very limited. Further research and development are needed in order to compare more complex real-world ecological networks.

All the discussion in this article is limited to *static* networks. The approaches we discuss may be utilized to analyze community dynamics data, but there are certainly some limitations. Faust and Raes [9] reviewed the usages of Lotka–Volterra differential equations (for species abundance data) and hypergeometric distribution (for presence–absence data) for modeling community dynamics. Strictly, these are still static network, and true dynamic or temporal networks involve the *evolution of network*. In true *dynamic* or *temporal* network, there is one static network (snapshot) for each time moment. There are subtle differences between the concepts of *temporal* network and *dynamic* network, and the field is still in its infancy. Similarly, there is the study of *spatial* networks, also known as *geometric graphs*, in which nodes or edges are spatial elements representing geometric objects and graph's topology is not sufficient to capture the full information. Mobile phone network is an example of temporal network, if the location information is ignored. However, it is also a spatial network if the location of mobile user is considered. In fact, it is a *spatiotemporal network*, and the theory of *spatiotemporal network* is still in development.

Before proceeding to present a case study, we should point out that network analysis has evolved into a rather broad cross-disciplinary field that has attracted extensive attentions from mathematicians, computer scientists, sociologists, biologists, and

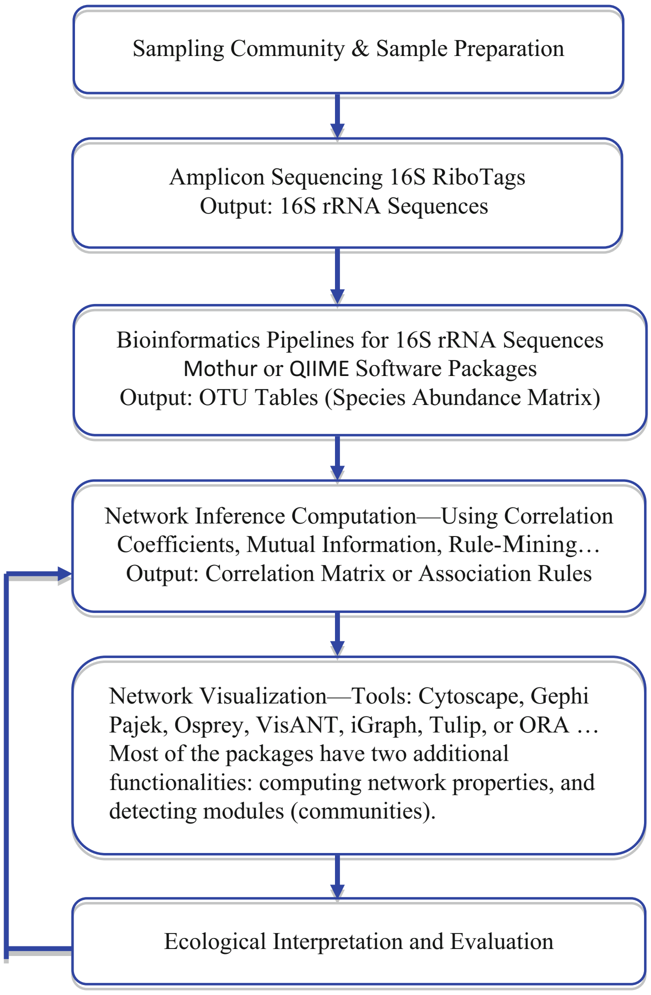

Sampling Community & Sample Preparation
Amplicon Sequencing 16S RiboTags
Output: 16S rRNA Sequences
Bioinformatics Pipelines for 16S rRNA Sequences
Mothur or QIIME Software Packages
Output: OTU Tables (Species Abundance Matrix)
Network Inference Computation—Using Correlation Coefficients, Mutual Information, Rule-Mining…
Output: Correlation Matrix or Association Rules
Network Visualization—Tools: Cytoscape, Gephi Pajek, Osprey, VisANT, iGraph, Tulip, or ORA …
Most of the packages have two additional functionalities: computing network properties, and detecting modules (communities).
Ecological Interpretation and Evaluation

Fig. 5 A flowchart of the bioinformatics and network analyses pipelines for microbial SINs

scientists and engineers from many other fields. Indeed, as argued previously, it has become a new *science of networks*, and has found applications in nearly every field of natural, social, and military sciences, as well as in engineering, medicine, and technology. Therefore, it is hardly possible to even touch the iceberg tip of the network science in this short article. For biologists (ecologists) with interests to acquire further information, we refer to the following excellent monographs [1, 64, 75–79].

Figure 5 is a diagram to demonstrate the bioinformatics and network analyses pipelines for performing network analysis with the human microbiome data. In the following subsection, a case study with the human oral microbiome is conducted to demonstrate the process.

4.2 A Case Study with the Human Oral Microbiome

The oral microbiome, primarily consists of several thousands of bacterial phylotypes inhabiting the human mouth, is of important biomedical implications to human oral health. It is known that periodontitis is caused by the proliferation of pathogenic bacteria within the mouth, and the inflammatory dental disease is also known as a risk factor for cardiovascular disease. The holistic relationship between the oral microbiome ecosystem and periodontitis was largely unknown until recently, although individual pathogens associated with the disease were much known. Liu et al. [80] recent study with the sequencing of both 16S rRNA marker gene and whole community DNA (metagenomics) have offered the first system-level glimpse of the global genetic, metabolic, and ecological changes associated with the disease. Their 16S rRNA gene analysis, including 11 healthy microbiome samples and 4 samples with periodontitis, suggested that the oral microbiome may experience a shift from a *gram-positive* dominated healthy bacterial community to a *gram-negative* dominated diseased community, and the latter is also enriched in a number of oral pathogens. They also found that the diseased community is generally more even with higher diversity. However, the molecular mechanism that leads to the shift remains unknown. In the following, we expand their study with the network analysis of their 16S rRNA datasets, and hope to shed light on the mechanism leading to the transition from healthy to diseased microbiome. The results from our network analyses are represented in Tables 1, 2, 3, and 4 as well as Figs. 6, 7, 8, and 9.

Table 1 listed 11 major network properties of both healthy and diseased oral microbiome SINs. First, it is obvious that there are significant differences between both the networks. For example, the *number of edges* in the healthy network is 3.2 times that in the diseased network, which indicates that the healthy network is much more strongly connected. According to the theory of ecological stability, the healthy community (network) should be more stable than the diseased network because of the high connectivity. The *number of nodes*, which correspond to the *species richness* in the community, is also significantly (1.8 times) more in the healthy network. The *degree* is one of the most basic properties of a network node, and it is defined as the number of edges adjacent (connected) to the node. The *average degree* reported in Table 1 is obtained by computing the *arithmetic average* of the degrees of all nodes in the network. It has been found that *degree* is indeed associated with some functions in biological networks. In more detailed network analysis, the degrees of all nodes are fitted to a statistical distribution such as the power law distribution or binomial distribution. As mentioned previously, while satisfying *binomial distribution* is a property of Erdos–Renyi random network, the *scale-free network* has a *power law* degree distribution. Table 3 shows the results of

Table 1
The properties of the species interaction networks (SINs) of the human oral microbiome

Sampling points	Number of nodes	Number of edges	Average degree	Avg. local cluster coefficient	Diameter	Average path length	Connected components	Network density	Network modularity	Number of communities	Ratio of positive to negative
Healthy	771	42,567	110.420	0.918	6	2.781	1	0.143	0.702	8	1.000 (2)
Diseased	428	13,188	61.626	0.820	11	4.525	1	0.144	0.635	5	0.967 (860)
H/D ratio	1.8	3.2	1.8	1.1	0.5	0.62	1	1	1.1	1.6	1.03

Table 2
The number of three- and four-node motifs in the SINs of the human oral microbiome

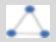
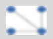
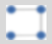
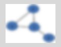
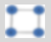
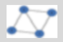
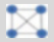
Sampling points	Three-motif type-I	Three-motif type-II	Four-motif type-I	Four-motif type-II	Four-motif type-III	Four-motif type-IV	Four-motif type-V	Four-motif type-VI	Global cluster coefficient
									
Healthy	321,151	1,866,866	267,931	8,990,976	15,629,571	9,548	5,917,030	71,516,927	0.946
Diseased	79,224	348,743	30,378	732,627	1,884,423	2,256	1,240,114	7,711,454	0.930
H/D ratio	4.1	5.4	8.8	12.3	8.3	4.2	4.8	9.3	1.02

Table 3
The *p*-value of fitting three degree-distributions and their parameters

Degree	Normal distribution	Poisson distribution	Power law distribution	Power law parameter (K)
Healthy	0.000	0.000	0.000	113.835
Diseased	0.000	0.000	1.000	74.197

Table 4
The most strongly connected *modules* in both healthy and diseased microbiome networks

Module no.	Score = Density × Nodes	Nodes	Edges
Healthy microbiome network			
1	96.08	195	18,736
2	56.70	233	13,210
3	38.77	132	5,117
4	22.79	140	3,190
5	14	29	406
6	8.90	19	169
7	7.31	16	117
Diseased microbiome network			
1	51.73	107	5,535
2	46.10	95	4,379
3	20.82	76	1,582
4	5.24	21	110
5	3.61	23	83
6	3.5	8	28
7	3.09	11	34
8	2.67	12	32
9	2	5	10
10	2	5	10

fitting three common degree-distributions, i.e., power law, Gaussian normal, and *Poisson* distribution (which can approximate the *binomial distribution*). It is shown that while the *degree* distribution of the healthy microbiome network does not follow any of the three distributions, the diseased network follows the power law distribution. The fact that the degree distribution of the diseased oral microbiome networks follows the power law suggests that they are most likely scale-free. Hence, periodontitis may have caused the microbiome network's shift to scale-free regime.

The network *diameter* is the maximal distance of any pair of the nodes in the network. The *average path length* or *characteristic path length* is defined as the average distance between all pairs of nodes in the network. The small-world network has a property that its

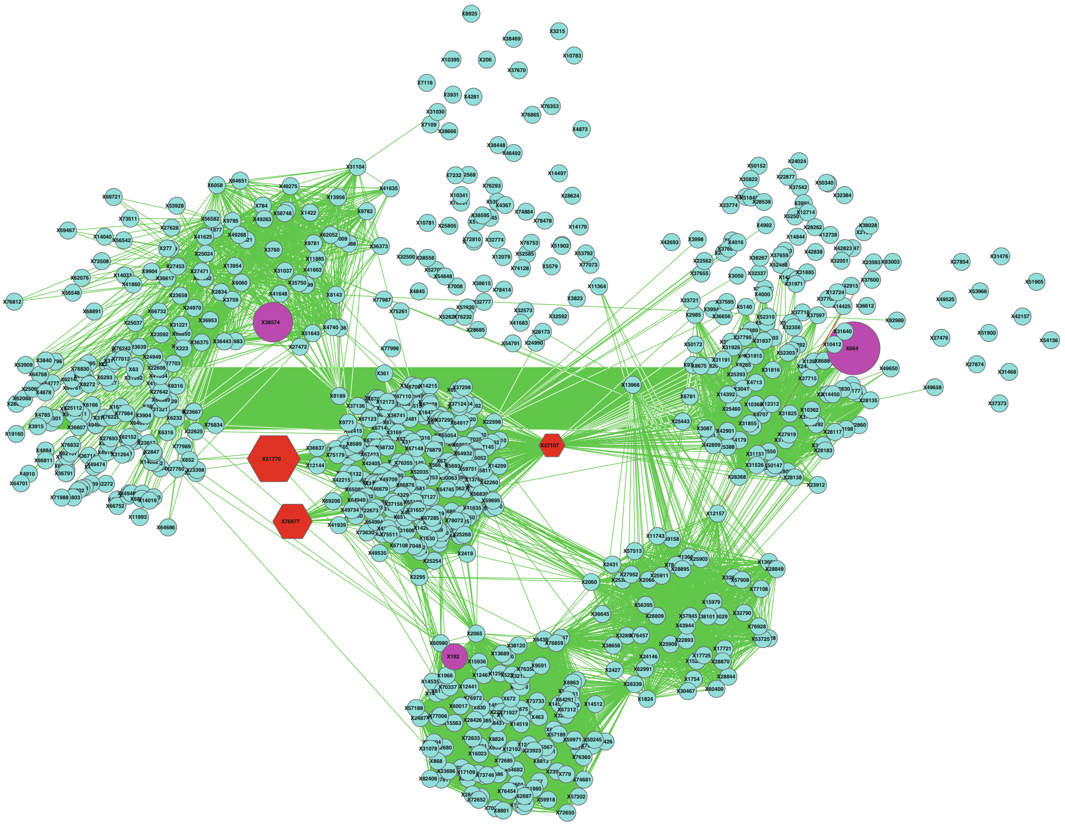


Fig. 6 The SIN network graph of the oral microbiome from healthy individuals

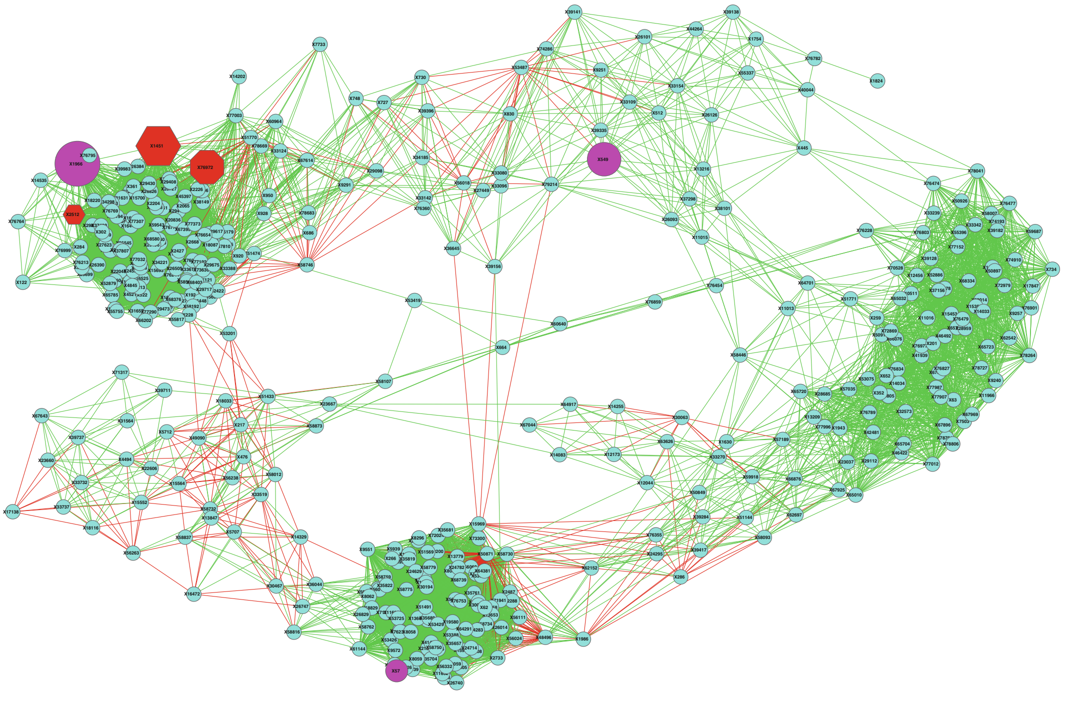
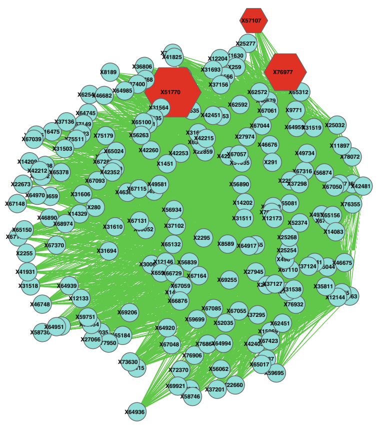
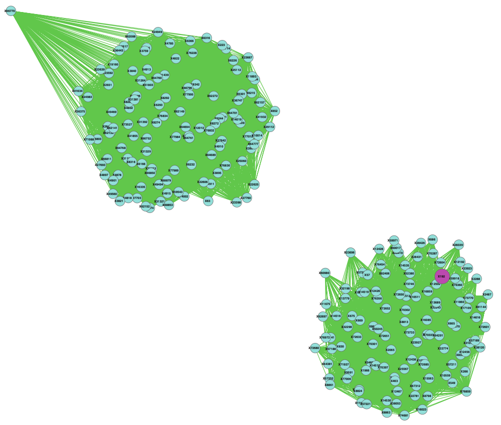


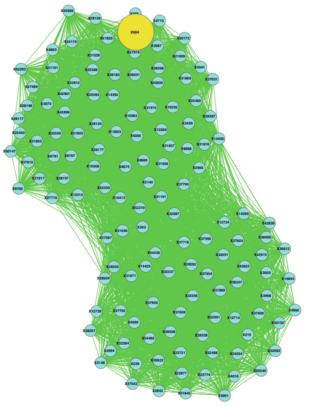
Fig. 7 The SIN network graph of the oral microbiome from individuals with periodontitis



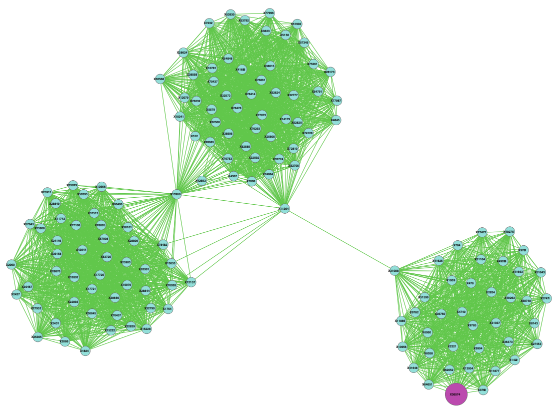
Module (1) in Healthy Microbiome



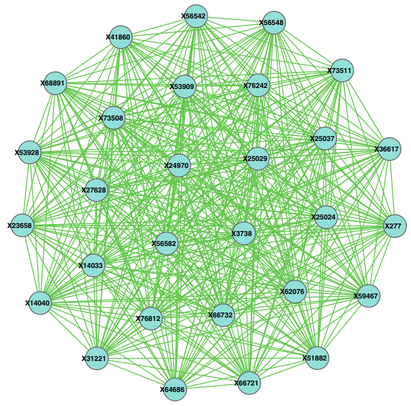
Module (2) in Healthy Microbiome



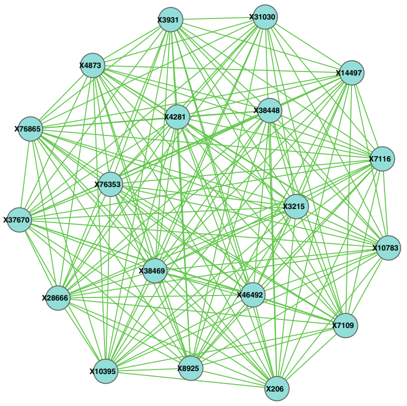
Module (3) in Healthy Microbiome



Module (4) in Healthy Microbiome

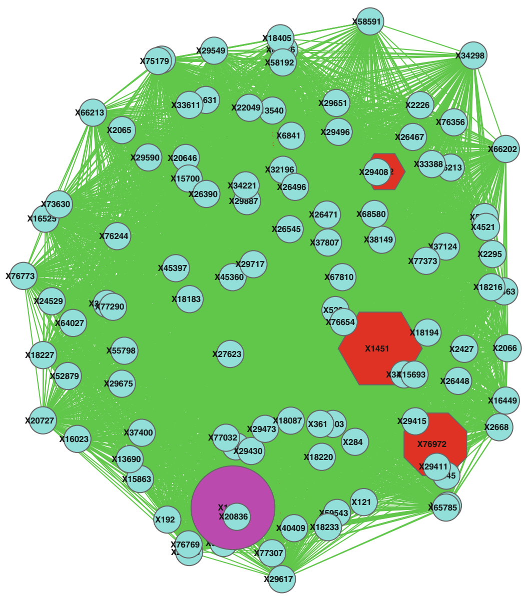


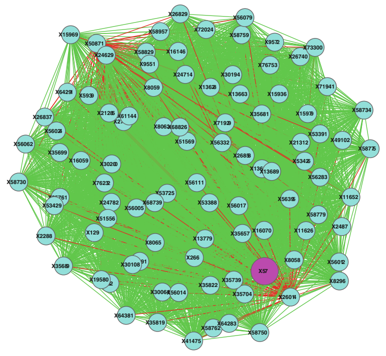
Module (5) in Healthy Microbiome

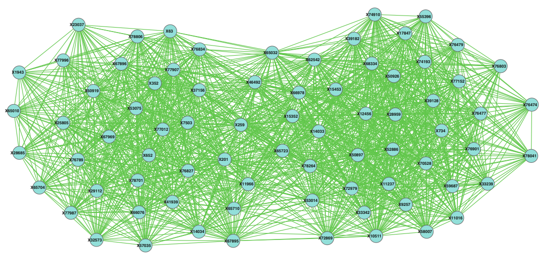


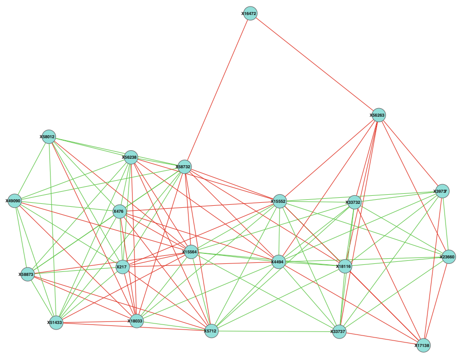
Module (6) in Healthy Microbiome

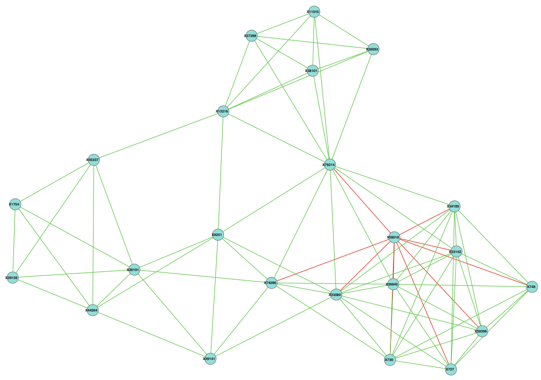
Fig. 8 The network modules detected in the SIN network of the oral microbiome from healthy subjects

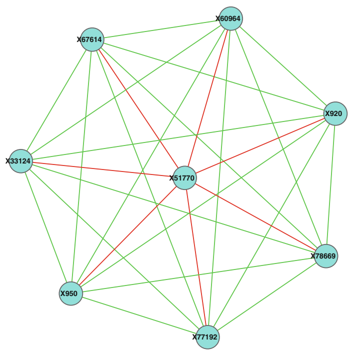

Module (1) in Healthy Microbiome


Module (2) in Healthy Microbiome


Module (3) in Healthy Microbiome


Module (4) in Healthy Microbiome


Module (5) in Healthy Microbiome



Module (6) in Healthy Microbiome

Fig. 9 The network modules detected in the SIN network of the oral microbiome from individuals with periodontitis

average path length (p) should be smaller than $\ln(N_v)$, i.e., $p < \ln(N_v)$, where N_v is the number of nodes in the network. In the case of the oral microbiome networks in Table 1, the inequality relationships are $2.8 < 6.6$ and $4.5 < 6.1$ for the healthy and diseased network, respectively. Therefore, both networks are most likely small-world networks. It should be noted that there are rigorous procedures for testing the *scale-free* or *small-world* networks; the power law and inequality properties are necessary but not sufficient conditions for judging the networks.

Network density measures how a network is densely populated with edges, and it has a value between 0 and 1. A network of totally isolated nodes has zero density and a clique (totally connected network) has the density of 1. Of course, network density is related to average degree, but the relationship is complex beyond simple positive correlation. *Network modularity* measures the level at which a network can be naturally divided into communities or modules (also known as cluster or groups). It also takes a value between 0 and 1. It is interesting to note that *network density* is actually one of the few properties that demonstrates nearly nil difference between the healthy and diseased microbiome networks.

The *local clustering coefficient* of a node n is defined as $C_n = 2e_n/k_n(k_n - 1)$, where k_n is the number of neighbors of n and e_n is the number of connected pairs between all neighbors of n. The *network-clustering coefficient* is the average of the clustering coefficients for all nodes in the network. It therefore measures the degree to which nodes in a network tend to cluster together, and is an indication of the *embeddedness* of single nodes. In contrast, the *global clustering coefficient* is defined as the three times the number of *type-2* triangle motifs divided by the number of *type-1* triangle motifs (see Table 2). In terms of both local and global clustering coefficients, the healthy network tends to be more clustered. Further mining of network modules (clusters) also confirmed this property of the healthy microbiome network (Table 4).

The last column in Table 1, the ratio of positive interaction links vs. the negative interaction links is actually not a '*formal*' network property previously defined. Nevertheless, we believe it should be equally important, if not more, than some of the most important network properties. The numbers in parentheses (i.e., healthy vs. diseased = 2 vs. 860) are more telling. That is, there are far more negative interactions in the diseased microbiome network than those in the healthy network. Obviously, this difference of over three magnitudes is the most remarkable among all properties we studied in this article. It is hardly possible to disassociate this dramatic change with the shift from healthy to diseased microbiome. We suggest that the rise of gram-negative bacteria, and their competitions with beneficial microbes, should be the cause of this abrupt rise of the negative interactions.

Table 2 lists the numbers of various three- and four-node motifs. The reason we limited the motif searching to four nodes is practical because finding arbitrary size motif is an NP-hard problem, which means that when the motif size is sufficiently large, the search may be too computationally time consuming to obtain all the motifs. Table 2 demonstrates that the numbers of various three and four-node motifs are 4–12 times more in the healthy microbiome network.

Figures 6 and 7 are the network graphs of the healthy and diseased microbiome networks, respectively. Some of the properties listed in Tables 1 and 2 may also be visually inspected in these two figures. For example, the healthy network is far more clustered, both locally and globally, showing more densely connected communities or clusters. In Figs. 6, 7, 8, and 9, the regular network nodes representing the OTUs other than the hubs and the most abundant OTUs are marked with small circle symbol in cyan color; the top three most connected OTU (hubs) are marked with hexagon symbol and the size of hexagon indicates the number of connections (i.e., degree); the top three most abundant OTUs are marked with larger circles in pink color. Also in these figures, the positive relationship (interaction) is marked with green line, and the negative relationship (interaction) is marked with red line. Each OTU is marked with its OTU number.

We used MCODE plug-in in the Cytoscape software [38] to systematically detect distinguishable modules and the results are listed in Table 4. Table 4 lists the number of nodes and edges of each module as well as their scores. The module *score* is defined as the number of nodes in the module multiplied by the module *density*, and it measures the strength of the internal interactions within the module. The higher the score is, and the stronger the internal interactions are. It is clear from Table 4 that the modules in the healthy network are more densely packed, indicated by much higher module scores. For example, the No. 1 module in the healthy network has a module score of 96.1, almost twice the score of its counterpart in the diseased network with a score of 51.7. Figures 8 and 9 show the top six highest scored modules in the healthy and diseased microbiome networks, respectively. The higher interaction densities in the healthy modules are obvious in these graphs. The distribution of negative interactions in the diseased modules can be readily identified in the module graphs. Among six displayed modules in the diseased network, four (No. 2, 4, 5, 6) contain negative interactions (*edges*). In contrast, the negative edges in the healthy modules are invisible, since there are only two edges in total and both are not classified into any modules. The differences between the healthy and diseased networks and their underlying communities are striking in these module graphs. We therefore have a plausible basis to conjecture that the transition from healthy regime to periodontitis regime is associated with the emergence of those negative interactions in the SIN of the human oral microbiome.

Acknowledgements

We thank Prof. Mihai Pop (University of Maryland) for generously providing us the OTU tables of the oral microbiome used for the case study. This research received funding from the following sources: National Science Foundation of China (Grants No. 61175071 & No. 71473243). "The Exceptional Scientists Program and Top Oversea Scholars Program" of Yunnan Province, "The Innovative Research Team on the Synthesis of Natural Evolution and Computational Evolution."

References

1. Pascual M, Dunne JA (2006) Ecological networks: linking structure to dynamics in food webs. Oxford University Press, New York
2. Montoya JM, Pimm SL, Solé RV (2006) Ecological networks and their fragility. Nature 442:259–264
3. Ings TC, Montoya JM, Bascompte J et al (2009) Ecological networks – beyond food webs. J Anim Ecol 78:253–269
4. Bastolla U, Fortuna MA, Pascual-García A et al (2009) The architecture of mutualistic networks minimizes competition and increases biodiversity. Nature 458:1018–1020
5. Bascompte J (2010) Structure and dynamics of ecological networks. Science 329:765–766
6. Greenblum S, Turnbaugh PJ, Borenstein E (2012) Metagenomic systems biology of the human gut microbiome reveals topological shifts associated with obesity and inflammatory bowel disease. Proc Natl Acad Sci U S A 109:594–599
7. Smillie CS, Smith MB, Friedman J et al (2011) Ecology drives a global network of gene exchange connecting the human microbiome. Nature 480:241–244. doi:10.1038/nature10571
8. Pocock MJO, Evans DM, Memmott J (2012) The robustness and restoration of a network of ecological networks. Science 335:973–977
9. Faust K, Raes J (2012) Microbial interactions: from networks to models. Nat Rev Microbiol 10:538–550
10. Faust K, Sathirapongsasuti JF, Izard J et al (2012) Microbial co-occurrence relationships in the human microbiome. PLoS Comput Biol 8, e1002606
11. Suweis S, Simini F, Banavar JR, Maritan A (2013) Emergence of structural and dynamical properties of ecological mutualistic networks. Nature 500:449–452
12. Heleno R, Garcia C, Jordano P et al (2014) Ecological networks: delving into the architecture of biodiversity. Biol Lett 10:20131000. doi:10.1098/rsbl.2013.1000
13. Tung J, Barreiro LB, Burns MB et al (2015) Social networks predict gut microbiome composition in wild baboons. eLife. doi:10.7554/eLife.05224
14. Sam Ma Z, Guan Q, Ye C et al (2015) Network analysis suggests a potentially "evil" alliance of opportunistic pathogens inhibited by a cooperative network in human milk bacterial communities. Sci Rep 5:8275. doi:10.1038/srep08275
15. Elton CS (1927) Animal ecology. Sidwich & Jackson, London
16. Lindeman RL (1991) The trophic-dynamic aspect of ecology. Bull Math Biol 53:167–191
17. May RM (2001) Stability and complexity in model ecosystems. Princeton University Press, Princeton, NJ
18. May RM (1983) Ecology: the structure of food webs. Nature 301:566–568. doi:10.1038/301566a0
19. Cohen JE, Newman CM (1985) A stochastic theory of community food webs: I. Models and aggregated data. Proc R Soc Lond Ser B, containing papers of a biological character Royal Society (Great Britain) 224:421–448
20. Cohen JE (1990) A stochastic theory of community food webs. VI. Heterogeneous alternatives to the cascade model. Theor Popul Biol 37:55–90

21. Turnbaugh PJ, Ley RE, Hamady M et al (2007) The human microbiome project. Nature 449:804–810
22. Human Microbiome Project Consortium (2012) A framework for human microbiome research. Nature 486:215–221
23. Human Microbiome Project Consortium (2012) Structure, function and diversity of the healthy human microbiome. Nature 486:207–214. doi:10.1038/nature11234
24. Gilbert JA, Meyer F, Antonopoulos D et al (2010) Meeting report: the terabase metagenomics workshop and the vision of an earth microbiome project. Stand Genomic Sci 3:243–248
25. Gilbert JA, O'Dor R, King N, Vogel TM (2011) The importance of metagenomic surveys to microbial ecology: or why Darwin would have been a metagenomic scientist. Microb Inf Exp 1:5. doi:10.1186/2042-5783-1-5
26. Prosser JI, Bohannan BJM, Curtis TP et al (2007) The role of ecological theory in microbial ecology. Nat Rev Microbiol 5:384–392. doi:10.1038/nrmicro1643
27. Costello EK, Stagaman K, Dethlefsen L et al (2012) The application of ecological theory toward an understanding of the human microbiome. Science 336:1255–1262. doi:10.1126/science.1224203
28. Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Appl Environ Microbiol 71:8228–8235. doi:10.1128/AEM.71.12.8228-8235.2005
29. Erdos P, Renyi A (1959) On random graphs. Publicationes Mathematicae, Debrecen
30. Erdős P, Rényi A (1960) On the evolution of random graphs. In: Publication of the Mathematical Institute of the Hungarian Academy of Sciences. pp 17–61
31. Bondy J, Murty U (1976) Graph theory with applications. Elsevier Science Ltd/North-Holland, New York
32. Newman MEJ, Watts DJ, Strogatz SH (2002) Random graph models of social networks. Proc Natl Acad Sci U S A 99:2566–2572. doi:10.1073/pnas.012582999
33. Barabási A, Albert R (1999) Emergence of scaling in random networks. Science 286 (5439):509–512
34. Albert R, Barabási A (2002) Statistical mechanics of complex networks. Rev Mod Phys 74:47–97
35. Schwöbbermeyer H (2008) Network motifs. In: Junker BH, Schreiber F (eds) Analysis of biological networks. Wiley, Hoboken, NJ, pp 85–111
36. Watts DJ, Strogatz SH (1998) Collective dynamics of "small-world" networks. Nature 393:440–442. doi:10.1038/30918
37. Taylor LR (1961) Aggregation, variance and the mean. Nature 189:732–735. doi:10.1038/189732a0
38. Shannon P, Markiel A, Ozier O et al (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Res 13:2498–2504
39. Gentleman RC, Carey VJ, Bates DM et al (2004) Bioconductor: open software development for computational biology and bioinformatics. Genome Biol 5:R80
40. Maere S, Heymans K, Kuiper M (2005) BiNGO: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks. Bioinformatics 21:3448–3449. doi:10.1093/bioinformatics/bti551
41. Taylor RC, Shah A, Treatman C, Blevins M (2006) SEBINI: Software Environment for Biological Network Inference. Bioinformatics 22:2706–2708
42. Csardi G, Nepusz T (2005) The Igraph software package for complex network research. I J Complex Sys (5):1–9
43. Hagberg A, Schult D, Swart P (2008) Exploring network structure, dynamics, and function using NetworkX. No. LA-UR-08-05495, Los Alamos National Laboratory (LANL)
44. Thomas S, Bonchev D (2010) A survey of current software for network analysis in molecular biology. Hum Genomics 4:353–360. doi:10.1186/1479-7364-4-5-353
45. Durrett R (2006) Random graph dynamics. Cambridge University Press, Cambridge
46. Stouffer D (2010) Scaling from individuals to networks in food webs. Functional Ecology 24:44–51
47. Lotka AJ (1925) Elements of physical biology. Williams & Wilkins, Baltimore
48. Volterra V (1926) Variazioni e fluttuazioni del numero d'individui in specie animali conviventi. Mem R Accad Naz dei Lincei Ser VI 2:31–113
49. Gaedke U (2008) Ecological networks. In: Junker BH, Schreiber F (eds) Analysis of biological networks. Wiley, Hoboken, NJ, pp 283–304
50. Dunne JA, Williams RJ, Martinez ND (2002) Food-web structure and network theory: the role of connectance and size. Proc Natl Acad Sci U S A 99:12917–12922. doi:10.1073/pnas.192407699

51. Dunne JA, Williams RJ, Martinez ND (2002) Network structure and biodiversity loss in food webs: robustness increases with connectance. Ecol Lett 5:558–567
52. Dunne JA (2005) The network structure of food webs. In: Pascual M, Dunne JA (eds) Ecological networks: linking structure to dynamics in food webs. Oxford University Press, Oxford
53. Ma Z, Krings AW (2011) Dynamic hybrid fault modeling and extended evolutionary game theory for reliability, survivability and fault tolerance analyses. IEEE Trans Reliab 60:180–196. doi:10.1109/TR.2011.2104997
54. MacArthur R (1955) Fluctuations of animal populations and a measure of community stability. Ecology 36:533–536. doi:10.2307/1929601
55. Pepper JW, Rosenfeld S (2012) The emerging medical ecology of the human gut microbiome. Trends Ecol Evol 27:381–384
56. Ma ZS (2012) A note on extending Taylor's power law for characterizing human microbial communities: inspiration from comparative studies on the distribution patterns of insects and galaxies, and as a case study for medical ecology. http://adsabs.harvard.edu/abs/2012arXiv1205.3504M
57. Palmer C, Bik EM, DiGiulio DB et al (2007) Development of the human infant intestinal microbiota. PLoS Biol 5, e177
58. Markowitz VM, Ivanova NN, Szeto E et al (2008) IMG/M: a data management and analysis system for metagenomes. Nucleic Acids Res 36:D534–D538. doi:10.1093/nar/gkm869
59. Glass EM, Wilkening J, Wilke A et al (2010) Using the metagenomics RAST server (MG-RAST) for analyzing shotgun metagenomes. Cold Spring Harb Protoc 2010. doi:10.1101/pdb.prot5368
60. Wooley JC, Godzik A, Friedberg I (2010) A primer on metagenomics. PLoS Comput Biol. doi:10.1371/journal.pcbi.1000667
61. Scholz MB, Lo C-C, Chain PS (2012) Next generation sequencing and bioinformatic bottlenecks: the current state of metagenomic data analysis. Curr Opin Biotechnol 23:9–15
62. Caporaso JG, Kuczynski J, Stombaugh J et al (2010) QIIME allows analysis of high-throughput community sequencing data. Nat Methods 7:335–336. doi:10.1038/nmeth.f.303
63. Schloss PD, Westcott SL, Ryabin T et al (2009) Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl Environ Microbiol 75:7537. doi:10.1128/AEM.01541-09
64. Junker BH, Schreiber F (2008) Analysis of biological networks. Wiley-InterScience, Hoboken, NJ
65. Breitkreutz B-J, Stark C, Tyers M (2003) Osprey: a network visualization system. Genome Biol 4:R22
66. Hu Z, Mellor J, Wu J et al (2005) VisANT: data-integrating visual framework for biological networks and modules. Nucleic Acids Res 33:W352
67. igraph – Network analysis software. http://igraph.org/. Accessed 13 Nov 2015
68. Bastian M, Heymann S, Jacomy M (2009) Gephi: an open source software for exploring and manipulating networks. Proceedings of international Association for the Advancement of Artificial Intelligence (www.aaai.org). conference on weblogs and social media
69. Brusco M, Doreian P, Mrvar A, Steinley D (2011) Two algorithms for relaxed structural balance partitioning: linking theory, models, and data to understand social network phenomena. Sociol Methods Res 40:57–87. doi:10.1177/0049124110384947
70. Carley KM (2014) ORA: a toolkit for dynamic network analysis and visualization. In: Rokne PJ, Alhajj PR (eds) Encyclopedia of social network analysis and mining. Springer, New York, pp 1219–1228
71. Kashtan N, Itzkovitz S, Milo R, Alon U (2002) Mfinder tool guide. Technical report, Department of Molecular Cell Biology and Computer Science & Applied Mathematics, Weizman Institute of Science
72. Schreiber F, Schwöbbermeyer H (2005) MAVisto: a tool for the exploration of network motifs. Bioinformatics 21:3572–3574. doi:10.1093/bioinformatics/bti556
73. Wernicke S, Rasche F (2015) FANMOD: a tool for fast network motif detection. http://bioinformatics.oxfordjournals.org. Accessed 13 Nov 2015
74. Sahraeian SME, Yoon B-J (2012) RESQUE: network reduction using semi-Markov random walk scores for efficient querying of biological networks. Bioinformatics 28:2129–2136. doi:10.1093/bioinformatics/bts341
75. Kepes F (2007) Biological networks. World Scientific, Singapore
76. Butenko S et al (2009) Clustering challenges in biological networks. World Scientific, Singapore
77. Dehmer M, Emmert-Streib F (2009) Analysis of complex networks: from biology to linguistics. Wiley-VCH Verlag, Weinheim

78. Networks: an introduction. http://www-personal.umich.edu/~mejn/networks-an-introduction/. Accessed 13 Nov 2015
79. Cagney G, Emili A (2011) Network biology: methods and applications. Humana Press, New York, NY
80. Liu B, Faller LL, Klitgord N et al (2012) Deep sequencing of the oral microbiome reveals signatures of periodontal disease. PLoS One. doi:10.1371/journal.pone.0037919
81. Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci U S A 99:7821–7826
82. Newman MEJ (2003) The structure and function of complex networks. SIAM Rev 45 (2):167–256
83. Newman MEJ (2010) Networks: an introduction. Oxford University Press, Oxford
84. Newman MEJ, Barabási AL, Watts DJ (eds) (2006) The structure and dynamics of networks. Princeton University Press, Princeton, NJ
85. Barabási AL, Gulbahce N, Loscalzo J (2011) Network medicine: a network-based approach to human disease. Nat Rev Genet 12:56–68
86. Barabási AL (2013) Network science. Philos Trans R Soc 371:1–3
87. PathBlast. http://www.pathblast.org/
88. NetworkBlast. http://www.cs.tau.ac.il/~bnet/networkblast.htm
89. NetAlign. http://netalign.ustc.edu.cn/NetAlign/
90. NetAlign. http://netalign.ustc.edu.cn/NetAlign/
91. IsoRankN. http://groups.csail.mit.edu/cb/mna/
92. Craemlin. http://omictools.com/graemlin-s5589.html
93. C-GRAAL. http://bio-nets.doc.ic.ac.uk/home/software/c-graal/
94. Tulip Website. http://tulip.labri.fr/TulipDrupal/?q=tutorials. Accessed 2 Sept 2015

Statistical Tools for Data Analysis

Sean M. Gibbons

Abstract

Microbial communities are complex and so are the data we use to describe them. In this chapter, we analyze a 16S amplicon data set from a marine time series using the open source QIIME software package. We first summarize complex, multivariate community composition data using ordination techniques. We then use the insights gained from ordination and multivariate statistical techniques to characterize the dominant relationships between environmental parameters and the microbial community in this marine ecosystem. Finally, we identify a list of taxa that show significant changes in relative abundance across seasons and describe the relationship between seasonality and community diversity. We go over several data visualization techniques that allow us to interpret our results. Analysis notes, QIIME commands, and sample data are provided. Finally, we discuss caveats regarding correlations and relative abundance data and describe how the statistical tools used in our 16S amplicon example can be applied to other types of "omics" data sets.

Keywords: Amplicon sequencing, Metadata, Molecular ecology, Multivariate statistics, Nonparametric, Ordination

1 Introduction

Early work in microbial ecology focused on morphology and culturing to characterize microbial communities. However, we know today that these techniques vastly underestimate microbial diversity in the environment (1). Modern molecular techniques, like phospholipid fatty acid analysis (PLFA), denaturing gradient gel electrophoresis (DGGE), and terminal restriction fragment length polymorphism (TRFLP), have provided a culture-independent view into microbial ecosystems, while sequencing techniques, such a Sanger sequencing, allowed for the investigation of the phylogenetic diversity of cultured isolates (2–5). The merging of molecular identification techniques with phylogenetic information was achieved with the advent of next-generation sequencing and has resulted in a revolution in the fields of molecular and microbial ecology (6, 7). As technologies have developed, data sets have grown very large, pushing microbial ecologists toward bioinformatics and multivariate statistics (8, 9). This chapter will be a practical introduction to statistical and computational tools used for the analysis and visualization of complex microbial ecology data.

2 Methods

In this chapter, we walk through an example analysis of a 16S amplicon data set from the English Channel (10). These data are available under study #1240 on the Earth Microbiome Project (11) data portal (http://microbio.me/emp) and are also available here, (https://dl.dropboxusercontent.com/u/68839641/HHLM_stats_chapter_data_gibbons.tgz) along with additional files and notes. The data include a closed-reference (Greengenes reference database) operational taxonomic unit (OTU; equivalent to "phylotype") abundance matrix (12, 13), a metadata mapping file, and a demultiplexed fasta file (data used to build OTU matrix). In this analysis, we skip the upstream data processing steps and use the provided closed-reference OTU abundance matrix and mapping files. All analyses are performed using the open source Quantitative Insights into Microbial Ecology (QIIME) v. 1.8.0 analysis package, in addition to the vegan package in R v. 3.1.2 (14, 15). QIIME installation instructions for multiple operating systems can be found at http://qiime.org. All QIIME commands provided in this chapter can be run from within the data folder that is provided in the link above. Otherwise, the user will need to modify the file paths. Output files and directories are designated by the "-o" argument in each command. See the QIIME documentation for detailed description of other command arguments. Installation instructions for R can be found at http://www.r-project.org/.

2.1 Anatomy of a Data Set

As scientists, we observe the world around us and try to construct models to make predictions about what will occur in the future. Perhaps the simplest example of such a model is the "particle in a box" from classical mechanics. In order to faithfully describe the behavior of a single particle in a box, you need to know the particle's position vector at some time t (three spatial coordinates – x, y, and z) and the particle's momentum vector (mass × velocity; velocity has x, y, and z components). Thus, with only six numbers to describe position and momentum (six-dimensional system), you could exactly predict the particle's behavior well into the future. Of course, the real world is a lot messier than the particle in a box, and we often do not know the number of dimensions that are required to build a useful model. In addition, we must struggle with the limited precision of our measurements and assume that our observations are drawn from a statistical distribution in order to quantify our error estimate.

Ecological systems are notoriously noisy and complex (16), and we are far away from building reliable, quantitative models (17). Current work focuses primarily on describing patterns in ecological survey data in order to construct a qualitative picture for how microbial communities are correlated with various factors in their environment, with an eye toward designing experiments that help build causal links between the biotic and abiotic components of ecosystems. Most community ecology studies include data on the biotic community and on the physicochemical properties of the environment. In the case of amplicon data or shotgun metagenomics, the biotic data is the sequence information, which is binned or annotated to produce an abundance matrix of genes or phylotypes. These matrices are often sparse, with a few abundant/ubiquitous phylotypes and many rare phylotypes that appear only in a small subset of samples. The distribution patterns of many phylotypes are often redundant, and these similarly behaved taxa can be grouped together to explain large swaths of the total community variance. Environmental metadata is often collected in parallel for each sample that is sequenced (e.g., geographic location, temperature, pH, nitrate concentration, etc.). In principle, the abundance of any given phylotype is causally influenced by many environmental parameters. However, it is often the case that only a small subset of *dominant* variables is responsible for explaining the vast majority of the community variation.

With thousands of phylotypes and dozens of environmental variables in a typical data set, it is impossible to consider the data in aggregate. However, as stated above, we can often reduce the complexity of our community data and metadata by looking for correlated behaviors. Therefore, in order to begin our statistical exploration of a marine microbial community (10), we need to order and classify our samples based on dominant trends to reduce the apparent dimensionality of our system.

2.2 Data Normalization

Failure to normalize the abundance matrix is known to create unwanted artifacts, so this is a crucial first step (18). In this analysis we normalize our OTU abundance table so that the same number of observations is recorded for each sample, although there are alternative normalization techniques that can be employed (19). We will use the single_rarefaction.py script in QIIME. A quick look at the OTU table shows that most samples contain more than 6,200 sequences. Therefore, we randomly draw 6,200 sequences from each sample, without replacement, to generate a rarefied OTU table. See scripts below for summarizing and rarefying the OTU table:

1. [summarize OTU table] Summarizes the contents of an OTU table in human-readable format: <biom summarize-table -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table.biom -o study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_ summary.txt>
2. [normalize data] Randomly subsamples with replacement to achieve an even number of sequences across samples: <single_rarefaction.py -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table.biom -d 6200 -o study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_d6200.biom>

For your own analyses, you may want to ask whether or not subsampling your data gives you reproducible results. This can be done using the jackknifed_beta_diversity.py script in QIIME, which randomly subsamples your data multiple times (multiple rarefaction at the same sampling depth) and plots the resulting variance in your beta-diversity metric (20).

2.3 Data Transformation

Now that we have a normalized data matrix, we can discuss transformations. It is possible to run ordination analyses directly on raw abundance data, but this is often not the optimal approach (21). For example, the sparseness of the data matrix results in many shared zeros between samples. An organism could be absent from two sites for many non-biological reasons (e.g., stochastic sampling effects), and it is usually not useful to consider this information to be ecologically relevant. Thus, many distance transformations down-weight shared absences (e.g., Hellinger distance). These transformations also tend to significantly reduce the nonlinearity of the data (e.g., log-transforms), which allows for the use of linear ordination methods, like principal component analysis (PCA) (21). Thus, using abundance-weighted distance transformations (e.g., Bray–Curtis, Hellinger, or weighted UniFrac) is an essential first step to any beta-diversity analysis (7, 21). For the English Channel analysis, we can generate distance matrices with the beta_diversity_through_plots.py command in QIIME. The default output of this command is weighted and unweighted UniFrac distances. However, we have not generated the required phylogenetic tree for our samples to calculate UniFrac distances, so we will use Bray–Curtis and Hellinger distances instead by passing a parameter file. The script and parameter file content are shown below:

1. [beta-diversity script] Generates beta-diversity distance matrices, principal coordinates, and 3-dimensional (3D) interactive plots: <beta_diversity_through_plots.py -i

study_1240_split_library_ seqs_and_mapping/study_1240_closed_reference_otu_table_d6200.biom -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file.txt -p beta_params_110714.txt -o bdiv_study_1240_d6200_bray_hellinger>

2. [contents of parameters file] The following lines should be added to a text file and passed after the "-p" argument in the prior command:

```
beta_diversity:metrics hellinger,bray_curtis
make_emperor:add_unique_columns True
make_emperor:ignore_missing_samples True
```

2.4 Unconstrained Ordination

There are numerous ordination methods for ordering and classifying multivariate data, which allow for visualization and quantification of dominant trends in the data. Many of these methods fall into the category of eigen-analyses or spectral decomposition, such as PCA and correspondence analysis (CA) (19). These methods yield a small number of orthogonal axes that account for dominant trends in the community variance, given assumptions of linearity or unimodality (for PCA or CA, respectively) (8, 19). Other methods, like Kruskal's nonmetric multidimensional scaling (NMDS), also generate a set of axes, but use an assumption-free heuristic that is not guaranteed to converge on the optimal solution (22). I will not review all of these techniques, but I refer you to an excellent paper by Albin Ramette (2007), which gives a practical overview of these methods and their application to microbial ecology (8), and to a recent work that uses manifold learning techniques to unravel highly complex nonlinear patterns in "omics" data (23, 24). Here, we will focus on metric and nonmetric multidimensional scaling techniques, in particular, principal coordinate analysis (PCoA) and NMDS (22, 25). PCoA is very similar to a PCA, except that the raw data has been converted into a distance metric (see prior section) before running a PCA. The PCA simply positions a set of orthogonal axes so that they span the largest ranges of community variance (i.e., the first axis spans the largest 1D variance range, the second axis spans the second largest 1D variance range that is orthogonal to the first axis, and so on). The PCoA plots are generated by the beta_diversity_through_plots.py script and can be accessed in the output folder from the previous section. PCoA assumes linearity, which is often a faulty assumption, especially when the total community variance is large. However, it is usually a decent approximation when using distance transformations (21). In order to check whether nonlinear effects obscure relevant patterns in the data, we calculate NMDS axes as well. Scripts for generating NMDS coordinates in QIIME can be found below:

1. [nmds coordinates] Generates NMDS coordinates from the Hellinger distance matrix: <nmds.py -i bdiv_study_1240_d6200_bray_hellinger/hellinger_dm.txt -o bdiv_study_1240_d6200_bray_hellinger/nmds_hellinger>

2. [plot nmds coordinates] Generates a 3D interactive plot using NMDS coordinates; note that you need to hack the output file from the prior script to look more like the PCoA coordinate output by replacing the first line with "pc vector number 1 2 3" and the last line with "% variation explained 30 20 10" (numbers don't matter, as long as they collectively add up to >50). Alternatively, you could plot these coordinates using a different software package (e.g., in R): <make_emperor.py -i bdiv_study_1240_d6200_bray_hellinger/nmds_hellinger_reformat.txt -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file.txt -o bdiv_study_1240_d6200_bray_hellinger/nmds_hellinger_emperor –ignore_missing_samples –add_unique_columns>
3. [plot first two PC coordinates for hellinger distance along with time] Creates an interactive 3D PCoA plot with time as the x-axis and the first two PC axes as the y- and z-axes: <make_emperor.py -i bdiv_study_1240_d6200_bray_hellinger/hellinger_pc.txt -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -o bdiv_study_1240_d6200_bray_hellinger/hellinger_emperor_time –ignore_missing_samples –add_unique_columns -a DAYS_SINCE_EPOCH>

Both ordination techniques reveal a striking pattern of seasonality in the data. Samples separated by a year, but taken during the same season, were much more similar to one another than to samples taken a few months apart (Fig. 1). The PCoA appears to do a slightly better job of capturing the seasonality of the data. Further inspection of the interactive Emperor (26) plots (these can be manipulated in a web browser by opening the .html files from the beta-diversity output) shows that there appear to be relationships between community structure and several metadata parameters (e.g., nitrate, salinity, silicate, phytoplankton abundance, etc.). In addition, there is coupling between seasonality and several of these environmental variables (e.g., nitrate in Fig. 2). Combining ordination with multivariate statistical techniques will allow us to identify the dominant environmental drivers of community composition. In the next section, we will discuss these statistical techniques.

2.5 Multivariate Statistical Analysis

Now that we have a better handle on the overall structure of our data set, we can start to ask specific questions using nonparametric statistics (8, 27). The difference between parametric and nonparametric statistics is that parametric statistics assume a particular probability distribution (28). However, we are usually ignorant of the form of the probability distribution for "omics" data, and it is safer to use assumption-free methods (i.e., nonparametric statistics), despite the fact that these methods give us less statistical power. Due to the strong seasonal signal we see in the PCoA, we will ask whether or not overall community structure is significantly

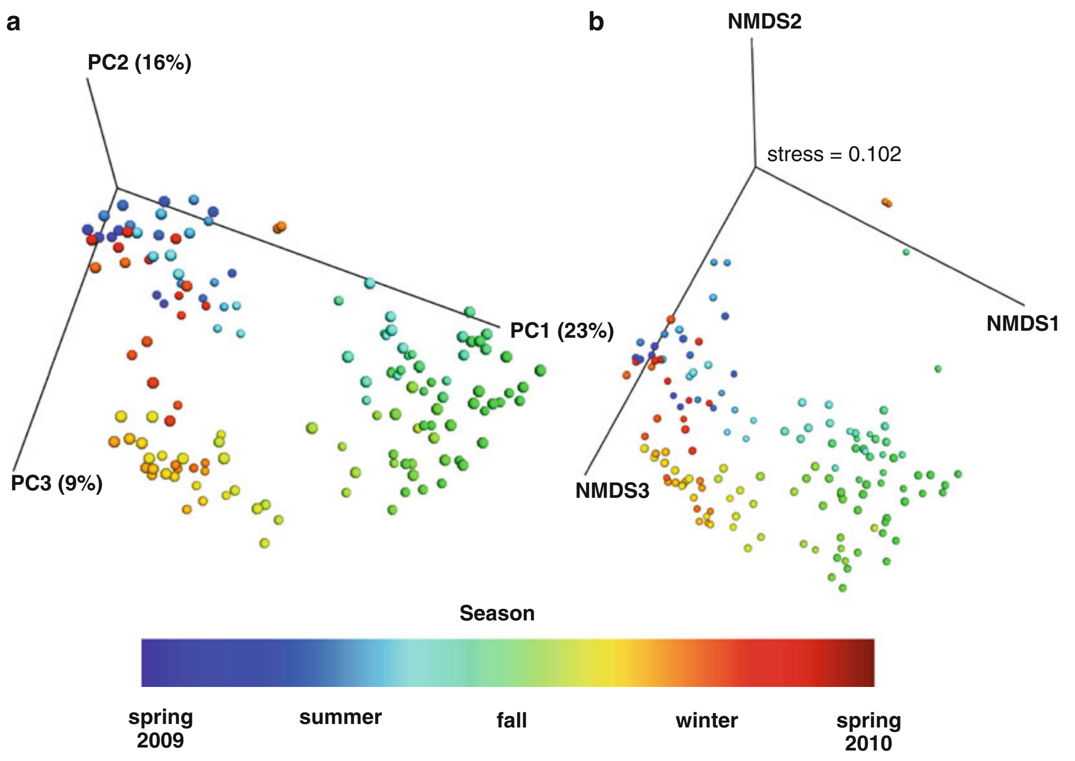


Fig. 1 PCoA (**a**) and NMDS (**b**) plots of the English Channel time series data, with samples colored by time of year. Principal component coordinates were calculated from a Hellinger distance transformation of the raw abundance data

different across seasons. We must bin our time series into seasons, assuming that spring begins on March 21, summer begins on June 21, fall begins on September 23, and winter begins on December 21, in the Northern Hemisphere. We add these categories to our metadata file under the column name "SEASON" (see modified mapping file in the data directory). Then we can run an ADONIS test (nonparametric PERMANOVA) (29), to show that samples cluster significantly based on the season they were sampled from ($F = 16.423$, $p < 0.001$; relevant QIIME commands are listed at the end of this section).

We will now look at the correlation between community structure and continuous time. In order to generate a temporal distance matrix, we need to add a column to the mapping file that converts dates into a monotonically increasing, standardized time unit. In this case, we will use Unix epoch time (labeled as "DAYS_SINCE_EPOCH" in metadata file), which is a standard time unit for Unix-based operating systems, which records the number of seconds that have elapsed since Thursday, January 1, 1970. The distance_matrix_from_mapping.py script in QIIME will take a continuous variable from a mapping file and compute all pairwise distances between samples based on that variable to generate distance matrix.

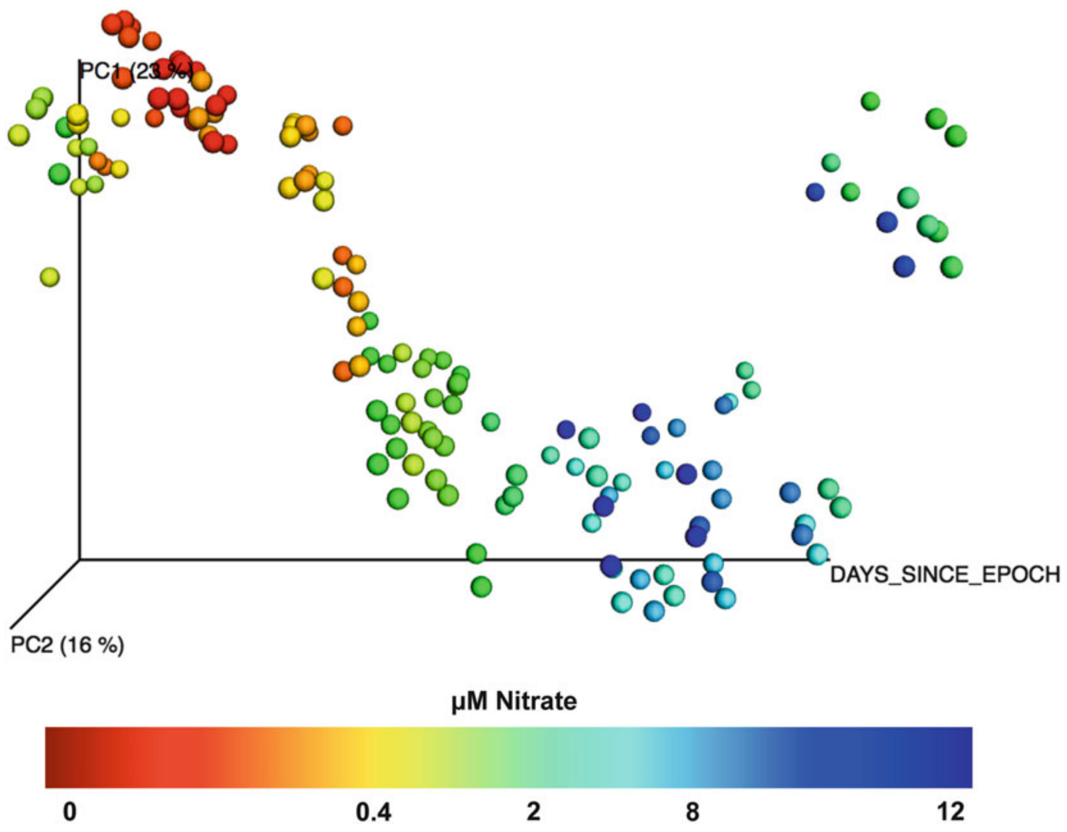


Fig. 2 Scatter plot of marine sample coordinates along first two PC axes and sampling time (DAYS_SINCE_EPOCH) shows the seasonal cycling in community structure. The position of the sample on PC1 defines the season (high = spring; low = fall). Coloring denotes nitrate concentration, which is higher in the summer and fall than in the spring and winter

The mantel test, which tests for linear correlations between two distance matrices, shows that there is, on average, a very significant correlation between time and community similarity (mantel correlation statistic $= 0.376, p < 0.001$) (30). However, we can see that there is a nonlinear periodicity to community structure over the course of the year, based on the sinusoidal beta-diversity pattern in Fig. 2. Thus, we expect that the sign of the linear correlation coefficient would change over different ranges of temporal distance. In order to test this idea, we generate a mantel correlogram, which summarizes mantel tests run across different distance classes (i.e., different ranges of temporal distance; *see* Fig. 3). Just as we suspected, the relationship between community structural distance and temporal distance changes across different distance classes. Over shorter temporal scales, there is a positive correlation between community dissimilarity and time, but over longer temporal distances, there is a negative correlation between community dissimilarity and time (Fig. 4). Therefore, we can conclude that

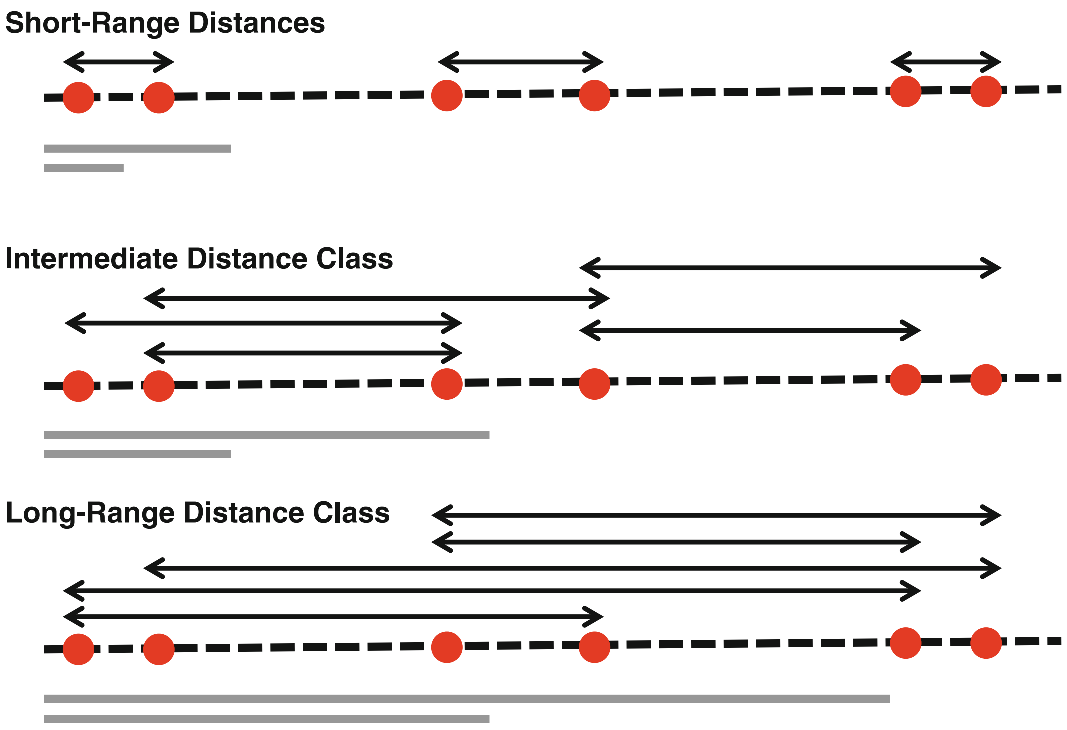


Fig. 3 Conceptual figure showing how distance classes are constructed for a mantel correlogram. *Dashed lines* indicate a gradient (time, space, pH, temperature, etc.), and *red dots* show a sample's coordinate along the gradient. Distances are binned into classes (short range to long range; going from top to bottom in the figure), with maximum and minimum distance thresholds for each class (*gray bars*). Double-headed arrows indicate distances that fall into a particular class. Gradient distances within each bin are then regressed against the equivalent community structural distances

communities diverge from one another over smaller timescales (months), but they become more similar again over longer timescales (a year). This fits with what we observed in Fig. 2, where the spring samples taken in 2009 looked more similar to the spring samples taken in 2010 than to samples collected during any other season (along PC1).

We have established a convincing link between season and microbial community structure, which is mostly accounted for by the first principal coordinate axis in Figs. 1 and 2. In order to identify which environmental forces are driving the seasonal patterns in the microbial community, and remove some of the cross-correlations between metadata variables, we need to determine the optimal set of variables that are most explanatory. There are many ways of running such an analysis, such as forward selection on constrained ordination axes (e.g., distance-based redundancy analysis or canonical correspondence analysis), which can be carried out using the vegan package in R (31). In this chapter, we will employ the Best Subset of Environmental Variables with Maximum (Rank) Correlation with Community Dissimilarities analysis (otherwise

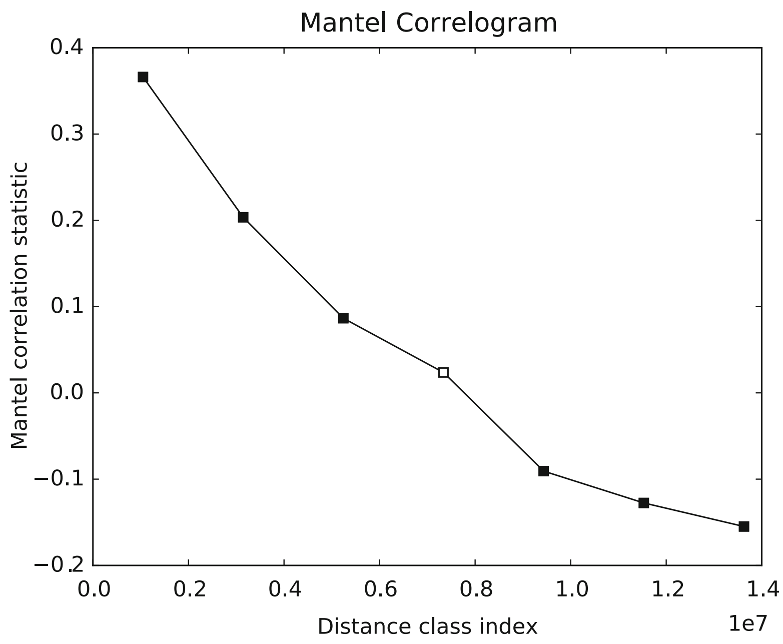


Fig. 4 Mantel correlogram showing the relationship between community structure and time over different temporal distance classes (short timescales on the left and longer timescales on the right; time measured in days). Over shorter timescales, there is a positive correlation between community dissimilarity and time, but this relationship is reversed over longer timescales. *Filled boxes* indicate statistical significance ($p < 0.05$)

known as BEST analysis), which is also in the vegan analysis package in R. This analysis will identify optimal n-parameter models for 1-n metadata variables. The "best" n-parameter model will have the highest rho value (test statistic generated by BEST) (32).

The optimal model contained four variables: silicate, dissolved oxygen, chlorophyll, and nitrite (in order of importance; $\rho = 0.4532$). Mantel tests show that these variables are all independently correlated with community composition ($p < 0.001$). If we plot the filtered data set that contains the complete set of metadata, with time along the x-axis and the first two PC coordinates along the *y*- and *z*-axes, we can visualize how these variables are related to time (Fig. 5):

1. [ADONIS test, hellinger distance vs. season, 10,000 permutations] Takes a beta-diversity distance matrix as input and runs a nonparametric statistical test to determine whether samples cluster based on a metadata category: <compare_categories.py –method adonis -i bdiv_study_1240_d6200_bray_hellinger/hellinger_dm.txt -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -c SEASON -n 10000 -o adonis_hellinger_season>

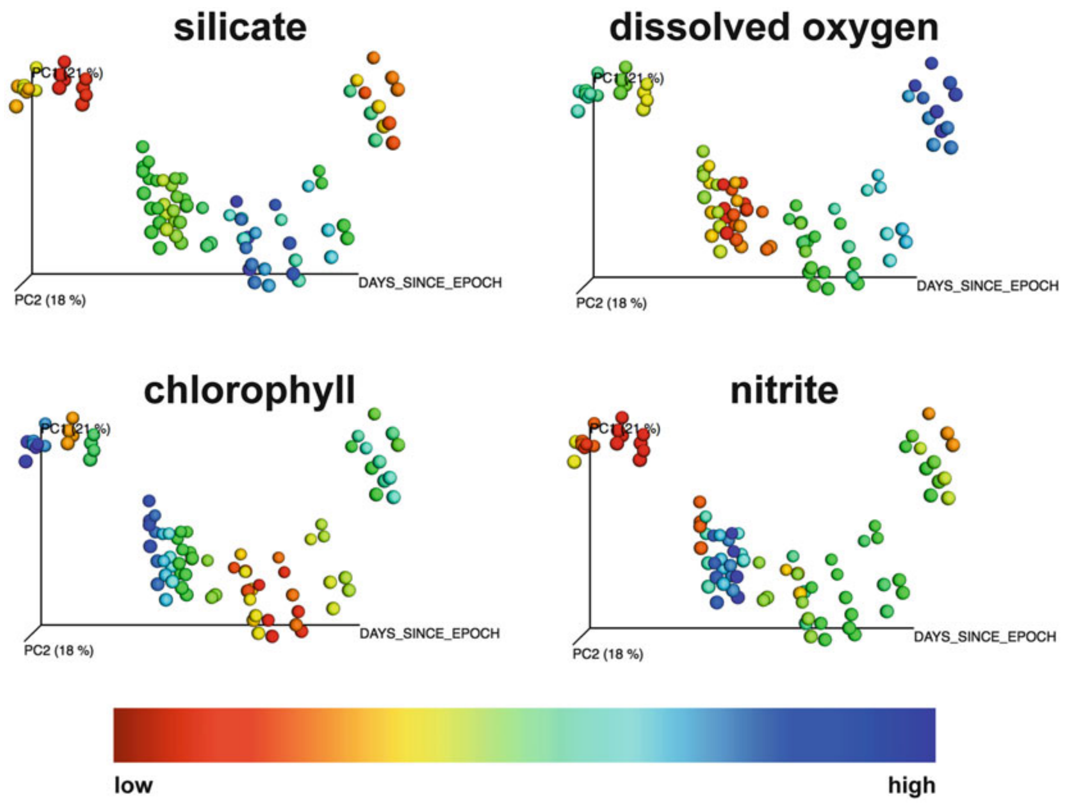


Fig. 5 Similar to Fig. 3, showing the 4 metadata variables that are optimal for explaining the variance in microbial community structure, according to a BEST analysis. *Red* denotes the lowest values for a particular variable, and *blue* denotes the highest values

2. [generate temporal distance matrix] Takes a continuous variable from the mapping file and creates a distance matrix, with pairwise comparisons of each sample: <distance_matrix_from_mapping.py -i study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -c DAYS_SINCE_EPOCH -o DAYS_SINCE_EPOCH.txt>
3. [basic mantel test with 10,000 permutations] Tests for linear correlations of sample distances from two distance matrices: <compare_distance_matrices.py –method mantel -i bdiv_study_1240_d6200_bray_hellinger/hellinger_dm.txt,DAYS_SINCE_EPOCH.txt -o mantel_days_since_epoch -n 10000>
4. [generate mantel correlogram for community structure distance vs. temporal distance] Runs a mantel correlation, as in the previous command, but over a range of distance classes: <compare_distance_matrices.py –method mantel_corr -i bdiv_study_1240_d6200_bray_hellinger/hellinger_dm.txt,DAYS_SINCE_EPOCH.txt -o mantel_days_since_epoch_correlogram -n 10000>

5. [BEST analysis for 14 metadata variables] Takes n continuous metadata parameters and computes the "best" 1-n parameter models that account for community variance: <compare_ categories.py –method best -i bdiv_study_1240_d6200_bray_ hellinger/hellinger_dm_fullmeta.txt -m study_1240_split_library_seqs_and_ mapping/study_1240_ mapping_file_ modified_fullmeta. txt -c AMMONIUM,RESPIRATION_COMMUNITY,NANOEUKARYOTE_COUNT,NITRATE,TEMP,SALINITY,SILICATE,DISS_OXYGEN_2,DISS_OXYGEN_1,DISS_OXYGEN_3,NITRITE,PHOSPHATE,TOTAL_BACTERIA_COUNT,CHLOROPHYLL - o best_hellinger_ fullmeta>
6. [generate pc axes for subset of data with full complement of metadata] Computes principal components for a community distance matrix: <principal_coordinates.py -i bdiv_study_ 1240_d6200_bray_hellinger/hellinger_dm_fullmeta.txt -o bdiv_ study_1240_d6200_bray_hellinger/hellinger_pc_fullmeta. txt>
7. [generate PCoA plots] Generates a 3D PCoA plot for samples with full set of metadata: <make_emperor.py -i bdiv_ study_1240_d6200_bray_hellinger/hellinger_pc_fullmeta.txt -m study_1240_split_library_seqs_and_mapping/study_1240_ mapping_file_modified_fullmeta.txt -o bdiv_study_1240_d6200_bray_hellinger/hellinger_emperor_ time_fullmeta –ignore_missing_samples –add_unique_ columns -a DAYS_SINCE_EPOCH>

2.6 Understanding Shifts in Community Composition

We have established that changes in the structure of the microbial community are tied to seasonal cycles in the environment. We also identified a subset of metadata variables that optimally explain these changes. Now, we will dig into the individual organisms that are responsible for these shifts in the community and see how they are related to season and metadata parameters. First, we can ask whether particular OTUs in our data set are more or less abundant across different seasons. We can use a Kruskal–Wallis test (i.e., a nonparametric ANOVA) to look for significant differences for all taxa in the data set (33). However, due to the large number of individual tests, it is possible that many of the results are false positives (i.e., the more tests you run, the probability that a "significant" *p*-value is obtained due to chance increases). Therefore, we will apply two different multi-test correction factors (Bonferroni and FDR). The Bonferroni correction simply multiplies the *p*-value by the number of tests that were performed and is considered to be quite conservative. The false discovery rate (FDR) correction is less conservative and uses a rank-based correction (34). QIIME commands are listed at the end of the section.

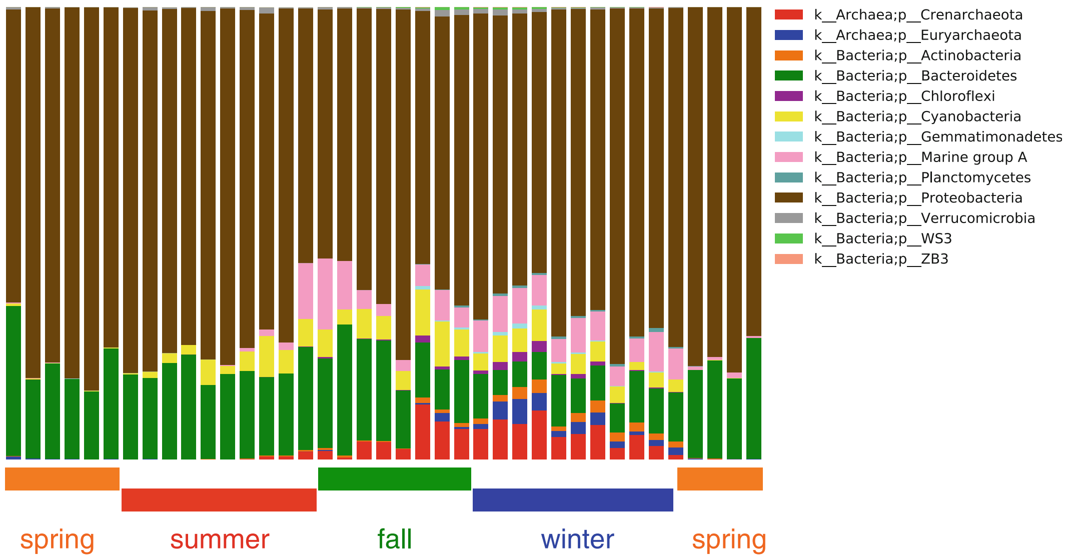


Fig. 6 Relative abundances of phyla for subset of OTUs ($n = 327$) that fluctuate significantly across seasons (Bonferroni-corrected $p < 0.05$). Each *bar* represents a sampling date, and time is increasing toward the right

Out of the 5,692 OTUs in the data set, 327 of them show significant shifts in abundance across seasons (Bonferroni-corrected $p < 0.05$). If we use this list of OTUs to filter our abundance matrix, we can re-normalize and plot the distribution of these taxa at the phylum level across seasons (Fig. 6). Figure 6 reveals an increase in community evenness (at the phylum level) during the fall and winter seasons. This naturally leads us to question whether or not alpha diversity is significantly higher during the fall and winter months. In order to test for these seasonal differences in alpha diversity, we generate replicate rarefactions at a given sequencing depth for the whole community and then run pairwise seasonal comparisons using the student's *t*-test (2-tailed). We find that, indeed, alpha diversity – both richness and evenness – is significantly greater in the fall and winter than in the spring and summer ($p < 0.05$):

1. [Kruskal–Wallis test - season vs. OTU abundance] Runs a non-parametric ANOVA to test for significant differences in OTU relative abundance across metadata categories: <group_ significance.py -i study_1240_split_library_seqs_and_mapping/study_ 1240_closed_reference_otu_table_d6200.biom -m study_1240_ split_library_seqs_and_mapping/study_1240_mapping_file_modified. txt -c SEASON -o kw_season_alldata_results.txt –permutations 10000>
2. [filter OTU table] Creates OTU table that only contains OTUs that change significantly with season (based on results from

prior script): <filter_otus_from_otu_table.py -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table.biom –negate_ids_to_exclude -e dynamic_otus_season.txt -o study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_seasonal_otus.biom>

3. [re-normalize] Subsamples OTU table to an even sequence depth per sample: <single_rarefaction.py -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_seasonal_otus.biom -d 1000 -o study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_seasonal_otus_d1000.biom>
4. [plot taxonomic summary plots] Pools samples based on sampling day and summarizes relative abundances at different taxonomic levels: <summarize_taxa_through_plots.py -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_seasonal_otus_d1000.biom -c SEASON -s -o taxa_summary_plots_seasonal_otus -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt>
5. [generate multiple rarefactions] Calculates alpha diversity at multiple rarefaction depths: <alpha_rarefaction.py -i study_1240_split_library_seqs_and_mapping/study_1240_closed_reference_otu_table_d6200.biom -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -o alpha_rarefaction -p alpha_params.txt>
6. [parameter file content]: alpha_diversity:metrics observed_species,shannon,simpson,equitability,dominance
7. [test for differences in alpha diversity based on season] Tests for alpha diversity differences across seasons, for both richness and evenness: <compare_alpha_diversity.py -i alpha_rarefaction/alpha_div_collated/observed_species.txt -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -c SEASON -o alpha_richness_ season.txt -n 10000>
   <compare_alpha_diversity.py -i alpha_rarefaction/alpha_div_collated/equitability.txt -m study_1240_split_library_seqs_and_mapping/study_1240_mapping_file_modified.txt -c SEASON -o alpha_equitability_season.txt -n 10000>

3 Summary

In this chapter, we covered a small number of available tools, but I have cited several reference texts that give a more comprehensive description of ordination methods and statistical tools for molecular ecology (8, 19, 27, 31). Despite their importance, we did not touch upon several classes of analysis, like the generation of species covariance matrices (35, 36), the application of network analyses (37–39), the time series analysis (40, 41), and the use of null models to explain community assembly patterns (42, 43). The reader is directed to the references provided for further reading on these topics. In addition, there is a long-standing issue in this field having to do with inferring correlations from relative abundances. Recent work has shown how spurious correlations are often obtained from relative abundance data, especially in low-diversity systems (35). A method now exists for correcting for these effects when computing linear correlations (35), but nonlinear methods have not yet been developed. This unresolved issue affects both pairwise correlations between relative abundances (e.g., OTU–OTU correlations) and correlations between relative and absolute measures (e.g., OTU–metadata correlations). Thus, these correlations should be viewed with a degree of skepticism.

Most of the statistical techniques we discussed in this chapter can also be applied to shotgun metagenome data or other types of data that can be summarized as abundance matrices (e.g., phospholipid fatty acid profiles, microarrays, mass spectrometry data, etc.). For example, you could run similar analyses on gene distributions across genomes or metagenomes, transcript distributions across transcriptomes, protein distributions across proteomes, or metabolite distributions across metabolomes. In fact, any abundance matrix can be converted into a QIIME-compatible format (http://biom-format.org/), which would allow you to use many of the scripts discussed in this article on other types of data. In addition to QIIME, many other analysis packages and websites are available for amplicon and metagenome sequence analyses, like mothur, VAMPS, MED, and MG-RAST, to name a few (44–47). Finally, I would also encourage any aspiring microbial ecologist to learn a coding language (e.g., python, R, Perl, MATLAB, etc.). Despite the wealth of bioinformatics and statistical tools available, there will always be custom analyses and data processing tasks that will make your life easier and your science better. Learning to code will be an invaluable investment in your future. A short list of coding resources is provided at the end of this chapter.

Every analysis is a journey into the unknown. I hope this brief example was a useful primer. Good luck!

Coding Resources

1. http://www.codecademy.com/
2. http://rosalind.info/problems/locations/
3. http://learnpythonthehardway.org/book/index.html

References

1. Hugenholtz P (2002) Exploring prokaryotic diversity in the genomic era. Genome Biol 3 (2):1-0003.0008
2. Muyzer G, Smalla K (1998) Application of denaturing gradient gel electrophoresis (DGGE) and temperature gradient gel electrophoresis (TGGE) in microbial ecology. A Van Leeuw J Microb 73(1):127–141
3. Frostegård Å, Tunlid A, Bååth E (1993) Phospholipid fatty acid composition, biomass, and activity of microbial communities from two soil types experimentally exposed to different heavy metals. Appl Environ Microbiol 59 (11):3605–3617
4. Marsh TL (1999) Terminal restriction fragment length polymorphism (T-RFLP): an emerging method for characterizing diversity among homologous populations of amplification products. Curr Opin Microbiol 2 (3):323–327
5. Olsen GJ, Lane DJ, Giovannoni SJ, Pace NR, Stahl DA (1986) Microbial ecology and evolution: a ribosomal RNA approach. Annu Rev Microbiol 40(1):337–365
6. Shokralla S, Spall JL, Gibson JF, Hajibabaei M (2012) Next-generation sequencing technologies for environmental DNA research. Mol Ecol 21(8):1794–1805
7. Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Appl Environ Microbiol 71 (12):8228–8235
8. Ramette A (2007) Multivariate analyses in microbial ecology. FEMS Microbiol Ecol 62 (2):142–160
9. Teeling H, Glöckner FO (2012) Current opportunities and challenges in microbial metagenome analysis—a bioinformatic perspective. Brief Bioinform 13(6): 728–742
10. Gilbert JA et al (2009) The seasonal structure of microbial communities in the Western English Channel. Environ Microbiol 11 (12):3132–3139
11. Gilbert JA et al (2010) Meeting report: the terabase metagenomics workshop and the vision of an Earth microbiome project. Stand Genomic Sci 3(3):243
12. Rideout JR et al (2014) Subsampled open-reference clustering creates consistent, comprehensive OTU definitions and scales to billions of sequences. PeerJ 2:e545
13. McDonald D et al (2012) An improved Greengenes taxonomy with explicit ranks for ecological and evolutionary analyses of bacteria and archaea. ISME J 6(3):610–618
14. Caporaso JG et al (2010) QIIME allows analysis of high-throughput community sequencing data. Nat Meth 7(5):335–336
15. R Development Core Team (2008) R: A language and environment for statistical computing Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org/
16. Lawton JH (1999) Are there general laws in ecology? Oikos 177–192
17. Larsen PE, Gibbons SM, Gilbert JA (2012) Modeling microbial community structure and functional diversity across time and space. FEMS Microbiol Lett 332(2):91–98
18. Caporaso JG et al. (2011) Global patterns of 16S rRNA diversity at a depth of millions of sequences per sample. Proc Natl Acad Sci USA 108(Suppl 1):4516–4522
19. Legendre P, Legendre LF (2012) Numerical ecology. Elsevier. Amsterdam, Netherlands
20. Lozupone C, Lladser ME, Knights D, Stombaugh J, Knight R (2011) UniFrac: an effective distance metric for microbial community comparison. ISME J 5(2):169
21. Legendre P, Gallagher ED (2001) Ecologically meaningful transformations for ordination of species data. Oecologia 129(2):271–280
22. Kruskal JB (1964) Nonmetric multidimensional scaling: a numerical method. Psychometrika 29(2):115–129
23. Jiang X, Hu X, Shen H, He T (2012) Manifold learning reveals nonlinear structure in metagenomic profiles. In: 2012 I.E. international conference on bioinformatics and biomedicine (BIBM), IEEE, pp 1–6
24. Cacciatore S, Luchinat C, Tenori L (2014) Knowledge discovery by accuracy maximization. Proc Natl Acad Sci USA 111 (14):5117–5122
25. Gower JC (2005) Principal coordinates analysis. Encyclopedia Biostat doi:10.1002/0470011815.b2a13070. http://onlinelibrary.wiley.com/doi/10.1002/0470011815.b2a13070/abstract?deniedAccessCustomisedMessage=&userIsAuthenticated=false
26. Vázquez-Baeza Y, Pirrung M, Gonzalez A, Knight R (2013) EMPeror: a tool for visualizing high-throughput microbial community data. GigaScience 2(1):16
27. Clarke KR (1993) Non-parametric multivariate analyses of changes in community structure. Aust J Ecol 18(1):117–143

28. Sheskin DJ (2003) Handbook of parametric and nonparametric statistical procedures CRC. Boca Raton, Florida, USA
29. Anderson MJ (2005) Permutational multivariate analysis of variance. Department of Statistics, University of Auckland, Auckland
30. Mantel N (1967) The detection of disease clustering and a generalized regression approach. Cancer Res 27(2 Part 1):209–220
31. Oksanen J (2011) Multivariate analysis of ecological communities in R: vegan tutorial. R package version 1(7) http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0CCcQFjAB&url=http%3A%2F%2Fcc.oulu.fi%2F~jarioksa%2Fopetus%2Fmetodi%2Fvegantutor.pdf&ei=M2LjVOfXLIWgNsaRhJAO&usg=AFQjCNHsvyIZ380_KPgiGMqah_gA5V2jLQ&sig2=fMlVe0QMmwc1yNxmvRuCVQ&bvm=bv.85970519,d.eXY
32. Clarke K, Ainsworth M (1993) A method of linking multivariate community structure to environmental variables. Mar Ecol Prog Ser 92:205
33. Sawilowsky S, Fahoome G (2005) Kruskal–Wallis test. Encyclopedia of Statistics in behavioral Science http://onlinelibrary.wiley.com/doi/10.1002/0470013192.bsa333/abstract?deniedAccessCustomisedMessage=&userIsAuthenticated=false
34. Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. J R Stat Soc Ser B 289–300
35. Friedman J, Alm EJ (2012) Inferring correlation networks from genomic survey data. PLoS Comput Biol 8(9):e1002687
36. Ruan Q et al (2006) Local similarity analysis reveals unique associations among marine bacterioplankton species and environmental factors. Bioinformatics 22(20):2532–2538
37. Barberán A, Bates ST, Casamayor EO, Fierer N (2011) Using network analysis to explore co-occurrence patterns in soil microbial communities. ISME J 6(2):343–351
38. Dunne JA, Williams RJ, Martinez ND (2002) Food-web structure and network theory: the role of connectance and size. Proc Natl Acad Sci USA 99(20):12917–12922
39. Alm E, Arkin AP (2003) Biological networks. Curr Opin Struc Biol 13(2):193–202
40. Xia LC et al (2011) Extended local similarity analysis (eLSA) of microbial community and other time series data with replicates. BMC Syst Biol 5(Suppl 2):S15
41. David LA et al (2014) Host lifestyle affects human microbiota on daily timescales. Genome Biol 15(7):R8
42. Stone L, Roberts A (1990) The checkerboard score and species distributions. Oecologia 85 (1):74–79
43. Gotelli NJ, Ulrich W (2012) Statistical challenges in null model analysis. Oikos 121 (2):171–180
44. Schloss PD et al (2009) Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl Environ Microbiol 75(23):7537–7541
45. Huse SM et al (2014) VAMPS: a website for visualization and analysis of microbial population structures. BMC Bioinformat 15 (1):41
46. Glass EM, Meyer F (2011) The metagenomics RAST server: a public resource for the automatic phylogenetic and functional analysis of metagenomes. handbook of molecular microbial ecology I. Wiley, Hoboken, New Jersey, USA pp 325–331.
47. Eren AM et al (2014) Minimum entropy decomposition: unsupervised oligotyping for sensitive partitioning of high-throughput marker gene sequences. ISME J. http://www.nature.com/ismej/journal/vaop/ncurrent/full/ismej2014195a.html

Statistical Tools for Study Design: Replication

Peter E. Larsen

Abstract

The ability to collect massive amounts of data relevant to hydrocarbon microbiology from marine and terrestrial environments continues to grow. Given the potential expense and effort of collecting this data however, it is important that careful consideration of experimental design is given before undergoing data collection to maximize the probability that experimental results will lead to statistically significant conclusions. One of the most important aspects of experimental design is the selection of appropriate number of experimental replications. The number of experimental replicates needed to achieve a desired level of statistical significance is dependent upon the known or estimated distributions of the experimental system, the statistical tool that will be used in data analysis, the size of the effect that is anticipated from the experiment, and the desired power of the statistical results. Here, protocols are presented for identifying the most appropriate number of experimental replications to achieve a desired level of statistical power.

Keywords: Ecology, Experimental design, Power analysis, Replication

1 Introduction

The ability to collect data from terrestrial and marine environments has never been greater. From satellite remote sensing and automated data collection buoys and remotely operated vehicles, to an array of omics technologies and high-throughput mass spectrometry, the ability to capture vast amounts of information has opened previously inaccessible views into biological systems at multiple ecological scales [1, 2]. The ability to collect this data is, however, not synonymous with the ability to analyze the data effectively. With the advances of data collection techniques, the need for the application of the best possible experimental design and statistical analysis methodologies is more crucial than ever. One important aspect of experimental design is selection of an appropriate number of experimental replicates. An insufficient number of samples can result in an experiment without the capacity to draw statistically significant conclusions. Conversely, oversampling can incur unnecessary costs and time for data collection. Estimation of the number

T.J. McGenity et al. (eds.), *Hydrocarbon and Lipid Microbiology Protocols,* Springer Protocols Handbooks (2016) 59–68, DOI 10.1007/8623_2015_95, © Springer-Verlag Berlin Heidelberg 2015, Published online: 09 June 2015

		Decision	
		Accept the Null Hypthesis H_0: There is no differnce between samples	Reject the Null Hypothysis H_0: Samples are different
Truth	There is no differnce between samples	**Correct** $1-\alpha$	Type I error α
	Samples are different	Type II error β	**Correct** $1-\beta$

Fig. 1 Decision to accept or reject the null hypothesis, H_0. When comparing two populations, the underlying populations may actually have the same average values or different values for a measurable characteristic ("Truth" in figure). Upon analysis of a random subset of populations, the null hypothesis (H_0) that the populations are identical can be accepted or rejected ("Decision" in figure). If populations are actually identical, but a decision to reject the null hypothesis has been made, then a Type I error has been made. If populations are actually different, but the null hypothesis has been accepted, then a Type II error has been made

of replicates therefore is a crucial step in experimental design and needs to be performed before any research effort is undertaken.

The basic premise of any statistical test is as follows: Given two or more sets of samples collected from potentially much larger populations, it is possible to determine if the larger populations are the same or are distinct from one another [3]. The hypothesis that the populations are the same is termed the null hypothesis, notated as H_0. Deciding that the populations are different is called the rejection of the null hypothesis. Errors occur when the null hypothesis is inappropriately accepted or rejected (Fig. 1). A Type I error occurs when the null hypothesis is rejected when it is actually true. That is, the analysis incorrectly identified a significant difference between samples when they are in fact the same. The probability of committing a Type I error is α. A Type II error occurs when the null hypothesis is not rejected when it is actually false. That is, the analysis incorrectly finds that the samples are the same when they are in fact different. The probability of committing a Type II error is β. The power of a test, defined as $1-\beta$, is the probability that a decision to reject the null hypothesis was correct. Power is an important component in the interpretation of experimental results and defines how much confidence can be assigned to experimental results. A power between 0.8 and 0.9 is usually considered acceptable, although the specific selection of power is derived from the specific experimental design and potential applications of the experimental results [3]. For example, the power desired for an experiment identifying the utility of a potential drug for human use might

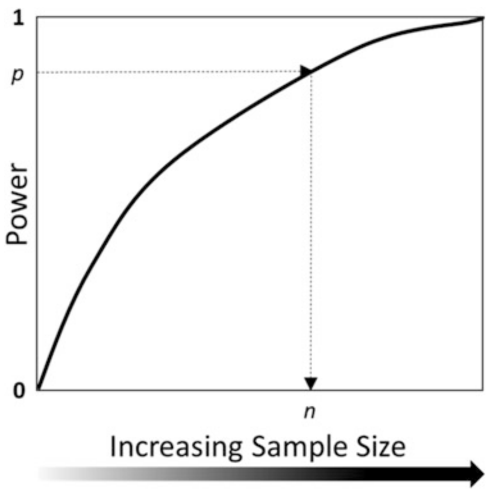


Fig. 2 Power is a function of sample size. In this idealized figure, power is shown to increase with sample size. Using a figure such as this, it becomes possible to select an experimental sample size, *n*, that will provide a given power, *p*

be higher that the power for an experiment studying the ecological distributions of a species of marine microorganism.

While it is possible to calculate the power of a test retrospectively, a better application is to incorporate power into experimental design. When the variance of the populations being measured is known or can be estimated, the number of replicates (*see* Note 1) can be selected that will provide a desired power (Fig. 2). The choice of appropriate sample size depends upon the statistical test that will be used in analysis and upon the effect size that is desired to be detected in the data. While there are a variety of possible statistical analysis approaches (*see* Note 2), several common approaches will be considered here:

- **Z-test**: Z-test statistics is appropriate when only one population is under consideration, and a determination is to be made if the average value of observations is significantly different, greater than, or less than a fixed value.
- ***t*-test**: When comparing the averages and standard deviations from two sets of samples, a *t*-test is most appropriate.
- **ANOVA**: ANOVA is used when comparing the averages and standard deviations from more than two sets of samples.
- **Proportions**: When multiple data observations can be expressed as ratios, a proportions test can be utilized.
- **Correlation**: The correlation coefficient between samples can be considered when an observation is comprised of multiple variables, such as a vector of abundances of microbial taxa or a vector of concentrations of several chemicals.

Effect size is the strength of difference between samples you want to be able to detect in your analysis. The larger the effect size, the bigger the difference between samples. Small effect sizes will require more replicates in order to detect significant differences for a particular *p*-value. There are three ways that effect size may be selected.

- Effect size may be inherent in the experimental goals and based on relevant knowledge. For example, a test may be designed to detect at least a 5% difference between measurements of a concentration of a specific contaminant, based on the knowledge that a 5% increase will have a meaningful biological effect.
- Effect size may be determined from knowledge of the averages and standard deviations of the populations being studied. This approach requires some significant pre-knowledge of the experimental system, as from a pilot experiment or previously published data.
- Effect sizes may be based on values derived from commonly used thresholds in power analysis. For example, a set of effect sizes for different statistical tests initially proposed by Jacob Cohen [4] are commonly used effect size values (*see* Note 3).

Just as there are many possible experimental designs, there is an array of tools available for generating statistical analyses. The methods presented here were selected to need only a moderate knowledge of statistics, to use only freely available software, and to require only modest computational power. The assumption here is that there is limited prior information available for estimating population variances.

No single approach to power analysis is appropriate for all possible experimental designs, and no set of assumptions regarding population distributions will be best suited to all possible biological observations. Determining power for many data sets requires having a large amount of prior data for the experimental system. While the procedures described here are suitable for a wide variety of applications for which little prior information is available, some additional tools for power analysis have been listed (*see* Note 4). For metagenomic data analysis, more specialized tools for power analysis may be appropriate (*see* Note 5).

2 Materials

The approaches outlined here require the installation and use of the software package "R." R is a free software environment for statistical computing and graphics (*see* Note 6). R-package "pwr" needs to be loaded into the "R" computing environment to use the following functions (*see* Note 7). R-package "pwr" implements power analysis as outlined by Cohen [4].

A knowledge of the "R" programming environment will be helpful but is not necessary to follow the steps described below.

3 Methods

In the following protocols, specific commands are given that are to be typed into the command line of the R computing environment. Copy the commands as presented, replacing text in italics with user-selected values as described in subsequent instructions. Due to the number of unknowns in a given proposed experiment, the most appropriate use of these tools is often to determine the replicate numbers required for a range of possible parameter values.

3.1 Z-Test

Example: How many samples are needed to determine if the concentration of substance X collected from site A is greater than 100 mg/mL?

Command-line input to calculate the power analysis of a Z-test is:

```
> pwr.norm.test(d = ES, n = NULL, sig.level = Alpha, power = P,
alternative = Hypothesis)
```

where

- *Alpha* is equal to the Type I error probability, e.g., 0.05.
- *P* = power (1 – Type II error probability), e.g., 0.8.
- For *Hypothesis*, select one of the following strings:
  - "two.sided" (μ is greater than or less than μ_0)
  - "less" (μ is less than μ_0)
  - "greater" (μ is greater than μ_0)
- *ES* is the effect size, which can be determined by:
  - $(|\mu_0 - \mu|)/\sigma$, where μ is the average of observations, σ is the standard deviation of observations, and μ_0 is the value the data is being tested against.

3.2 t-Tests

Example: Samples can be collected from two locations: site A and site B. How many samples are needed to determine if the concentrations of substance X are significantly different at site A compared to site B?

3.2.1 Same Sample Sizes

When the sample sizes for populations 1 and 2 are equal, the command-line input to calculate the power analysis of a Z-test is:

```
> pwr.t.test(d = ES, n = NULL, sig.level = Alpha, power = P, alter-
native = Hypothesis)
```

where

- *Alpha* is equal to the Type I error probability, e.g., 0.05.
- P = power (1 – Type II error probability), e.g., 0.8.
- For *Hypothesis*, select one of the following strings, including quotes:
  - "two.sided" (μ_1 is greater than or less than μ_0)
  - "less" (μ_1 is less than μ_0)
  - "greater" (μ_1 is greater than μ_0)
- *ES* is the effect size, which is equal to:
  - $(|\mu_1 - \mu_2|)/\sigma$, where μ_1 and μ_2 are the average of observations for populations 1 and 2 and σ is the standard deviation from all observations.
  - Cohen suggests 0.2 (small), 0.5 (medium), and 0.8 (large) effect sizes.

3.2.2 Different Sample Sizes

When the sample sizes of populations 1 and 2 are not equal, the command-line input to calculate the power analysis of a Z-test is:

```
> pwr.t2n.test(d = ES, n1 = N, n2 = NULL, sig.level = Alpha, power =
P, alternative = Hypothesis)
```

where input variables are mostly the same as Sect. 3.2.1 with the addition of n_1 and n_2 for the unequal sample sizes for the two populations. In the above command line, N is the known sample size of population 1, and the results will return the size of population 2 required for power P.

3.3 ANOVA

3.3.1 Single-Factor ANOVA

Example: Samples can be collected from multiple locations: sites A, B, C, and D. How many samples are needed from each location to determine if the concentrations of substance X are significantly different for at least one of the sites?

When the number of groups in the experimental design is g, the command-line input to calculate the power analysis of ANOVA is:

```
> pwr.anova.test(k= g, n = NULL, f = ES, sig.level = Alpha, power = P)
```

where

- g is the number of groups.
- *Alpha* is equal to the Type I error probability, e.g., 0.05.
- P = power (1 – Type II error probability), e.g., 0.8.
- *ES* is the effect size, which can be calculated.
  - As eta squared (η^2):

    $f = \sqrt{\frac{\sum_{i=1}^{g} \frac{1}{g} * (\mu_i - \mu)^2}{\delta^2}}$, where μ_i is the average of samples in group i, μ is the average of all samples, σ is the standard deviation of all samples, and g is the number of groups.

– Cohen suggests 0.1 (small), 0.25 (medium), and 0.4 (large) effect sizes.

Note that the resulting population size from this is the number of sample required for *each* of the *g* groups and assumes that each group is of equal size.

3.3.2 Multifactor ANOVA

Example: Samples can be collected from multiple locations: sites A, B, C, and D. Sites A and B are at a natural marine hydrocarbon seep, while sites C and D are not. There are two factors in this experiment: location (i.e., sites A–D) and carbon seep (i.e., a seep is present or absent). How many samples are needed from each location to determine if the concentrations of substance X are significantly different in at least one of the sites and/or due to the presence of hydrocarbon seep?

When considering multifactorial ANOVA, typically consider only the power of the most important factor (i.e., the factor with the largest contribution to the variance). The effect size needs to be calculated taking into account the total degrees of freedom (partial eta squared). Using *f* from previous single-factor ANOVA, the effect size can be calculated for multifactor ANOVA as:

- partial $\eta^2 = \frac{df_1 f}{df_{1f+}\, df_2}$, where df_1 and df_2 are the degrees of freedom for factors 1 and 2, respectively.
- The degree of freedom is the number of categories in the factor minus 1.

R environment command-line input is otherwise identical to that of single-factor ANOVA above.

3.4 Proportions

Example: Ecological effects of a marine hydrocarbon spill can induce disease state in marine animals, measured as fraction of captured animals showing disease. How many samples need to be collected to determine if the fraction of affected animals at site A is significantly different from the fraction of animals affected at site B?

3.4.1 Against Specific Proportion

When the proportion of a single sample is compared against a known proportion, the command line for power analysis of a proportions test is:

```
> pwr.p.test(h = ES, n = NULL, sig.level = Alpha, power = P, alter-
native = Hypothesis)
```

where

- p_1 is the experimentally observed proportion and p_2 is the known proportion against which p_1 is being tested.
- *Alpha* is equal to the Type I error probability, e.g., 0.05.
- P = power (1 – Type II error probability), e.g., 0.8.

- For *Hypothesis*, select one of the following strings:
  - "two.sided" (p_1 is greater than or less than p_0)
  - "less" (p_1 is less than p_0)
  - "greater" (p_1 is greater than p_0)
- *ES* is the effect size, which can be calculated:

  $h = 2 * \text{arc}\,sin\left(\sqrt{p_1}\right) - 2 * \text{arc}\,sin\left(\sqrt{p_2}\right)$, where p_1 and p_2 are the proportions being tested.

  Cohen suggests 0.2 (small), 0.5 (medium), and 0.8 (large) effect sizes.

The proportions test, as presented here, only considers a pair-wise comparison between samples. If the experimental design includes more than two conditions, then all possible pair-wise comparisons in data could be considered, using the largest calculated sample size determined for each sample condition. However, in the case that there are multiple conditions in the experimental design, it is possible that ANOVA is the more appropriate test that should be considered.

3.4.2 Compare Two Proportions with Same Sample Sizes

When the sample sizes of populations 1 and 2 are equal, the command-line input to calculate the power analysis of a proportions test is:

```
> pwr.2p.test(h = ES, n = NULL, sig.level = Alpha, power = P, alter-
native = Hypothesis)
```

where input variables are the same as in Sect. 3.2.1.

3.4.3 Compare Two Proportions with Different Sample Sizes

When the sample sizes of populations 1 and 2 are not equal, the command-line input to calculate the power analysis of a proportions test is:

```
> pwr.2p.test(h = ES, n1 = N, n2 = NULL, sig.level = Alpha, power =
P, alternative = Hypothesis)
```

where input variables are mostly the same as Sect. 3.2.1 with the addition of n_1 and n_2 for the unequal sample sizes for the two populations. In the above command line, *N* is the known sample size of population 1, and the results will return the size of population 2 required for power *P*.

3.5 Correlation

Example: Microbial community structures can be assayed from marine samples, resulting in a list of detected species with their abundances per sample. How many metagenomic samples are required to determine if the abundances of bacterial population A correlate significantly with the abundances of population B, over the set of analyzed samples?

The command-line input to calculate the power analysis of correlations is:

```
> pwr.r.test(n = NULL, r = Correl, sig.level = Alpha, power = P)
```

where

- *Alpha* is equal to the Type I error probability, e.g., 0.05.
- *P* = power (1 – Type II error probability), e.g., 0.8.
- *Correl* is the effect size, which can be determined by:
  - As the desired threshold correlation between populations to be tested.
  - Cohen recommends 0.02 (small), 0.15 (medium), and 0.35 (large) effect sizes.

4 Notes

1. The number of replicates required to achieve desired power will probably be larger than you expected.
2. It would be a good idea to make friends with a statistician. It pays off to involve experienced statisticians in experimental design and data analysis.
3. The use of the Cohen recommended effect sizes is a bit of a blunt instrument in these calculations as they were not generated from marine or ecological data and do not take into account the specific variances and distributions for your unique experimental subjects or analysis methodology. But as information about all the possible variances and Type I error frequencies may not be known for your experimental system, the Cohen estimates for "small," "medium," and "large" effect sizes are an okay place to start, and at least you will have historical precedence on your side.
4. Some additional tools for power analysis under more specialized conditions are listed here
   - For an approach to using unbalanced, time-course data, see Huang and Fitzmaurice [5].
   - For an approach that focuses on computational modeling of biochemical data, see Stegmaier et al. [6].
   - For a computational tool that explicitly addresses multi-level modeling, see Field [7].
5. Metagenomic and microbial population data analyses, due to the very large number of data points relative to the number of observations and the interdependence of microbial species within a population, often require an approach that takes

these conditions into account. Some recent articles addressing power analysis for microbial population studies are listed below

- See Fernandes et al. [8].
- See Friedman and Alm [9].
- See Holmes et al. [10].
- See La Rosa et al. [11].

6. http://cran.r-project.org/.
7. Pwr: http://cran.r-project.org/web/packages/pwr/.

References

1. Larsen P, Hamada Y et al (2012) Modeling microbial communities: current, developing, and future technologies for predicting microbial community interaction. J Biotechnol 160 (1–2):17–24
2. Larsen PE, Gibbons SM et al (2012) Modeling microbial community structure and functional diversity across time and space. FEMS Microbiol Lett 332(2):91–98
3. Vidakovic B (2011) Statistics for bioengineering sciences: with MATLAB and WinBUGS support. Springer, New York
4. Cohen J (1988) Statistical power analysis for the behavioral sciences. L. Erlbaum Associates, Hillsdale
5. Huang W, Fitzmaurice GM (2005) Analysis of longitudinal data unbalanced over time. J R Stat Soc Series B Stat Methodol 67:135–155
6. Stegmaier J, Skanda D, Lebiedz D (2013) Robust optimal design of experiments for model discrimination using an interactive software tool. PLoS One 8(2), e55723
7. Field A (1998) Statistical software for microcomputers: Mlwin and nQuery advisor. Br J Math Stat Psychol 51:367–370
8. Fernandes AD, Macklaim JM, Linn TG, Reid G, Gloor GB (2013) ANOVA-like differential expression (ALDEx) analysis for mixed population RNA-seq. PLoS One 8(7):67019. doi:10.1371/journal.pone.0067019
9. Friedman J, Alm EJ (2012) Inferring correlation networks from genomic survey data. PLoS Comput Biol 8(9):1002687. doi:10.1371/journal.pcbi.1002687
10. Holmes I, Harris K, Quince C (2012) Dirichlet multinomial mixtures: generative models for microbial metagenomics. PLoS One 7 (2):30126. doi:10.1371/journal.pone.0030126
11. La Rosa PS, Brooks JP, Deych E et al (2012) Hypothesis testing and power calculations for taxonomic-based human microbiome data. PLoS One 7(12):52078. doi:10.1371/journal.pone.0052078

MG-RAST, a Metagenomics Service for the Analysis of Microbial Community Structure and Function

Elizabeth M. Glass and Folker Meyer

Abstract

Molecular ecology approaches are rapidly advancing our understanding of microbial communities involved in the synthesis and degradation of hydrophobic organics involved with major consequences for applications in climate change, environmental pollution, human health, and biotechnology. Metagenomics allows researchers to inventory microbial genes in various environments to understand the genetic potential of uncultured bacteria and archaea. Metagenomics enables us to sequence genomes from a complex assemblage of microbes in a culture-independent manner. Amplicon and WGS studies are the most widely employed methods to estimate "who is there" and "what they are doing." Metagenomics allows researchers to access the functional and metabolic diversity of microbial communities.

Since 2008, MG-RAST serves as a repository for metagenomic datasets and as an analysis provider. Currently, the system has analyzed and hosts over 130,000 datasets. Over the years, MG-RAST has undergone a significant number of revisions to accommodate the dramatic growth in dataset size, new data types, and wider system adoption among the research community.

Keywords Automated pipeline, Comparative analysis, High throughput, Metagenomics, Sequence quality

1 Introduction

Metagenomics allows researchers to inventory microbial genes in various environments to understand the genetic potential of uncultured bacteria and archaea. Metagenomics powers our ability to sequence genomes from a complex assemblage of microbes in a culture-independent manner. Amplicon (16 s, 18 s, ITS) and WGS (whole shotgun sequence) studies are the most widely employed methods to estimate "who is there" and "what they are doing." However, understanding the sequence quality is of primary importance. Metagenomics analyses rely on automated analysis tools, and therefore, the estimation of the quality of the sequences is of great importance. Trying to derive meaningful biological inferences from potentially questionable data would be detrimental.

T.J. McGenity et al. (eds.), *Hydrocarbon and Lipid Microbiology Protocols*, Springer Protocols Handbooks (2016) 69–87,
DOI 10.1007/8623_2015_119, © Springer-Verlag Berlin Heidelberg 2015, Published online: 26 July 2015

Concerns about sequence quality are not the only hurdle the scientific community faces with the study of metagenomics data. Another problem is the gap between the shrinking costs of sequencing and the more or less stable costs of computing. Analysis of the sequence data has become the limiting factor, not initial data generation, as seen in the past. Wilkening et al. [1] provide a real currency cost for the analysis of 100 gigabasepairs of DNA sequence data using BLASTX on Amazon's EC2 service: $300,000. A more recent study by the University of Maryland researchers [2] estimates the computation for a terabase of DNA shotgun data using their CLOVR metagenome analysis pipeline at over $5 million per terabase.

In order to be able to conduct meaningful comparative analyses of metagenomic samples, one must consider metadata (contextual data). This provides an essential complement to experimental data, helping to answer questions about its source, mode of collection, and reliability. Metadata collection and interpretation have become vital to the genomics and metagenomics community, but considerable challenges remain, including exchange, curation, and distribution.

Since 2008, MG-RAST [3] serves as a repository for metagenomic datasets and as an analysis provider. Currently, the system has analyzed and hosts over 130,000 datasets. Over the years, MG-RAST has undergone a significant number of revisions to accommodate the dramatic growth in dataset size, new data types, and wider system adoption among the research communities. We have made both substantial engineering changes and modifications to the bioinformatics pipeline to accommodate the evolving needs of novel sequencing technologies and growing data volumes. The MG-RAST system has been an early adopter of the minimal checklist standards and the expanded biome-specific environmental packages devised by the Genomics Standards Consortium [4] and provides an easy-to-use uploader for metadata capture at the time of data submission.

2 Materials

2.1 Database

The MG-RAST automated analysis pipeline uses the M5nr (MD5-based nonredundant protein database) [5] for annotation. The M5nr is an integration of many sequence databases into one single, searchable database (plus an index). A single similarity search (using BLAST [6] or BLAT [7]) will allow the user to retrieve similarities to several databases:

- EBI: European Bioinformatics Institute [8]
- GO: Gene Ontology [9]

- JGI: Joint Genome Institute [10]
- KEGG: Kyoto Encyclopedia of Genes and Genomes [11]
- NCBI: National Center for Biotechnology Information [12]
- Phantome: Phage Annotation Tools and Methods [13]
- SEED: The SEED Project [14]
- UniProt: UniProt Knowledgebase [15]
- VBI: Virginia Bioinformatics Institute [16]
- eggNOG: evolutionary genealogy of genes – non-supervised orthologous groups [17]

Computing of sequence similarity results is becoming a limiting factor in metagenome analysis. Sequence similarity search results encoded in an open, exchangeable format distributed with the sequence sets have the potential to limit the needs for computational reanalysis of datasets.

2.2 Analysis Pipeline

The pipeline shown in Fig. 1 contains a significant number of improvements to previous versions. We have transitioned from using just the SEED database to using the M5nr, made several key algorithmic improvements that were needed to support the flood of user-generated data, and used dedicated software to perform gene prediction instead of using a similarity-based approach which reduces runtime requirements. The additional clustering of proteins at 90% identity reduces data while preserving biological signals. We also restrict the pipeline annotations only to protein-coding genes and ribosomal RNA (rRNA) genes.

In Sect. 3, we describe each step of the pipeline in detail. All datasets generated by the individual stages of the processing pipeline are made available as downloads.

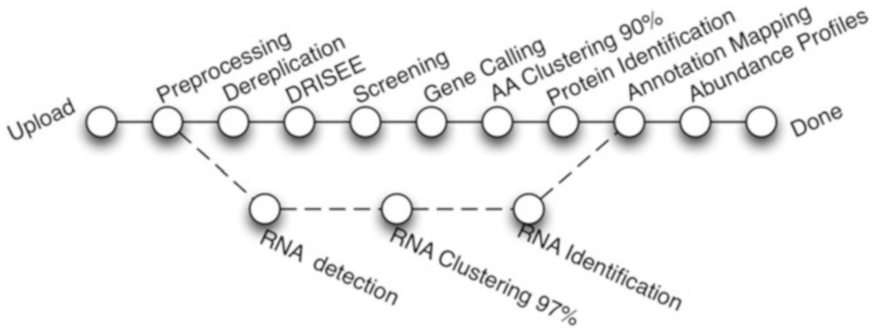


Fig. 1 Details of the analysis pipeline for MG-RAST. After upload, the pipeline diverges for amplicon and WGS datasets. Amplicon samples run through RNA detection, clustering, and identification, while WGS has more steps that start with preprocessing and data quality assessment through annotation and abundance profile creation

2.3 Compute Resources

While originally MG-RAST has been built like a traditional cluster-based bioinformatics system, the most recent version of MG-RAST embraces cloud technologies to enable the use of computer resources anywhere. MG-RAST data is stored in SHOCK and computing is orchestrated by AWE [18]. These technologies were developed to enable execution on a variety of computational platforms; currently, computational resources are contributed by the DOE Magellan cloud at Argonne National Laboratory, Amazon EC2 Web Services, and finally a number of traditional clusters. An installation of the pipeline exists at DOE's NERSC supercomputing center. Currently, the system handles over 2 terabasepairs per month.

3 Methods

The pipeline diverges after upload for amplicon and whole genome shotgun (WGS) samples. The WGS pipeline is composed of several steps from the removal of low-quality reads, dereplication, gene calling, and annotation to creation of abundance profiles. rRNA samples run through RNA detection, clustering, and identification.

3.1 The WGS Pipeline

3.1.1 Preprocessing

After upload, data is preprocessed by using SolexaQA [19] to trim low-quality regions from FASTQ data. Platform-specific approaches are used for 454 data submitted in FASTA format: reads more than two standard deviations away from the mean read length are discarded thereafter [20]. All sequences submitted to the system are available, but discarded reads will not be analyzed further.

3.1.2 Dereplication

For shotgun metagenome and shotgun metatranscriptome datasets, we perform a dereplication step. We use a simple k-mer approach to rapidly identify all 20 character prefix identical sequences. This step is required in order to remove artificial duplicate reads (ADRs) [21]. Instead of simply discarding the ADRs, we set them aside and use them later. We note that dereplication is not suitable for amplicon datasets that are likely to share common prefixes.

3.1.3 DRISEE

MG-RAST v3 uses DRISEE (duplicate read inferred sequencing error estimation) [22] to analyze the sets of ADRs and determine the degree of variation among prefix-identical sequences derived from the same template. DRISEE provides positional error estimates that can be used to inform read trimming within a sample. It also provides global (whole sample) error estimates that can be used to identify samples with high or varying levels of sequencing error that may confound downstream analyses, particularly in the case of studies that utilize data from multiple sequencing samples.

3.1.4 Screening

The pipeline provides the option of removing reads that are near-exact matches to the genomes of a handful of model organisms, including fly, mouse, cow, and human. The screening stage uses Bowtie (a fast, memory-efficient, short read aligner) [23], and only reads that do not match the model organisms pass into the next stage of the annotation pipeline. This option will remove all reads similar to the human genome and render them inaccessible. This decision was made in order to avoid storing any human DNA on MG-RAST.

3.1.5 Gene Calling

The previous version of MG-RAST used similarity-based gene predictions, an approach that is significantly more expensive computationally than de novo gene prediction. After an in-depth investigation of tool performance [24], we have moved to a machine learning approach: FragGeneScan [25]. Using this approach, we can now predict coding regions in DNA sequences of 75 bp and longer. Our novel approach also enables the analysis of user-provided assembled contigs. FragGeneScan is trained for prokaryotes only. While it will identify proteins for eukaryotic sequences, the results should be viewed as more or less random.

3.1.6 AA Clustering

MG-RAST builds clusters of proteins at the 90% identity level using the UCLUST [26] implementation in QIIME [27] preserving the relative abundances. These clusters greatly reduce the computational burden of comparing all pairs of short reads, while clustering at 90% identity preserves sufficient biological signals.

3.1.7 Protein Identification

Once created, a representative (the longest sequence) for each cluster is subjected to similarity analysis. Instead of BLAST, we use sBLAT, an implementation of the BLAT algorithm, which we parallelized using Open MPI for this work.

Once the similarities are computed, we present reconstructions of the species content of the sample based on the similarity results. We reconstruct the putative species composition of the sample by looking at the phylogenetic origin of the database sequences hit by the similarity searches.

3.1.8 Annotation Mapping

Sequence similarity searches are computed against a protein database derived from the M5nr, which provides nonredundant integration of many databases. Users can easily change views without recomputating data. For example, COG and KEGG views can be displayed, which both show the relative abundances of histidine biosynthesis in a dataset of four cow rumen metagenomes.

Help in interpreting results: MG-RAST searches the nonredundant M5nr and M5RNA databases in which each sequence is unique. These two databases are built from multiple sequence database sources, and the individual sequences may occur multiple times in different strains and species (and sometimes genera) with 100% identity. In these circumstances, choosing the "right taxonomic information" is not a straightforward process. To optimally serve a number of different use cases, we have implemented three methods – best hit, representative hit, and lowest common ancestor – for end users to determine the number of hits (occurrences of the input sequence in the database) reported for a given sequence in their dataset.

- *Best hit*: The best hit classification reports the functional and taxonomic annotation of the best hit in the M5nr for each feature. In those cases where the similarity search yields multiple same-scoring hits for a feature, we do not choose any single correct label. For this reason MG-RAST double counts all annotations with identical match properties and leaves determination of truth to our users. While this approach aims to inform about the functional and taxonomic potential of a microbial community by preserving all information, subsequent analysis can be biased because of a single feature having multiple annotations, leading to inflated hit counts. For users looking for a specific species or function in their results, the best hit classification is likely what is wanted.
- *Representative hit*: The representative hit classification selects a single, unambiguous annotation for each feature. The annotation is based on the first hit in the homology search and the first annotation for that hit in the database. This approach makes counts additive across functional and taxonomic levels and thus allows, for example, the comparison of functional and taxonomic profiles of different metagenomes.
- *Lowest common ancestor* (*LCA*): To avoid the problem of multiple taxonomic annotations for a single feature, MG-RAST provides taxonomic annotations based on the widely used LCA method introduced by MEGAN [28]. In this method all hits are collected that have a bit score close to the bit score of the best hit. The taxonomic annotation of the feature is then determined by computing the LCA of all species in this set. This replaces all taxonomic annotations from ambiguous hits with a single higher-level annotation in the NCBI taxonomy tree.

3.1.9 Abundance Profiles

Abundance profiles are the primary data product that MG-RAST's user interface uses to display information on the datasets. Using the abundance profiles, the MG-RAST system defers making a decision on when to transfer annotations. Since there is no well-defined threshold that is acceptable for all use cases, the abundance profiles contain all similarities and require their users to set cutoff values. The threshold for annotation transfer can be set by using the following parameters: e-value, percent identity, and minimal alignment length.

The taxonomic profiles use the NCBI taxonomy. All taxonomic information is projected against this data. The functional profiles are available for data sources that provide hierarchical information. These currently comprise SEED subsystems, KEGG orthologs, and COGs.

The SEED subsystems represent an independent reannotation effort that powers, for example, the RAST [29] effort. Manual curation of subsystems makes them an extremely valuable data source. The page at http://pubseed.theseed.org//SubsysEditor.cgi allows browsing the subsystems.

Subsystems represent a four-level hierarchy:

1. Subsystem level 1 – highest level
2. Subsystem level 2
3. Subsystem level 3 – similar to a KEGG pathway
4. Subsystem level 4 – actual functional assignment to the feature in question

KEGG Orthologs. MG-RAST uses the KEGG enzyme number hierarchy to implement a four-level hierarchy. We note that KEGG data is no longer available for free download. We thus have to rely on using the latest freely downloadable version of the data:

1. KEGG level 1 – first digit of the EC number (EC:X.*.*.*)
2. KEGG level 2 – first two digits of the EC number (EC:X.Y.*.*)
3. KEGG level 3 – first three digits of the EC number (EC:X:Y:Z:.*)
4. KEGG level 4 – entire four digits of the EC number

The high-level KEGG categories are as follows:

1. Cellular processes
2. Environmental information processing
3. Genetic information processing
4. Human diseases
5. Metabolism
6. Organizational systems

COG and eggNOG Categories. The high-level COG and eggNOG categories are as follows:

1. Cellular processes
2. Information storage and processing
3. Metabolism
4. Poorly characterized

3.2 The rRNA Pipeline

This pipeline takes rRNA sequence data or WGS data (annotates only RNAs in WGS) that proceeds through the following steps.

3.2.1 Preprocessing

After upload, data is preprocessed by using SolexaQA to trim low-quality regions from FASTQ data. Platform-specific approaches are used for 454 data submitted in FASTA format: reads more than two standard deviations away from the mean read length are discarded following. All sequences submitted to the system are available, but discarded reads will not be analyzed further.

3.2.2 rRNA Detection

Reads are identified as rRNA through a simple rRNA detection step. An initial BLAT search against a reduced RNA database efficiently identifies RNA. The reduced database is a 90% identity clustered version of the SILVA database and is used merely to rapidly identify sequences with similarities to ribosomal RNA.

3.2.3 rRNA Clustering

The rRNA-similar reads are then clustered, using UCLUST, at 97% identity, and the longest sequence is picked as the cluster representative (abundance values are retained, however).

3.2.4 rRNA Identification

A BLAT similarity search for the longest cluster representative is performed against the M5RNA database, integrating SILVA [30], Greengenes [31], and RDP [32].

3.3 Using the MG-RAST User Interface

The MG-RAST system provides a rich web user interface that covers all aspects of the metagenome analysis, from data upload to ordination analysis. The web interface can also be used for data discovery. Metagenomic datasets can be easily selected individually or on the basis of filters such as technology (including read length), quality, sample type, and keyword, with dynamic filtering of results based on similarity to known reference proteins or taxonomy. For example, a user might want to perform a search such as “phylum eq. ‘actinobacteria’ and function in KEGG pathway Lysine Biosynthesis and sample in ‘Ocean’” to extract sets of reads matching the appropriate functions and taxa across metagenomes. The results can be displayed in familiar formats, including bar charts, trees that incorporate abundance information, heatmaps, or principal component analyses, or exported in tabular form. The raw or processed data can be recovered via download pages. Metabolic reconstructions based on mapping to KEGG pathways are also provided.

Sample selection is crucial for understanding large-scale patterns when multiple metagenomes are compared. Accordingly, MG-RAST supports MIxS and MIMARKS [33] (as well as domain-specific plug-ins for specialized environments not extending the minimal GSC standards); several projects, including TerraGenome, HMP (Human Microbiome Project), TARA (Oceans Science), and EMP (Earth Microbiome Project), use these GSC standards, enabling standardized queries that integrate new samples into these massive datasets.

One key aspect of the MG-RAST approach is the creation of smart data products enabling the user at the time of analysis to determine the best parameters for, for example, a comparison between samples. This is done without the need for recomputing results.

3.3.1 Navigation

The MG-RAST web site is rich with functionality and offers a lot of different options. The site at http://metagenomics.anl.gov has five main pages and a home page, shown in blue in Fig. 2:

- Download page – lists all publicly available data for download. The data is structured into projects.
- Browse page – allows interactive browsing of all datasets and is powered by metadata.
- Search page – allows identifier, taxonomy, and function-driven searches against all public data.
- Analysis page – enables in-depth analyses and comparisons between datasets.
- Upload page – allows users to provide their samples and metadata to MG-RAST.
- Home (overview) page – provides an overview for each individual dataset.

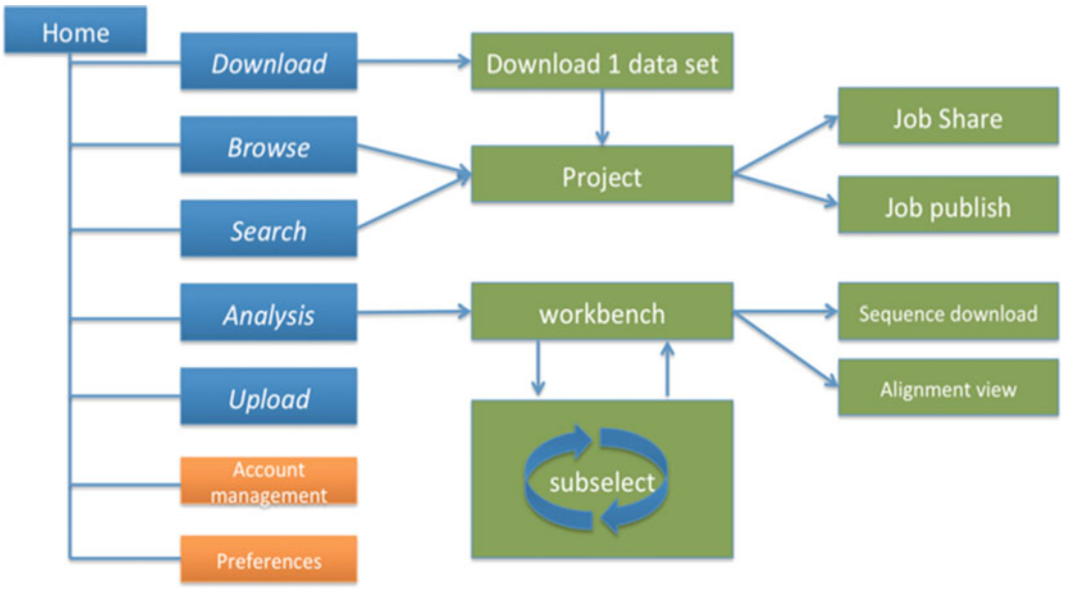


Fig. 2 Sitemap for the MG-RAST web site. On the site map, the main pages are shown in *blue* and management pages in *orange*. The *green boxes* represent pages that are not directly accessible from the home page

3.3.2 Upload Page

Data and metadata can be uploaded in the form of spreadsheets along with the sequence data by using both the ftp and the http protocols. The web uploader will automatically split larger files and allow parallel uploads. MG-RAST supports datasets that are augmented with rich metadata using the standards and technology developed by the GSC. Each user has a temporary storage location inside the MG-RAST system. This inbox provides temporary storage for data and metadata to be submitted to the system. Using the inbox, users can extract compressed files, convert a number of vendor-specific formats to MG-RAST submission-compliant formats, and obtain an MD5 checksum for verifying that transmission to MG-RAST has not altered the data. The web uploader has been optimized for large datasets of over 100 gigabasepairs, often resulting in file sizes in excess of 150 GB.

3.3.3 Browse Page: Metadata-Enabled Data Discovery

The browse page lists all datasets visible to the user. This page also provides an overview of the nonpublic datasets submitted by the user or shared with users. The interactive metagenome browse table provides an interactive graphical means to discover data based on technical data (e.g., sequence type or dataset size) or metadata (e.g., location or biome).

3.3.4 Project Page

The project page provides a list of datasets and metadata for a project. The table at the bottom of the project page provides access to the individual metagenomes by clicking on the identifiers in the first column. In addition, the final column provides downloads for metadata, submitted data, and the analysis results via the three labeled arrows. For the dataset owners, the project page provides an editing capability using a number of menu entries at the top of the page:

- Share Project – make the data in this project available to third parties via sending them access tokens.
- Add Jobs – add additional datasets to this project.
- Edit Project Data – edit the contents of this page.
- Upload Info – upload information to be displayed on this page.
- Upload MetaData – upload a metadata spreadsheet for the project.
- Export MetaData2 – export the metadata spreadsheet for this project.

3.3.5 Overview Page

MG-RAST automatically creates an individual summary page for each dataset. This metagenome overview page provides a summary of the annotations for a single dataset. The page is made available by the automated pipeline once the computation is finished. This page is a good starting point for looking at a particular dataset. It provides a significant amount of information on technical details and biological content. The page is intended as a single point of reference for metadata, quality, and data. It also provides an initial overview of the analysis results for individual datasets with default parameters. Further analyses are available on the analysis page.

Technical Details on Sequencing and Analysis

The overview page provides the MG-RAST ID for a dataset, a unique identifier that is usable as an accession number for publications. Additional information such as the name of the submitting PI and organization and a user-provided metagenome name are displayed at the top of the page as well. A static URL for linking to the system that will be stable across changes to the MG-RAST web interface is provided in the MG-RAST user manual found at: ftp://ftp.metagenomics.anl.gov/data/manual/mg-rast-manual.pdf.

MG-RAST provides an automatically generated paragraph of text describing the submitted data and the results computed by the pipeline. By means of the project information, we display additional information provided by the data submitters at the time of submission or later.

One of the key diagrams in MG-RAST is the sequence breakdown pie chart (Fig. 3) classifying the submitted sequences submitted into several categories according to their annotation status. As detailed in the description of the MG-RAST v3 pipeline above, the features annotated in MG-RAST are protein-coding genes and ribosomal proteins.

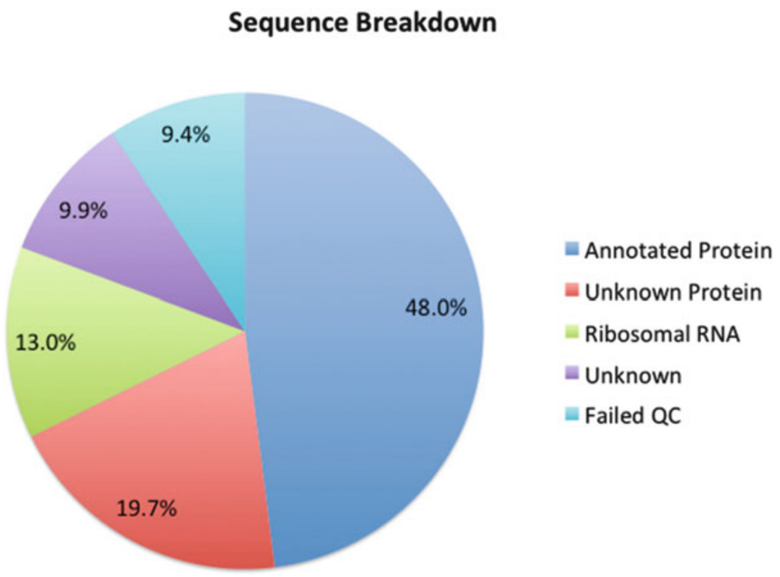


Fig. 3 Sequences to the pipeline are classified into one of five categories: *gray* = failed the QC, *red* = unknown sequences, *yellow* = unknown function but protein coding, *green* = protein coding with known function, and *blue* = ribosomal RNA. For this example, over 50% of sequences were either filtered by QC or failed to be recognized as either protein coding or ribosomal

Note that for performance reasons no other sequence features are annotated by the default pipeline. Other feature types such as small RNAs or regulatory motifs (e.g., CRISPRs [34]) not only will require significantly higher computational resources but also are frequently not supported by the unassembled short reads that constitute the vast majority of today's metagenomic data in MG-RAST.

The quality of the sequence data coming from next-generation instruments requires careful design of experiments, lest the sensitivity of the methods is greater than the signal-to-noise ratio the data supports.

The overview page also provides metadata for each dataset to the extent that such information has been made available. Metadata enables other researchers to discover datasets and compare annotations. MG-RAST requires standard metadata for data sharing and data publication. This is implemented using the standards developed by the Genomics Standards Consortium.

All metadata stored for a specific dataset is available in MG-RAST; we merely display a standardized subset in this table. A link at the bottom of the table ("More Metadata") provides access to a table with the complete metadata. This enables users to provide extended metadata going beyond the GSC minimal standards. A mechanism to provide community consensus extensions to the minimal checklists and the environmental packages is explicitly encouraged but not required when using MG-RAST.

Metagenome Quality Control

The analysis flowchart and analysis statistics provide an overview of the number of sequences at each stage in the pipeline. The text block next to the analysis flowchart presents the numbers next to their definitions.

Source Hits Distribution

The source hits distribution shows what percentage of the predicted protein features could be annotated with similarity to a protein of known function and which database those functions were from. In addition, ribosomal RNA genes are mapped to the rRNA databases.

The display will show the number of records in the M5nr protein database and in the M5RNA ribosomal databases.

Other Statistics

MG-RAST also provides a quick link to other statistics. For example, the analysis statistics and analysis flowchart provide sequence statistics for the main steps in the pipeline from raw data to annotation, describing the transformation of the data between steps. Sequence length and GC histograms display the distribution before and after quality control steps. Metadata is presented in a searchable table that contains contextual metadata describing sample location, acquisition, library construction, and sequencing using GSC compliant metadata. All metadata can be downloaded from the table.

3.3.6 Biological Part of the Overview Page

The taxonomic hit distribution display divides taxonomic units into a series of pie charts of all the annotations grouped at various taxonomic ranks (domain, phylum, class, order, family, genus). The subsets are selectable for downstream analysis; this also enables downloads of subsets of reads, for example, those hitting a specific taxonomic unit.

Rank Abundance

The rank abundance plot provides a rank-ordered list of taxonomic units at a user-defined taxonomic level, ordered by their abundance in the annotations.

Rarefaction

The rarefaction curve of annotated species richness is a plot (*see* Fig. 4) of the total number of distinct species annotations as a function of the number of sequences sampled. The slope of the right-hand part of the curve is related to the fraction of sampled species that are rare. On the left, a steep slope indicates that a large fraction of the species diversity remains to be discovered. If the curve becomes flatter to the right, a reasonable number of individuals are sampled: more intensive sampling is likely to yield only few additional species. Sampling curves generally rise quickly at first and then level off toward an asymptote as fewer new species are found per unit of individuals collected.

The rarefaction curve is derived from the protein taxonomic annotations and is subject to problems stemming from technical artifacts. These artifacts can be similar to the ones affecting amplicon sequencing [35], but the process of inferring species from protein similarities may introduce additional uncertainty.

Alpha Diversity

In this section we display an estimate of the alpha diversity based on the taxonomic annotations for the predicted proteins. The alpha diversity is presented in context of other metagenomes in the same project.

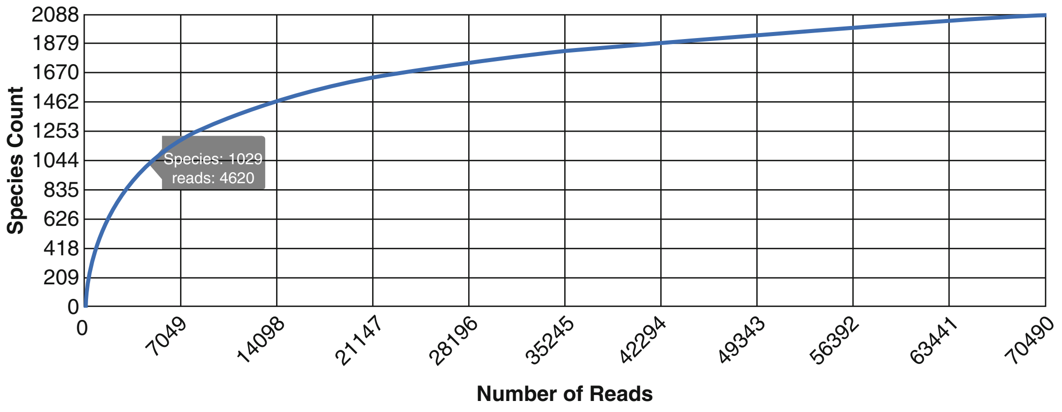


Fig. 4 Rarefaction plot showing a curve of annotated species richness. This curve is a plot of the total number of distinct species annotations as a function of the number of sequences sampled

The alpha diversity estimate is a single number that summarizes the distribution of species-level annotations in a dataset. The Shannon diversity index is an abundance-weighted average of the logarithm of the relative abundances of annotated species. We compute the species richness as the antilog of the Shannon diversity.

Functional Categories

This section contains four pie charts providing a breakdown of the functional categories for KEGG, COG, SEED subsystems, and eggNOGs. Clicking on the individual pie chart slices will save the respective sequences to the workbench. The relative abundance of sequences per functional category can be downloaded as a spreadsheet, and users can browse the functional breakdowns.

A more detailed functional analysis, allowing the user to manipulate parameters for sequence similarity matches, is available from the analysis page.

3.3.7 Analysis Page

The MG-RAST annotation pipeline produces a set of annotations for each sample; these annotations can be interpreted as functional or taxonomic abundance profiles. The analysis page can be used to view these profiles for a single metagenome or to compare profiles from multiple metagenomes using various visualizations (e.g., heatmap) and statistics (e.g., PCoA, normalization).

The page is divided into three parts following a typical workflow:

1. Data type
   Selection of an MG-RAST analysis scheme, that is, selection of a particular taxonomic or functional abundance profile mapping. For taxonomic annotations, since there is not always a unique mapping from hit to annotation, we provide three interpretations: best hit, representative hit, and lowest common ancestor. When choosing the LCA annotations, not all downstream tools are available. The reason is the fact that for the LCA annotations, not all sequences will be annotated to the same level: classifications are returned on different taxonomic levels.

   Functional annotations can be grouped into mappings to functional hierarchies or can be displayed without a hierarchy. In addition, the recruitment plot displays the recruitment of protein sequences against a reference genome. Each selected data type has data selections and data visualizations specific for it.
2. Data selection of sample and parameters. This dialog allows the selection of multiple metagenomes that can be compared individually or selected and compared as groups. Comparison is always relative to the annotation source, e-value, and percent identity cutoffs selectable in this section. In addition to the metagenomes available in MG-RAST, sets of sequences previously saved in the workbench can be selected for visualization.

3. Data visualization. Data visualization and comparison. Depending on the selected profile type, the profiles for the metagenomes can be visualized and compared by using bar charts, trees, spreadsheet-like tables, heatmaps, PCoA, rarefaction plots, circular recruitment plot, and KEGG maps.

The data selection dialog provides access to datasets in four ways. The four categories can be selected from a pull-down menu:

- Private data – list of private or shared datasets for browsing under available metagenomes.
- Collections – defined sets of metagenomes grouped for easier analysis. This is the recommended way of working with the analysis page.
- Projects – global groups of datasets grouped by the submitting user. The project name will be displayed.
- Public data – display of all public datasets.

When using collections or projects, data can also be grouped into one set per collection or project and subsequently compared or added.

Normalization

Normalization refers to a transformation that attempts to reshape an underlying distribution. A large number of biological variables exhibit a log-normal distribution, meaning that when the data is transformed with a log transformation, the values exhibit a normal distribution. Log transformation of the count data makes a normalized data product that is more likely to satisfy the assumptions of additional downstream tests such as ANOVA or t-tests. Standardization is a transformation applied to each distribution in a group of distributions so that all distributions exhibit the same mean and the same standard deviation. This removes some aspects of intersample variability and can make data more comparable. This sort of procedure is analogous to commonly practiced scaling procedures but is more robust in that it controls for both scale and location.

Rarefaction

The rarefaction view is available only for taxonomic data. The rarefaction curve of annotated species richness is a plot (*see* Fig. 5) of the total number of distinct species annotations as a function of the number of sequences sampled. As shown in Fig. 5, multiple datasets can be included.

The rarefaction curve is derived from the protein taxonomic annotations and is subject to problems stemming from technical artifacts. These artifacts can be similar to the ones affecting amplicon sequencing [31], but the process of inferring species from protein similarities may introduce additional uncertainty.

On the analysis page, the rarefaction curves are calculated from the table of species abundance. The curves represent the average number of different species annotations for subsamples of the complete dataset.

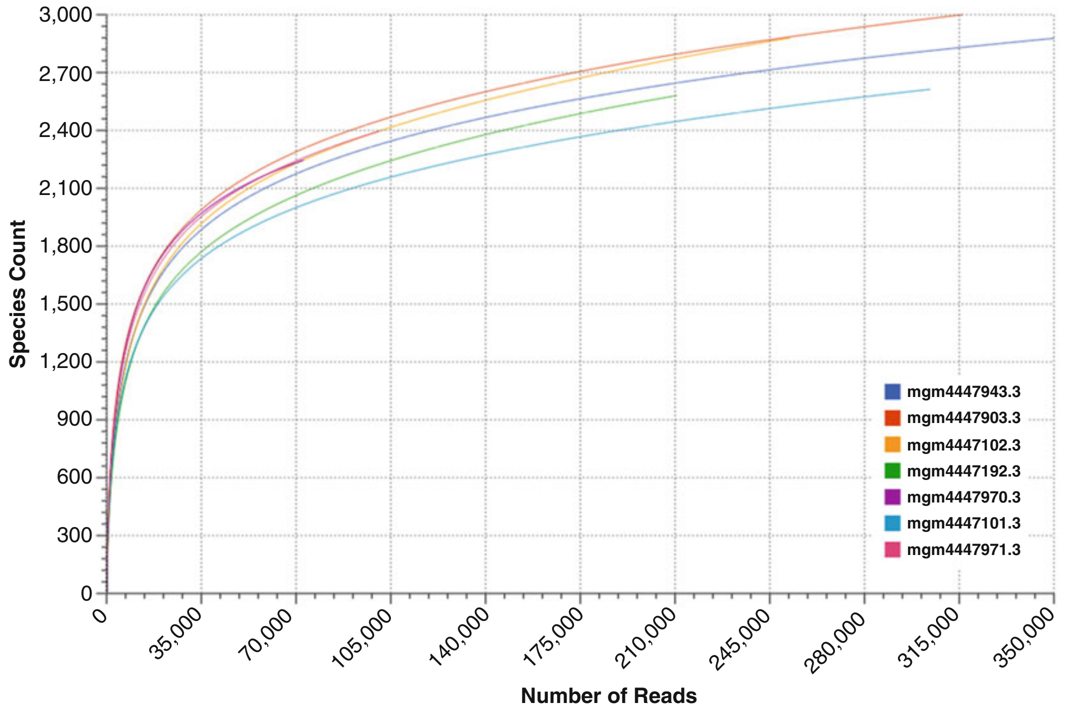


Fig. 5 Rarefaction plot showing a curve of annotated species richness. This curve is a plot of the total number of distinct species annotations as a function of the number of sequences sampled

Heatmap/Dendrogram

The heatmap/dendrogram allows an enormous amount of information to be presented in a visual form that is amenable to human interpretation. Dendrograms are trees that indicate similarities between annotation vectors. The MG-RAST heatmap/dendrogram has two dendrograms, one indicating the similarity/dissimilarity among metagenomic samples (x-axis dendrogram) and another indicating the similarity/dissimilarity among annotation categories (e.g., functional roles, the y-axis dendrogram). A distance metric is evaluated between every possible pair of sample abundance profiles. A clustering algorithm (e.g., ward-based clustering) then produces the dendrogram trees. Each square in the heatmap dendrogram represents the abundance level of a single category in a single sample. The values used to generate the heatmap/dendrogram figure can be downloaded as a table by clicking on the download button.

Ordination

MG-RAST uses principal coordinate analysis (PCoA) to reduce the dimensionality of comparisons of multiple samples that consider functional or taxonomic annotations. A key feature of PCoA-based analyses is that users can compare components not just to each other but also to metadata-recorded variables (e.g., sample pH, biome, DNA extraction protocol) to reveal correlations between extracted variation and metadata-defined characteristics of the samples.

It is also possible to couple PCoA with higher-resolution statistical methods in order to identify individual sample features (taxa or functions) that drive correlations observed in PCoA visualizations.

This coupling can be accomplished with permutation-based statistics applied directly to the data before calculation of distance measures used to produce PCoAs; alternatively, one can apply conventional statistical approaches (e.g., ANOVA or Kruskal-Wallis test) to groups observed in PCoA-based visualizations.

Bar Charts

The bar chart visualization option on the analysis page has a built-in ability to drill down by clicking on a specific category. You can expand the categories to show the normalized abundance (adjusted for sample sizes) at various levels. The abundance information displayed can be downloaded into a local spreadsheet. Once a subselection has been made (e.g., the domain *Bacteria* selected), data can be sent to the workbench for detailed analysis. In addition, reads from a specific level can be added into the workbench.

Tree Diagram

The tree diagram allows comparison of datasets against a hierarchy (e.g., subsystems or the NCBI taxonomy). The hierarchy is displayed as a rooted tree, and the abundance (normalized for dataset size or raw) for each dataset in the various categories is displayed as a bar chart for each category. By clicking on a category (inside the circle), detailed information can be requested for that node.

Table

The table tool creates a spreadsheet-based abundance table that can be searched and restricted by the user. Tables can be generated at user-selected levels of phylogenetic or functional resolution. Table data can be visualized by using Krona [36] or can be exported in BIOM format to be used in other tools (e.g., QIIME). The tables also can be exported as tab-separated text. Abundance tables serve as the basis for all comparative analysis tools in MG-RAST, from PCoA to heatmap/dendrograms.

Workbench

The workbench was designed to allow users to select subsets of the data for comparison or export. Specifically, the workbench supports selecting sequence features and submitting them to further analysis or other analysis. A number of use cases are described below. An important limitation with the current implementation is that data sent to the workbench exists only until the current session is closed.

3.3.8 Metadata, Publishing, and Sharing

MG-RAST is both an analytical platform and a data integration system. To enable data reuse, for example, for meta-analyses, we require that all data being made available to third parties contain at least minimal metadata. The MG-RAST team has decided to follow the minimal checklist approach used by the GSC.

MG-RAST provides a mechanism to make data and analyses publicly accessible. Only the submitting user can make data public on MG-RAST. As stated above, metadata is mandatory for dataset publication.

In addition to publishing, data and analysis can also be shared with specific users. To share data, users simply enter their email address via clicking sharing on the overview page.

References

1. Wilkening J, Wilke A, Desai N, Meyer F (2009) Using clouds for metagenomics: a case study. In: Cluster. IEEE Computer Society, pp. 1–6. ISBN: 978-1-4244-5012-1
2. Angiuoli S, Matalka M, Gussman A et al (2011) Clovr: a virtual machine for automated and portable sequence analysis from the desktop using cloud computing. BMC Bioinformatics 12:356
3. Meyer F, Paarmann D, D'Souza M et al (2008) The metagenomics RAST server – a public resource for the automatic phylogenetic and functional analysis of metagenomes. BMC Bioinformatics 9(1):386
4. Field D, Amaral-Zettler L, Cochrane G et al (2011) The genomic standards consortium. PLoS Biol 9(6), e1001088
5. Wilke A, Harrison T, Wilkening J et al (2012) The m5nr: a novel non-redundant database containing protein sequences and annotations from multiple sources and associated tools. BMC Bioinformatics 13:141
6. Altschul SF, Gish W, Miller W et al (1990) Basic local alignment search tool. J Mol Biol 215:403–410
7. Kent WJ (2002) Blat–the blast-like alignment tool. Genome Res 12(4):656–64
8. Brooksbank C, Bergman MT, Apweiler R et al (2014) The European bioinformatics Institute's data resources 2014. Nucleic Acids Res 42(Database issue):D18–25
9. Reference Genome Group of the Gene Ontology Consortium (2009) The gene ontology's reference genome project: a unified framework for functional annotation across species. PLoS Comput Biol 5(7):e1000431
10. Markowitz VM, Ivanova NN, Szeto E et al (2008) IMG/M: a data management and analysis system for metagenomes. Nucleic Acids Res 36(Database issue):D534–538
11. Kanehisa M (2002) The KEGG database. Novartis Found Symp 247:91–101
12. Benson DA, Cavanaugh M, Clark K (2013) Genbank. Nucleic Acids Res 41(Database issue):D36–42
13. Dwivedi B, Schmieder R, Goldsmith DB et al (2012) PhiSiGns: an online tool to identify signature genes in phages and design PCR primers for examining phage diversity. BMC Bioinformatics 4(13):37
14. Overbeek R, Begley T, Butler RM et al (2005) The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. Nucleic Acids Res 33(17):5691–5702
15. Magrane M, Uniprot Consortium (2011) UniProt knowledgebase: a hub of integrated protein data. Database (Oxford). doi:10.1093/database/bar009
16. Snyder EE, Kampanya N, Lu J et al (2007) PATRIC: the VBI pathosystems resource integration center. Nucleic Acids Res 35(Database issue):D401–406
17. Jensen LJ, Julien P, Kuhn M et al (2008) Eggnog: automated construction and annotation of orthologous groups of genes. Nucleic Acids Res 36(Database issue):D250–4
18. Tang W, Wilkening J, Desai N, Gerlach W, Wilke A, Meyer F (2013) A scalable data analysis platform for metagenomics. In: IEEE international conference on Big Data, IEEE, pp. 21–26
19. Cox MP, Peterson DA, Biggs PJ (2010) Solexaqa: at-a-glance quality assessment of illumina second-generation sequencing data. BMC Bioinformatics 11:485
20. Huse SM, Huber JA, Morrison HG et al (2007) Accuracy and quality of massively parallel DNA pyrosequencing. Genome Biol 8(7):R143
21. Gomez-Alvarez V, Teal TK, Schmidt TM (2009) Systematic artifacts in metagenomes from complex microbial communities. ISME J 3(11):1314–1317
22. Keegan KP, Trimble WL, Wilkening J et al (2012) A platform-independent method for detecting errors in metagenomic sequencing data: drisee. PLoS Comput Biol 8(6), e1002541
23. Langmead B, Trapnell C, Pop M et al (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10(3):R25

24. Trimble WL, Keegan KP, D'Souza M et al (2012) Short-read reading-frame predictors are not created equal: sequence error causes loss of signal. BMC Bioinformatics 13(1):183
25. Rho M, Tang H, Ye Y (2009) Fraggenescan: predicting genes in short and error prone reads. Nucleic Acids Res 38(20), e191
26. Edgar RC (2010) Search and clustering orders of magnitude faster than blast. Bioinformatics 26(19):2460–2461
27. Caporaso JG, Kuczynski J, Stombaugh J et al (2010) QIIME allows analysis of high-throughput community sequencing data. Nat Methods 7(5):335–336
28. Huson DH, Auch AF, Qi J et al (2007) Megan analysis of metagenomic data. Genome Res 17 (3):377–86
29. Aziz R, Bartels B, Best A et al (2008) The RAST server: rapid annotations using subsystems technology. BMC Genomics 9(1):75
30. Pruesse E, Quast C, Knittel K et al (2007) SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB. Nucleic Acids Res 35(21):7188–7196
31. DeSantis TZ, Hugenholtz P, Larsen N et al (2006) Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible with ARB. Appl Environ Microbiol 72 (7):5069–5072
32. Cole JR, Chai B, Marsh TL et al (2003) The ribosomal database project (RDP-II): previewing a new autoaligner that allows regular updates and the new prokaryotic taxonomy. Nucleic Acids Res 31(1):442–443
33. Yilmaz P, Kottmann R, Field D et al (2011) Minimum information about a marker gene sequence (MIMARKS) and minimum information about any (x) sequence (MIxS) specifications. Nat Biotechnol 29(5):415–420
34. Bolotin A, Quinquis B, Sorokin A et al (2005) Clustered regularly interspaced short palindrome repeats (CRISPRS) have spacers of extrachromosomal origin. Microbiology 151 (Pt 8):2551–2561
35. Reeder J, Knight R (2009) The 'rare biosphere': a reality check. Nat Methods 6 (9):636–637
36. Ondov BD, Bergman NH, Phillippy AM (2011) z. BMC Bioinformatics 12:385

Using QIIME to Evaluate the Microbial Communities Within Hydrocarbon Environments

Luke K. Ursell, Adam Robbins-Pianka, Nicole Scott, Antonio Gonzalez, Dan Knights, Jai Ram Rideout, Ajay Kshatriya, J. Gregory Caporaso, and Rob Knight

Abstract

Interest in the microbiology of hydrocarbon-rich environments has increased rapidly in recent years, with applications ranging from oil spill cleanup to reservoir management. Modern techniques for reading out microbial communities using high-throughput sequencing produce vast amounts of data that cannot be processed by traditional methods. Here, we describe the use of the QIIME (Quantitative Insights into Microbial Ecology) pipeline for analyzing hydrocarbon-rich samples. This pipeline starts with the raw data from high-throughput sequence analysis and produces graphical and statistical analyses suitable for publication or for decision support. We illustrate the use of the QIIME pipeline using several recent datasets including sediments and water from the Deepwater Horizon spill and crude oil from a hydraulically fractured oil field in the United States, allowing better understanding of the microbiology of these systems. QIIME is a free, open-source software and can be deployed on systems ranging from laptops to cloud computing environments.

Keywords: Hydrocarbons, Microbial ecology

Abbreviations

QIIME	Quantitative Insights into Microbial Ecology, canonically pronounced “chime.”
OTU	An operational taxonomic unit is a generic taxonomic grouping of organisms, similar to a phylum, genus, or species. When OTUs are defined based on gene sequence data, OTU definitions are generally based on sequence similarity. For example, in most sequence-based studies of microbial communities, 97% identity between a pair of ribosomal RNA sequences is used to group sequences into OTUs roughly corresponding to species-level groupings.

In recent years the clinical, biological, and ecological sciences have increasingly acknowledged the transformative role of microbial communities in their specific fields. The ability to actively monitor and manipulate microbial communities has shown promise for addressing both clinical and environmental dysbiosis [1]. Culture-independent methods using next-generation DNA sequencing allow much deeper analysis of the entire population of microbes in each sample (with the caveat that it is impossible to distinguish live cells from traces of DNA remaining in the environment). These techniques give us glimpses into a previously unknown biological world – the so-called long tail of the rank abundance distribution of microbial organisms or the "rare biosphere" [2]. However, computational analysis of the thousands to millions of sequences produced from each sample using these techniques requires improved methods of quality control, identification, annotation, statistical analysis, and visualization in order to interpret the results, especially for increasingly accessible study designs involving hundreds or thousands of samples. Here, we describe a pipeline and set of workflows known as Quantitative Insights into Microbial Ecology (QIIME; pronounced "chime") that enable a range of analyses to test and generate hypotheses about microbial sequence data [3].

The workflows for operational taxonomic unit (OTU) picking in QIIME, as described here and in more detail in [4], are useful for discovering new microbial taxa that may not be present in existing databases, which are typically biased toward medically relevant taxa [4]. Thus, environmentally collected communities with many undescribed members in samples (such as those collected through the Earth Microbiome Project (EMP; [5]) can be described and analyzed. The results of this work has shown the increased microbial diversity of oil-contaminated, terrestrial soil and stream water samples compared to many other environments, including marine and host-associated samples (EMP 2015, unpublished).

Until recently, the few microbial community-based investigations in oil-contaminated environmental samples have only used older methods of OTU identification, including clone-based sequencing methods [6–12] or Denaturing Gradient Gel Electrophoresis (DGGE) [13], limiting the ability to capture the full spectrum of genetic diversity. Despite these limitations, several of these studies specifically noted the increased diversity of microorganisms present in the oil-contaminated samples as compared to other environmental samples [8], especially in the oil portion as compared to the water phases of produced oil fluids [12]. The vast majority of sequences in oil-contaminated samples from these studies were attributed to *Gammaproteobacteria* [6, 8–10].

A handful of studies that employ high-throughput next-generation sequencing methods and analysis, many using the QIIME pipeline, have confirmed most of these findings, but have also gone beyond them by observing with higher resolution the shifts in microbial communities in hydrocarbon-contaminated samples. Of particular importance, and highlighting the need for better analysis methods, is comparison of sediment and water column microbiology during and after events such as the Deepwater

Horizon (DWH) oil spill in the Gulf of Mexico [14, 15]. The DWH oil spill began as a blowout from a deep water well (Macondo 252; MC252) and continued over the course of three months in April to June 2010. The spill was unique in that a dispersant (COREXIT 9500) was placed at the wellhead to break apart the oil particles and aid in bioremediation before the oil ran aground. The result of this dispersant application was the formation of a mid-water column oil plume at ~1,000–1,300 m [16]. This oil plume created complex spatial-temporal shifts in microbial community structure as lighter hydrocarbons were preferentially favored, and the microbial-accessible fraction shifted over time. For instance, early in the spill, the *Oceanospirillales*, capable of degrading cycloalkanes, dominated the oil plume; these were later replaced by *Oleispira* then *Colwellia* [17]. *Colwellia* also dominated aggregations and flocs of cells in enrichment cultures created from mixtures of MC252 oil and the COREXIT dispersant. OTU abundance frequency plots created in QIIME showed the shifts of *Colwelliaceae* in samples exposed to these different mixtures through time and in the flocs [17]. In other microcosm studies, *Colwellia* showed the potential for activity in ethane, propane, and benzene oxidation during the spill [18]. Methylotrophs dominated after the spill [19]. Further, in sediments taken three months after the spill ended, OTUs were identified in high abundance that matched uncultured gammaproteobacterium and *Colwellia* sequences previously identified in the oil plume, although *Oceanospirillales* prevalent in the plume had relatively low abundance in the sediments [14]. Understanding ecological succession during and after hydrocarbon spills is difficult because the composition of the "normal" oil-degrading community is unknown and "control" samples often inaccessible. Consequently, the ability to use QIIME as an analysis tool to evaluate microbial communities across environmental gradients in assessing both space and time, and to examine replicated patterns in response at different sites, is crucial.

1 Installing and Working with QIIME

The analyses mentioned above were performed using the Quantitative Insights into Microbial Ecology (QIIME) package [3]. QIIME is an open-source, GPL-licensed package of scripts written in Python 2 that either implement algorithms relevant to microbial ecology or wrap other tools' implementations (sometimes written in languages other than Python) to integrate them into a coherent microbial analysis suite.

QIIME runs natively in UNIX-like environments (including various Linux distributions and Mac OS X) primarily through the command line, although some results, such as beta diversity plots (*see* Sect. 2.2.3), taxa summary plots (*see* Sect. 2.2.5), and other visualizations, are presented through graphical interfaces such as web pages. Also, in some cases it may be convenient to use an external program with a graphical interface to generate input data files rather than rely solely on a terminal text editor (e.g., using a spreadsheet editor to generate and manage metadata *mapping files*; *see* Sect. 2.1). QIIME has also been integrated with Galaxy [20], which provides a web-based interface for interacting with QIIME.

Choosing a system for QIIME deployment depends primarily on two considerations: the size of the dataset to be analyzed and the availability of computing resources. When first experimenting with QIIME, the most convenient installation option is the QIIME virtual box (see http://qiime.org/install/virtual_box.html for more information). The virtual box comes preloaded with QIIME and all of its required dependencies (some optional dependencies with restrictive licenses are not included, but can be easily added). Although this convenience comes at the cost of slightly reduced performance compared with a native installation, and the virtual box is limited by the hardware capabilities of the host, it can be adequate for primary analysis if the dataset is small. Note that for users of non-UNIX-like operating systems (such as Microsoft Windows), the virtual box is currently the only option for local installation.

On the other end of the data size continuum, very large datasets require more and faster processors and greater memory capacity. In these cases, installation on a cluster (a private cluster, an institutional cluster, or a cloud-based compute cluster) is recommended. Many QIIME scripts that implement particularly time-consuming processing steps in an analysis can be run in parallel, and these environments will permit the greatest degree of parallelization and minimize memory limitations. Installation on a cluster differs little from a standard native installation, although some extra settings need to be configured to allow QIIME to communicate with the cluster's job queuing system; QIIME works most easily with Torque, Sun Grid Engine (SGE), or SLURM-based queues (see http://qiime.org/tutorials/parallel_qiime.html for more detailed instructions). For deployment on Amazon's Elastic Compute Cloud (EC2), an Amazon Machine Image (AMI) is made available for major releases of QIIME (see http://qiime.org/tutorials/working_with_aws.html for more details).

Most often, users' needs fall somewhere between these two extremes. A typical analysis will involve a number of samples that will fit on a single run of an Illumina MiSeq instrument. In these cases, a laptop or workstation will usually suffice. Like Linux and UNIX users, Mac OS X users can run QIIME natively since OS X is based on BSD, a UNIX-like operating system. One good option for OS X users is MacQIIME (http://www.wernerlab.org/software/macqiime), which conveniently packages QIIME and its main dependencies in an easily installable format. Alternatively, QIIME and its dependencies can be installed manually. The QIIME base install is achievable using *pip*, a Python package manager, and details on installing individual dependences are available online at http://qiime.org/install/install.html.Note that not every dependency is required for every analysis, and the base install is sufficient for the vast majority of users. To facilitate obtaining a complete native QIIME installation on Linux distributions, one can use qiime-deploy (https://github.com/qiime/qiime-deploy), which automatically fetches and installs many of QIIME's optional dependencies. Although QIIME wraps many third-party software packages, some of which can be tricky to install on certain systems, the vast majority of analyses can be handled by the QIIME base install, which is substantially easier to get running in a native environment.

1.1 Learning QIIME Hands-On

After installing QIIME, new users should work through one or more of the QIIME tutorials (http://qiime.org/tutorials). We recommend that new users start with the Illumina Overview Tutorial (http://qiime.org/tutorials/illumina_overview_tutorial.html) and the 454 Overview Tutorial (http://qiime.org/tutorials/tutorial.html). These tutorials introduce users to working with data generated on different platforms. While some topics are covered in both, they provide complementary background information and highlight different QIIME scripts, so working through both is likely to be beneficial to most new users. Next, users should begin working on their own datasets by adapting the steps in those tutorials to their own analysis. In most cases, the steps in the Illumina Overview Tutorial are what users are looking for for their "first-pass" analysis. (Note that raw Illumina data is provided in a variety of different formats. The preparing raw Illumina data tutorial (http://qiime.org/tutorials/processing_illumina_data.html) provides instructions on working with the most common of these different formats.) The other QIIME tutorials provide guidance on additional analyses that can be run using QIIME, including performing supervised classification of samples based on their microbiome profiles (http://qiime.org/tutorials/running_supervised_learning.html, a topic we reviewed in [21], and comparing categories of microbiome samples (http://qiime.org/tutorials/category_comparison.html). Finally, users looking for a comprehensive list of the scripts packaged with QIIME (and a few related packages) can refer to the QIIME script index (http://scripts.qiime.org).

Users looking for help with using QIIME should follow the instructions at Getting Help with QIIME (http://help.qiime.org) for getting technical support. Users who are also new to the UNIX command line are likely to find the Software Carpentry (http://software-carpentry.org) lessons and workshops extremely useful. Users interested in getting hands-on help with QIIME or interacting with the developers will likely be interested in attending a QIIME workshop (http://workshops.qiime.org).

2 Data Analysis

New QIIME users should be aware of a few important facts before starting their first analysis. First, the status of a QIIME installation can be inspected using the `print_qiime_config.py` script. Running this script will output the values of important configuration settings as well as versions of dependencies. One very useful feature of this script is its ability to run basic tests on the installation, available through the `-t` option, which will identify errors in the installation or missing dependencies required for a QIIME base install.

Second, access to the documentation of any script is available by running the script and passing the `-h` parameter:

```
print_qiime_config.py -h
```

which will display a brief description of the script's purpose, its available options, usage examples, and, where appropriate, additional background information about the technique implemented by the script. Similar documentation for the complete list of scripts available in the latest version of QIIME is accessible at the QIIME script index (http://scripts.qiime.org). Documentation for older versions of QIIME is also available; for example, documentation for QIIME version 1.8.0 is available at http://qiime.org/1.8.0/.

2.1 Working with Illumina Raw Data

To start working with QIIME, a mapping file containing all of the information about the samples that might be relevant to the analysis must be provided as a table in tab-separated text format. This should minimally include columns for the sample identifier (ID), the linker and primer sequences used during PCR, the sample barcode added during PCR, and a description of each sample. For the most up-to-date information on the required mapping file format, please see http://qiime.org/documentation/file_formats.html#metadata-mapping-files.

After creating a mapping file, it is important to validate it to ensure compatibility with QIIME. The following command will determine whether or not a metadata mapping file is compatible with QIIME and, if it is not, highlight the problems:

```
validate_mapping_file.py -m your_map_file.txt -o your_output_folder
```

The metadata mapping file ("mapping file" for short) is an extremely important component of an analysis. The required columns mentioned above are the *minimal* set of columns that QIIME requires in order to process data. However, the inclusion of additional user-provided columns is strongly encouraged, since discovering trends and correlations related to characteristics of the microbiome is contingent on having detailed information about the samples themselves. Also, recording additional information about the preparation of the samples (e.g., the lot numbers of the reagents used in DNA extraction) can help identify possible technical biases and other confounding factors.

The other required files for analysis are the raw sequence files, which come directly from the sequencing center. The preferred format for these raw reads is one FASTQ file containing the forward reads, one FASTQ file containing the reverse reads (if applicable), and one FASTQ file containing the index reads (also known as the barcode reads). However, there are a variety of formats returned by sequencing centers. For example, some sequencing centers instead return to their clients demultiplexed reads. In these cases, there may be only one (the forward reads) or two files (the forward reads and the reverse reads) depending on whether the sequencing used paired-end primers, and the index reads will have been moved into the header line of the sequences. Or, the index reads file will not be present if the sequencing center did not trim the barcodes from the reads. QIIME includes a script called `extract_barcodes.py`, which has an `--input_type` parameter that specifies how the sequencing center's files can be converted into QIIME's preferred format:

```
-c INPUT_TYPE, -input_type=INPUT_TYPE
                    Specify the input type. barcode_single_end: Input is a
                      single fastq file, that starts with the barcode
                    sequence. barcode_paired_end: Input is a pair of fastq
                    files (-fastq1 and -fastq2) that each begin with a
                    barcode sequence. The barcode for fastq1 will be
                    written first, followed by the barcode from fastq2.
                    barcode_paired_stitched: Input is a single fastq file
                    that has barcodes at the beginning and end. The
                    barcode from the beginning of the read will be written
                    first followed by the barcode from the end of the
                    read, unless the order is switched with
                    -switch_bc_order. barcode_in_label: Input is a one
                    (-fastq1) or two (-fastq2) fastq files with the
                    barcode written in the labels. [default:
                    barcode_single_end]
```

Note the `barcode_paired_stitched` option. The use of some paired-end primers together with sufficiently long read lengths can result in partially overlapping forward and reverse reads. Forward reads typically have higher quality than the reverse reads, so if using only one or the other, forward reads are typically preferred [22, 23]. The other option is to join, or stitch, the sequences together based on their overlapping region to form a longer read. Using longer reads such as joined paired-end sequences can yield more accurate taxonomic assignments [24]. To join sequences in QIIME using the default method *fastq-join* [25] and use both the forward and reverse reads together with the barcodes, you should run

```
join_paired_ends.py -f forward_reads.fastq -r reverse_reads.
fastq -b barcodes.fastq -o fastq-join_joined
```

After you have decided whether to use the forward, reverse, or joined reads, the next step is to demultiplex and quality filter the sequences. In QIIME, this demultiplexing and quality filtering step is called "split libraries," and by default the script has permissive quality filtering settings, which allows for more sequences to be retained without adversely affecting the biological conclusions generated from the analysis [26]. The basic command format for this step in QIIME is:

```
split_libraries_fastq.py -i input.fastq.gz -b input_barcode.
fastq.gz -o split_libraries/ -m your_map_file.txt
```

In this example, gzipped (compressed) versions of the input files are passed to the script, but note that you can also use uncompressed versions.

Some of the most commonly used options are related to correcting for read orientation. For example, one of the first two options below would be useful if the barcodes specified in the mapping file are reverse-complemented relative to the barcodes in the sequence file:

```
-rev_comp_barcode  reverse complement barcode reads before lookup
                   [default: False]
-rev_comp_mapping_barcodes
                    reverse complement barcode in mapping before
                    lookup (useful if barcodes in mapping file are
                    reverse complements of golay codes)
                    [default: False]
-rev_comp          reverse complement sequence before writing to
                   output file (useful for reverse-orientation
                   reads) [default: False]
```

The output of this script comprises three files: *seqs.fna*, which contains the sequences that passed quality filtering in FASTA format with modified headers and sequence labels amenable to processing by other QIIME scripts downstream; *histogram.txt*, a text file containing a histogram of read lengths before and after processing; and *split_library_log.txt*, which provides a summary of the input and output files, including data about the number of sequences that passed quality filtering and were successfully assigned to each sample. A manual review of *split_library_log.txt* should always be performed because it allows a user to check how many sequences were eliminated and for what reasons. Some examples of reasons that a sequence would be eliminated include final read length falling outside specified bounds, sequencing errors in the barcodes that prevent accurate assignment to the source sample, or contiguous regions of low-quality reads.

Generating this *seqs.fna* file is the first major step in conducting a QIIME analysis. The next section describes how to use QIIME to cluster demultiplexed sequences into operational taxonomic units (OTUs).

2.2 Analysis Pipeline

Note: QIIME has several different OTU picking methods and workflows that have evolved with each major release version. Here we describe the methods and workflows for QIIME 1.9.0, the current version as of this writing.

2.2.1 OTU Picking with QIIME

QIIME supports three high-level workflows for clustering sequences into operational taxonomic units (OTUs). These are typically referred to as de novo, closed-reference, and open-reference OTU picking and are described in detail in [4]. QIIME itself does not implement the sequence clustering processes that it uses, but rather provides Python wrappers invoked through the command line for C/C++ programs that implement these algorithms. Though this increases the complexity of installing QIIME, this strategy was taken for two important reasons. First, because sequence clustering is very computationally expensive, implementation in Python will be orders of magnitude slower than an equivalent implementation in C/C++ (or similar compiled programming languages). Second, it was not an efficient use of limited developer time to reinvent the wheel by re-implementing these methods because existing implementations had already been developed and were widely used and supported by active development teams. QIIME's wrappers offer users several options for OTU picking software, including UCLUST [27], USEARCH [27], CD-HIT [28], mothur [29], SortMeRNA [30], swarm [31], and sumaclust [http://metabarcoding.org/sumatra]. As of QIIME 1.9.0, UCLUST is the default OTU picker in QIIME, but it is unfortunately closed-source software with an expensive license agreement for commercial use. For this reason, we have added beta support for SortMeRNA, swarm, and sumaclust, three very promising open-source alternatives, and we expect that one or a combination of these will be the default in the next major release of QIIME. As of QIIME 1.9.0, these should be considered experimental, as they have only recently begun to be applied to real datasets by users not involved in their development.

In this section we briefly describe the three high-level workflows for OTU picking in QIIME and refer readers to a recent paper on this topic for additional detail [4]. In addition to OTU picking, the three workflows described below perform taxonomic assignment, sequence alignment, and phylogenetic reconstruction of OTU representative sequences and construct a BIOM-formatted [32] "OTU table," a two-dimensional matrix of OTU counts per sample.

`Pick_de_novo_otus.py` performs de novo OTU picking, meaning that the user's sequences are clustered only against one another, with no external reference. This approach was widely used when sequencing datasets were smaller (e.g., containing up to about 1 million sequences), but has become less popular with larger datasets because parallel implementations are not available. With modern-sized datasets, this workflow is often only used if a suitable reference database does not exist, because it can take a long time (e.g., weeks or months) to run on a single processor depending on the number of input sequences, their length, and the biological diversity of the samples. QIIME's `pick_de_novo_otus.py` workflow uses UCLUST by default for OTU picking, but its open-source alternatives swarm and sumaclust can easily be substituted.

`pick_closed_reference_otus.py` performs reference-based OTU picking against a user-provided reference database (the default in QIIME 1.9.0 is the Greengenes [33] 13_8 97% reference OTUs). In closed-reference OTU picking, reads are compared against sequences in the reference database and are assigned to OTUs based on their best hit or excluded from subsequent analyses if a suitable match (based on user-defined criteria, but 97% sequence similarity by default) cannot be found in the database. This method was widely used as sequence datasets began to grow in size, both with the initial applications of the 454 sequencer for amplicon surveys (until faster de novo cluster software became available [34]) and with the initial applications of the Illumina GAIIx for amplicon surveys (until subsampled open-reference OTU picking became available [4]. This approach was popular because it is trivially parallelizable, easily making use of multiprocessor workstations or computer clusters. Another advantage to this approach is that multiple OTU tables from different studies can be combined into a single analysis (called a meta-analysis) as long as the individual OTU tables were generated using the same reference database and other technical factors (such as read length) are controlled for. The drawback to the closed-reference protocol of course is that sequences that cannot be assigned to a reference database are discarded. For environments that are less well characterized (including oil and gas samples), organisms that play important roles in the systems may not be represented in reference databases, which therefore would lead to exclusion of important information from community surveys. QIIME's `pick_closed_reference_otus.py` uses UCLUST by default, but SortMeRNA is an open-source alternative.

`pick_open_reference_otus.py` is the currently preferred strategy for performing OTU picking in QIIME. It is essentially a combination of the previous two approaches. First, sequences are clustered against a reference database in parallel, using closed-reference OTU picking. However instead of excluding sequences that fail to hit the reference database from subsequent analyses, those sequences are clustered de novo. This therefore provides the benefits of both approaches: the closed-reference step can be run in parallel, dramatically reducing the total runtime of the analysis, and the de novo step ensures that even sequences that are not included in the reference database can be included in the analyses. Recent advances [4] have allowed for partial parallelization of the de novo step of this process by using a random subsample of the sequences that fail to hit the reference as a reference for the remainder. This approach has allowed the open-reference OTU picking strategy to be applied to even the largest of modern-sized datasets, such as sequences from 30,000 samples in the Earth Microbiome Project. QIIME's `pick_open_reference_otus.py` uses UCLUST by default, but SortMeRNA (for the closed-reference step) and sumaclust (for the open-reference step) are open-source alternatives.

Regardless of which high-level OTU picking algorithm is chosen from above, the result will be an OTU table. This contingency table takes the basic form of a matrix where samples form the columns and observations form the rows. "Observation" is a general term, the meaning of which depends on the data being represented in the table. For the 16S rRNA amplicon studies that are basis of this chapter, an observation is an OTU, but an observation could be a particular metabolite in a metabolome study or KEGG orthology group in a functional study. When this sample-by-observation matrix is represented in a tab-delimited text format, it is referred to as a "classic" OTU table because early sequencing studies had 1000s of sequences rather than millions, and samples numbered in the 10s rather than the 100s to 1000s, and plain text files would suffice. However, the high throughput of modern sequencing technologies and the sparse nature of resulting OTU tables (sparse because many OTUs will not be observed in most samples, resulting in contingency tables rife with zeroes) meant that a new, more computationally efficient way to store an OTU table was required. Therefore, rather than using tab-delimited text files as a storage medium for OTU tables, QIIME by default produces OTU tables in the BIOM format [32], which can store vast amounts of data in a space-efficient and computationally accessible format. The BIOM format also allows sample and observation metadata to be stored directly in the OTU table. This format is easily readable by the QIIME scripts and permits larger datasets to be stored far more computationally efficiently than a "classic" OTU table. However, being a binary format, BIOM tables in the HDF5 format are not human readable. To inspect a BIOM table manually, the BIOM convert command can be used to convert the table to tab-separated text. To get help with this command on a system with a working QIIME base install, you can run:

```
biom convert -h
```

This command gives the user options for converting between a classic OTU table and a BIOM-formatted table. For example, the command used to convert from a BIOM format to a classic format OTU table would be:

```
biom convert –i otu_table.biom –o otu_table_classic.txt –to-tsv –
header-key taxonomy
```

The `--header-key` option informs the script that the observation metadata to be output to the classic table is held under a "taxonomy" entry in the BIOM table, and therefore the taxonomic lineages will appear in a new column in the classically formatted OTU table that is generated. Note that a classic table can have only one column for observation metadata, while the BIOM format table can have an arbitrary number of observation metadata entries. Conversely, the user can convert back to a BIOM-formatted table with the command:

```
biom convert -i otu_table_classic.txt -o otu_table.biom –to-hdf5 –
table-type "OTU table" –process-obs-metadata taxonomy
```

A summary of the contents of a BIOM-formatted OTU table (e.g., the number of samples, the number of observations, and the total sequence count) can be obtained as follows:

```
biom summarize-table –i otu_table.biom
```

The output is a summary that will contain information such as the number of samples, the number of observations (OTUs), and the number of sequences associated with each sample. The summary file may reveal that some samples have zero or very few sequences associated with them in the table. This situation might arise for a couple different reasons. For example, the sample may have had very few sequences associated with it after demultiplexing and quality filtering, which would be indicated in the `split_libraries_log.txt`. This can happen if the sample failed to amplify in PCR or if the majority of that sample's sequences were eliminated. Alternatively, if the sample did have sequences assigned to it after quality filtering and demultiplexing, but the sample has no observations associated with it in the OTU table, then those sequences were not assignable to any OTU. Failure to assign to an OTU is a common occurrence in closed-reference OTU picking, especially for samples taken from poorly characterized environments where there may be a large number of OTUs not present in the reference database, which is part of the reason that the open-reference OTU picking protocol is recommended.

The OTU table summary file might also reveal that there is a surprisingly large number of OTUs. This is especially common when using the open-reference or de novo OTU picking protocols. Samples with very few associated observations and OTUs with very few associated sequences can hinder downstream analyses, especially statistical tests, and may need to be eliminated. Both of these situations can be addressed using filtering scripts.

A common practice for downstream QIIME analysis is to remove low-abundance OTUs from the OTU table. This occurs for a number of reasons. One is that if an OTU is represented by only a few sequences in an entire dataset, the likelihood that it is a spurious observation due to sequencing error is relatively higher than OTUs represented by a large number of sequences. Filtering out these extremely low-abundance, and possibly spurious, OTUs will decrease the penalty for multiple hypothesis testing in otc:

```
filter_otus_from_otu_table.py
```

As an example, "singletons," OTUs represented by only a single sequence in the dataset, are often assumed to have been the result of sequencing error and are typically removed. OTUs that are observed only a single time may be of biological relevance, but may confound downstream statistical tests and increase run time. The script's options for removing OTUs based on their total count is `-n` and based on the number of samples that they appear in is `-s`. For example, the following command will remove OTUs that are represented by fewer than 100 sequences or were observed in less than two samples. Note that the output OTU table is renamed slightly to include a suffix containing an abbreviated description of what was done to produce the table:

```
filter_otus_from_otu_table.py -i otu_table.biom -o otu_table_
n100_s2.biom –n 100 –s 2
```

Another filtering script is used to alter the samples that are included in the OTU table. For example, if only samples labeled as "Oil" in the mapping file's Sample_Type column are to be retained, then the command would be:

```
filter_samples_from_otu_table.py –i otu_table_full.biom –o otu_-
table_oil.biom –m mapping_file.txt –s "Sample_Type:Oil"
```

Samples can also be filtered by sample ID using the `--sam-ple_id_fp` option and passing in a text file containing a list of sample IDs, one per line:

```
filter_samples_from_otu_table.py –i otu_table.biom –o otu_table_-
filtered.biom –sample_id_fp ids_to_keep.txt
```

A powerful new feature implemented in QIIME 1.9.0 is the ability to collapse groups of samples into one logical sample. For example, if ten samples were collected from each of three oil wells

(Well_1, Well_2, and Well_3), and the well of origin for each sample was recorded in a mapping file column called "OilWell," each group of 10 samples could be combined to produce a single "sample" per oil well. The collapse_samples.py script accomplishes this and has a number of methods for determining the value of each OTU to be included in the collapsed sample. For instance, the script could choose a random sample to represent that well.

```
collapse_samples.py -i otu_table.biom -m mapping_oilwells.txt -
output_biom_fp   otu_table_wells_collapsed.biom   -output_map-
ping_fp mapping_wells_collapsed.txt -collapse_fields OilWell -
collapse_method random
```

The resulting OTU table will now list Well_1, Well_2, and Well_3 as the sample IDs, and the mapping file will also have this "collapsed" sample ID listed, along with the combined metadata for each of the 10 samples that were combined. Alternatively, the script could use the mean, median, or sum of each OTU count to represent the collapsed sample or first sample in the group. The script also provides functionality to normalize OTU abundances such that the sum of each sample's OTU abundances equals 1.0. This type of table is called a "relative abundance table" because it records not the actual counts of each OTU in a sample, but rather the fraction of a sample's total number of a observations contributed by each OTU.

```
collapse_samples.py -i otu_table.biom -m mapping_oilwells.txt -
output_biom_fp   otu_table_wells_collapsed.biom   -output_map-
ping_fp mapping_wells_collapsed.txt -collapse_fields OilWell -
collapse_method sum -normalize
```

The `collapse_samples.py` script is also immensely helpful for collapsing technical replicates into one sample for downstream analysis.

2.2.2 Alpha Diversity and Rarefaction

Alpha diversity is a measure of the amount of within sample diversity (usually either richness, evenness, or a combination of the two). There are a variety of ways to measure alpha diversity, such as counting the number of observed species or quantifying the amount of a phylogenetic tree that is represented by OTUs in a sample. In general, different measures of diversity provide different, but complementary, insights into the structure of a microbial community. QIIME includes numerous alpha diversity measures for your samples, which can all be explicitly listed by calling:

```
alpha_diversity.py -s
```

In the default QIIME pipeline, observed species, phylogenetic diversity, and Chao1 are used to calculate alpha diversity. To find out more information concerning about QIIME's alpha diversity metrics, see http://scikit-bio.org/docs/latest/generated/skbio.diversity.alpha.html.

Most high-throughput sequencing technologies result in a variable number of sequences per sample. This is a problem for measurements of diversity, because one would automatically expect to encounter more kinds of organisms with more sampling (e.g., if you collect one hundred sequences from an environment, it is impossible to detect more than one hundred species, but if you collect one thousand sequences from an environment, you may identify hundreds of species). Rarefaction is the process of subsampling without replacement to the same number of sequences per sample. (This process is under debate for use in microbial ecology, but some normalization is essential for comparing samples directly; [35].) One common use of rarefaction is for generating alpha rarefaction curves, for understanding how measures of alpha diversity change with sampling depth (Fig. 1).

QIIME can generate alpha rarefaction plots using its alpha_rarefaction.py workflow. The full OTU table will be rarefied to several depths between a defined min and max number of sequences per sample, alpha diversity computed for each, and the results summarized in a single plot. Figure 1 is an example of two output plots produced through the following command:

```
alpha_rarefaction.py -i otu_table.biom -m mapping_oilwells.txt -
o arare_out/ -e 5000 -min_rare_depth 1000 -n 10
```

Using this command, we have specified that the minimum number of sequences per sample to rarefy at is 1,000 and the max number of sequences per sample to rarefy at is 5,000, and we would like to rarefy at 10 steps between those min and max boundaries.

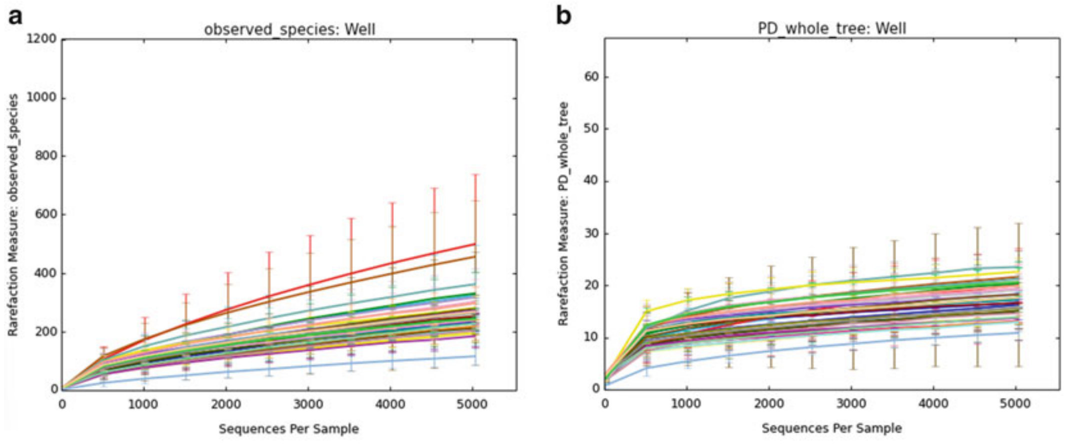


Fig. 1 Alpha diversity rarefaction curves of oil well samples measured by (**a**) Observed species and (**b**) PD whole tree. Each *colored line* represents the alpha diversity of an oil well across a range of sequences per sample. Standard error bars are added to each line based on performing 10 iterations of rarefaction at each depth. The leveling off of the curves suggests that most samples are nearing complete sampling, meaning that more sequences do not translate into higher measures of diversity, by 5,000 sequences per sample. In practice, the slope of these curves does not often hit zero, as it is still very difficult to differentiate sequencing error from real sequences that are observed in low abundance

2.2.3 Beta Diversity

Beta diversity measures differences in the microbial communities between two or more samples and produces a distance between the samples using a specified distance metric. Broadly, there are two types of distance metrics: phylogenetically aware and phylogenetically naive. Common phylogenetically naive distance metrics include Bray-Curtis, Hellinger, and Euclidean, whereby the distances between samples are calculated using only the OTU abundances in each sample, or only the presence or absence of the OTUs in each sample, regardless of the abundances. A drawback of using these metrics to analyze microbial communities is that they assume a "star phylogeny," where all OTUs are phylogenetically equidistant from one another. In most cases, a more accurate phylogeny relating the OTUs is known or can be generated, the use of which will yield a more biologically meaningful measure of microbial diversity in a sample. The most popular phylogenetically aware distance metric for microbial ecology is the UniFrac metric [36]. UniFrac uses a phylogeny, represented as a tree, to measure the fraction of the tree's total branch length that is unique to the OTUs present in each sample. UniFrac can be calculated either weighted, where the abundances of OTUs are taken into account, or unweighted, where the abundances are ignored beyond presence/absence. In the case of QIIME's closed-reference OTU picking protocol, the tree from the reference database should be used. When using the results of QIIME's open-reference or de novo OTU picking protocols, one of the outputs is a phylogenetic tree that incorporates all of the OTUs, whether those OTUs are reference OTUs or de novo OTUs. The full list of metrics that QIIME can calculate can be listed by:

```
beta_diversity.py -s
```

The output of the beta diversity script is a pairwise (sample-by-sample) distance matrix, where each entry in the matrix represents the distance between two samples. Note that the output matrix will be symmetric since the distance between sample X and sample Y is the same as the distance between sample Y and sample X. If the analysis has n samples, then the generated distance matrix defines a space with $n-1$ dimensions. It is common practice to use Principal Coordinates Analysis (PCoA), a dimensionality reduction technique, to visualize the distances between samples in a two- or three-dimensional space. Calculating the beta diversity for an OTU table, reducing the dimensionality of the matrix, and visualizing the samples in reduced-dimension space can be accomplished with the `beta_diversity_through_plots.py` workflow:

```
beta_diversity_through_plots.py -i otu_table.biom -m mapping.txt
-t gg_13_8_otus/trees/97_otus.tree -o bd_out/
```

Emperor [37] is used to visualize the PCoA plots. This command will run each of the individual steps of the workflow using default parameters, but a parameter file can be generated that changes the behavior of the individual scripts (e.g., to calculate additional distance metrics) and passed to `beta_diversity_through_plots.py` using the `-p` option. Also, this script can be run in parallel when working with a multicore computer or cluster:

```
beta_diversity_through_plots.py -i otu_table.biom -m mapping.txt
-t gg_13_8_otus/trees/97_otus.tree -o bd_out/ -a -O 10
```

Here, we have run the same beta diversity command in parallel (`−a` flag) using 10 parallel jobs (`−O 10`).

By default, this workflow script will calculate weighted and unweighted UniFrac (hence, a phylogenetic tree must be supplied using the `-t` option) and therefore will produce plots illustrating both weighted and unweighted UniFrac distances. The Emperor plots can be visualized by opening the `index.html` file that is in the output directory. Because the script was supplied a mapping file, we can color our samples in PCoA space by their metadata. Emperor readily allows coloring of both continuous and categorical variables to produce publication-quality figures (Fig. 2).

As seen above, there appears to be a gradient in PCoA space that correlates with higher levels of gas and oil production. PCoA therefore represents a very powerful tool for interrogating your data and determining which statistical tests would be appropriate to further verify findings.

2.2.4 Statistical Tests

QIIME incorporates a number of routinely used statistical tests that users may find helpful for their analysis. It is important to note here, however, that your data can and should be plugged into other statistical tools if they help you with your analysis and that QIIME is not exhaustive in its statistical offerings. For example, you may find a package in R that is appropriate for a test you wish to conduct on your data. In that case, you could load your BIOM table into R and then perform that analysis outside of QIIME. That being said, one classical ecological test is to see if the biological differences between samples correlate with a physical parameter, for instance, if UniFrac distances correlate with the pH differences in oil samples. To accomplish this, you could use QIIME's `compare_distance_matrices.py` script, which compares two distance matrices with a Mantel test. From the beta diversity analysis above, you already have a UniFrac distance matrix. To compare this to differences in pH, you'll also need a matrix of differences in pH between samples. To generate that from a mapping file containing a pH column, you can use the `distance_matrix_from_mapping.py` script:

```
distance_matrix_from_mapping.py –i mapping_oilwells.txt –c pH –o
oilwell_ph_dm.txt
```

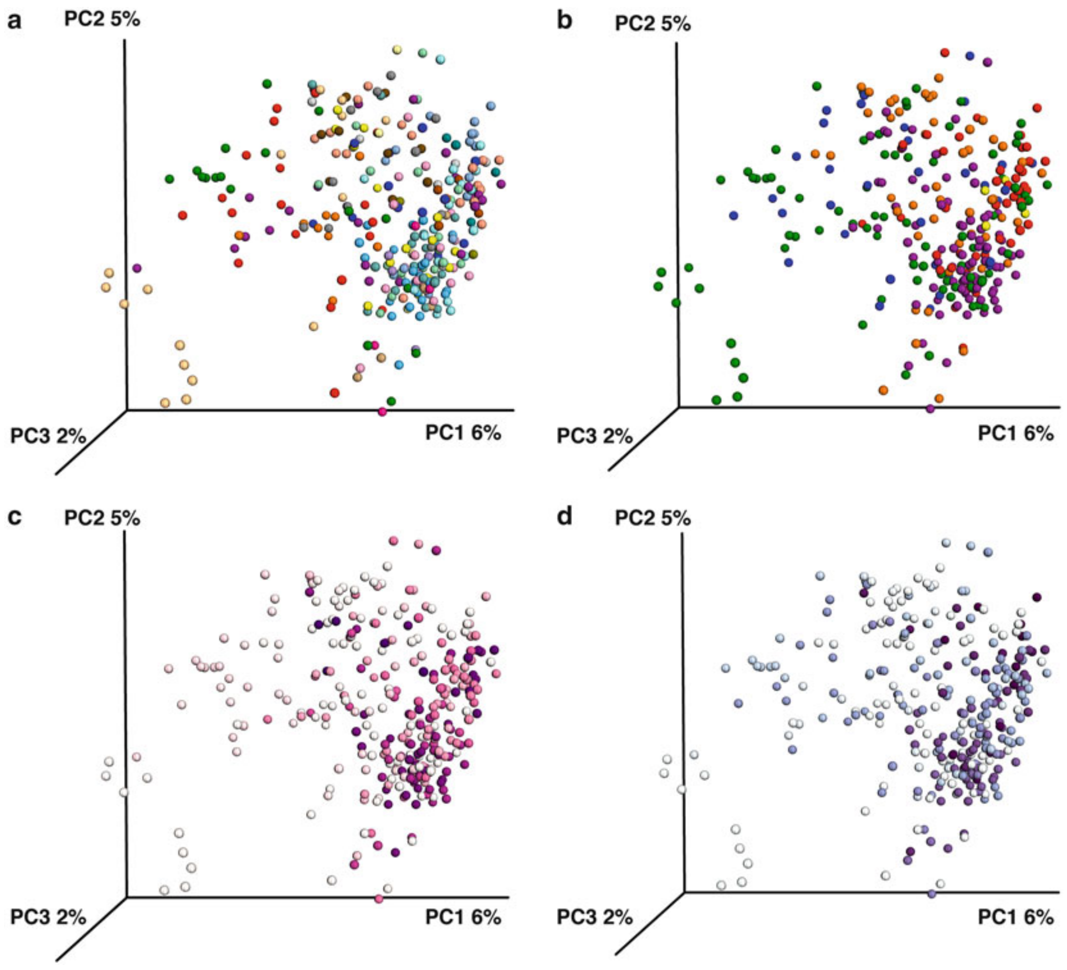


Fig. 2 PCoA plots of representative oil wells. The same set of oil well samples has been colored by different sample-associated metadata in Emperor. (**a**) Samples are colored by the individual oil well they originate from. (**b**) Samples are colored by the interval or comingled intervals the oil is likely being produced from. (**c**) Samples are colored along a gradient from *white* to *purple* that represents low to high gas production. (**d**) Samples are colored along a gradient from *white* to *blue* that represents low to high oil production

The resulting distance matrix can be compared to the UniFrac distance matrix as follows:

```
compare_distance_matrices.py  -i  oilwell_ph_dm.txt,unweighte-
d_unifrac.txt –method mantel -o mantel_out -n 999
```

The output directory will contain a file called `mantel_results.txt`, providing the Mantel *r* statistic, *p*-value, number of permutations, and tail type for the test. The *p*-value is determined non-parametrically using a Monte Carlo simulation, and the number of permutations can be adjusted through the `-n` flag. Above, we used 999 permutations. More permutations will provide a more precise *p*-value but will require more time to compute. Correlations between two distance matrices can also be calculated controlling for the effect of a third "control" matrix via a partial Mantel test. One example might be correlating oil well biological distances to pH alterations, controlling the physical distances between the wells:

```
compare_distance_matrices.py -i oilwell_ph_dm.txt,unweighte-
d_unifrac.txt -method partial_mantel -c physical_distances_dm.
txt -n 999
```

Another set of classic community ecology statistics used in QIIME is ADONIS and ANOSIM, which are based off of the ANOVA family of methods, where the observed variance in one group is compared to another. QIIME doesn't implement these methods directly, but rather wraps implementations from the R vegan package [38]. It is important to note here that ADONIS makes assumptions about the underlying distribution of the data and the independence of the metadata categories that 16S data frequently violates. Even though the *p*-values are computed non-parametrically, it is critical to assess both the *p*-value and the R^2 value when making conclusions. In this case, we might want to assess if the distances within an oil well are significantly less than the distances between oil wells:

```
compare_categories.py -i unweighted_unifrac_dm.txt -m mappin-
g_oilwells.txt -c OilWell -n 999 -method adonis -o
adonis_out_oilwells
```

The ADONIS method will provide the R^2 value and the non-parametrically calculated *p*-values. The R^2 value describes the amount of total variation between the samples that is attributed to differences between oil wells.

2.2.5 Taxa Summary Plots

Given that the OTU table contains taxonomic information, it is frequently helpful to visualize the relative abundance of different taxonomic groups across samples and across categories in your mapping file. This can be accomplished using the `summarize_-taxa_through_plots.py` script. The output of this file will be bar charts and area charts that show the relative breakdown of taxonomic groups across your samples. The Greengenes taxonomic convention refers to "levels" to denote taxonomic hierarchy; level 1 is kingdom, level 2 is phylum, etc., and level 7 is species. By default, levels 2 through 6 are included in the output taxa summary plots.

```
summarize_taxa_through_plots.py -i otu_table_oil_and_wellcut-
tings.biom -o sum_taxa_out
```

Figure 3 shows a typical bar chart that results from running this script. Here, each bar represents the relative abundances of phyla within a sample. Depending on the number of OTUs in your dataset, you might find that summary plots become hard to

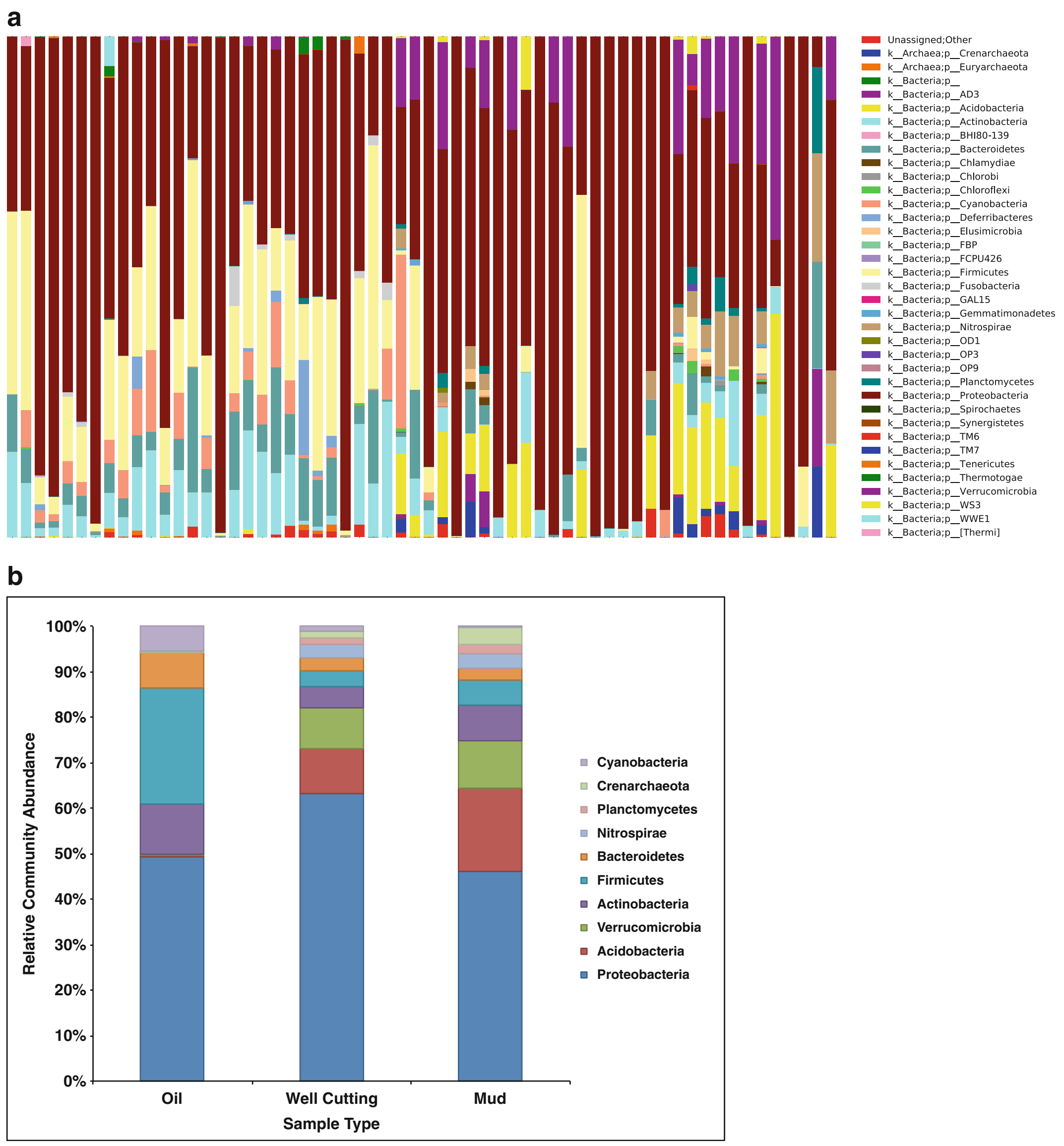


Fig. 3 Taxa summary plots. (**a**) A default taxonomy graph produced by QIIME. In this graph each bar represents a unique oil, well cutting, or mud sample. A taxonomic legend is also produced. (**b**) Taxa summary plot where samples have been summarized by sample type and therefore there are only columns for oil, well cutting, and mud. In this case, the raw data was transferred to another program for further customization

interpret. Therefore, the raw data (relative abundance of each taxonomic level per sample) is also included in the output, so that the user can more manually manipulate and filter to what they wish to see.

Instead of viewing each individual sample, it is common to want to know what the average relative abundance of a type of sample looks like. In our well samples, we have oil, well cuttings, and mud samples. Let’s summarize using only these groups:

```
summarize_taxa_through_plots.py -i otu_table_oil_and_wellcuttin-
gs.biom -o sum_taxa_by_type_out/ -m mapping_oil_wellcuttings.txt -
c SampleType
```

2.2.6 Source Tracker Analysis

Another common goal in microbial ecology studies is identifying sources of contamination, or, in more general terms, understanding the contribution of several potential “source” communities to a single “sink” community. For example, you might consider your hand to be a “sink” community that is comprised of some combination of microbes that you acquire from touching different “source” communities, such as soil, saliva, or feces, which all harbor distinct microbial communities. It is good practice to sample not only your community of interest but also the communities that might be resulting in some level of contamination, for instance, the hands of the sample collectors, any communities on the equipment being used, general lab space, etc. In this way, possible contamination issues can be elucidated. To accomplish this, we encourage the use of SourceTracker [39], which takes QIIME-formatted files as input and can assign likelihood of contamination from source communities in user-defined sink communities. This tutorial only allows for a brief introduction into SourceTracker.

Here we’ll apply SourceTracker to identify any possible sources of contamination in our collection of oil samples. To accomplish this, our oil sample table has been merged with an OTU table containing samples from many different possible contamination sources, including human skin, feces, saliva, water, and soil sources. SourceTracker will use the relative abundances of OTUs found in our contamination “sources” and the frequency at which pairs of OTUs are found together in each environment, to estimate what proportion of the composition of each of our four oil samples can be attributed to each “source” (Fig. 4). SourceTracker also assumes that there is an unknown “source” that can contribute OTUs. The SourceTracker command (which assumes a working installation of SourceTracker) is:

```
Rscript $PWD/sourcetracker_for_QIIME.r -i otu_table -m mapping_-
sourcetracker.txt
```

3 Conclusions

In this chapter, we have provided an overview of some of the important considerations in conducting analysis of hydrocarbon-relevant samples, primarily using published work on the Deepwater Horizon spill (much of the analysis of which was conducted with

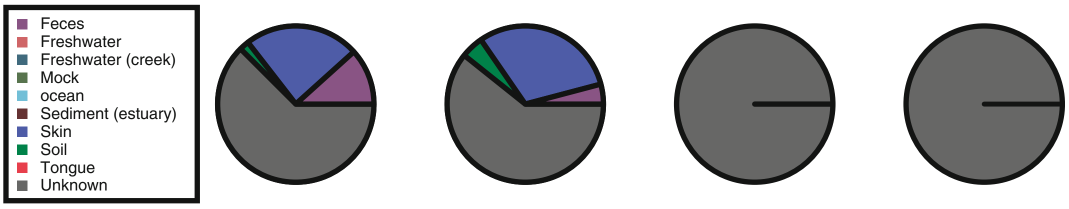


Fig. 4 SourceTracker analysis reveals proportion of each sample that likely comes from defined sources. Each of the pie charts represents a unique oil sample. The relative contribution of each of the "sources" is revealed as proportions of the pie with colors that matches up to the legend. This demonstrates that the majority of our oil samples come from the "unknown" source, meaning not one of the source sample types provided as input. The first two samples contain some fraction of "skin" as a source, indicating possible contamination of those samples with human skin microbes, possibly from the person performing the sampling or the lab work

QIIME) and with our own unpublished data from a hydraulically fractured oil field relating oil field characteristics such as production to microbiology. We have focused here on taxonomic analysis using amplicons from the 16S ribosomal RNA gene, although many other analyses including fungal analysis with ITS, and shotgun metagenomic and metatranscriptomic analyses, are also possible in QIIME.

The basic workflow that is essential for understanding data involves demultiplexing, OTU picking, taxonomy assignment, alpha and beta diversity analysis and visualization, and source tracking. These represent a small subset of the types of analysis that are possible in QIIME, and readers are strongly encouraged to investigate additional features as documented in the online tutorial.

Although, as this book demonstrates, the use of microbiology is rapidly expanding in the oil and gas field, there is still a long way to go before the richness of the datasets and the complexity of the study designs match what is currently being done in biomedical settings (in part, because the NIH-funded Human Microbiome Project provided much of the impetus for tool development). Consequently, applying techniques primarily pioneered in the biomedical arena will continue to play an important role in using microbial communities for biofuels, reservoir management, and bioremediation into the foreseeable future. In some ways, a bioreactor, a well, or a spill site can be seen as a patient, and the same types of precision medicine techniques that are poised to revolutionize healthcare can be applied directly. In others ways, environmentally focused studies will be more comparable in design and analysis techniques, or special-purpose tools for the oil and gas field may be required. By using general-purpose microbial analysis packages such as QIIME, which rely on the contributions of many different types of human- and ecologically focused projects for inspiration, it is possible to benefit from many of the lessons learned by exploring microbes in other environments in order to make rapid progress in systems relevant to energy.

References

1. Goodrich JK, Di Rienzi SC, Poole AC, Koren O, Walters WA, Caporaso JG, Knight R, Ley RE (2014) Conducting a microbiome study. Cell 158(2):250–262. doi:10.1016/j.cell.2014.06.037
2. Reeder J, Knight R (2009) The 'rare biosphere': a reality check. Nat Methods 6 (9):636–637. doi:10.1038/nmeth0909-636
3. Caporaso JG, Kuczynski J, Stombaugh J, Bittinger K, Bushman FD, Costello EK, Fierer N, Peña AG, Goodrich JK, Gordon JI, Huttley GA, Kelley ST, Knights D, Koenig JE, Ley RE, Lozupone CA, McDonald D, Muegge BD, Pirrung M, Reeder J, Sevinsky JR, Turnbaugh PJ, Walters WA, Widmann J, Yatsunenko T, Zaneveld J, Knight R (2010) QIIME allows analysis of high-throughput community sequencing data. Nat Methods 7 (5):335–336. doi:10.1038/nmeth.f.303, Epub 2010 Apr 11
4. Rideout JR, He Y, Navas-Molina JA, Walters WA, Ursell LK, Gibbons SM, Chase J, McDonald D, Gonzalez A, Robbins-Pianka A, Clemente JC, Gilbert JA, Huse SM, Zhou HW, Knight R, Caporaso JG (2014) Subsampled open-reference clustering creates consistent, comprehensive OTU definitions and scales to billions of sequences. PeerJ 2:e545. doi:10.7717/peerj.545, eCollection 2014
5. Gilbert JA, Jansson JK, Knight R (2014) The Earth Microbiome project: successes and aspirations. BMC Biol 12(1):69
6. Dong Y, Kumar CG, Chia N, Kim P-J, Miller PA, Price ND, Cann IK, Flynn TM, Sanford RA, Krapac IG, Locke RA 2nd, Hong PY, Tamaki H, Liu WT, Mackie RI, Hernandez AG, Wright CL, Mikel MA, Walker JL, Sivaguru M, Fried G, Yannarell AC, Fouke BW (2014) Halomonas sulfidaeris-dominated microbial community inhabits a 1.8 km-deep subsurface Cambrian Sandstone reservoir. Environ Microbiol 16(6):1695–1708. doi:10.1111/1462-2920.12325
7. Korenblum E, Souza DB, Penna M, Seldin L (2012) Molecular analysis of the bacterial communities in crude oil samples from two Brazilian offshore petroleum platforms. Int J Microbiol 2012:156537
8. Kobayashi H, Endo K, Sakata S, Mayumi D, Kawaguchi H, Ikarashi M, Miyagawa Y, Maeda H, Sato K (2012) Phylogenetic diversity of microbial communities associated with the crude-oil, large-insoluble-particle and formation-water components of the reservoir fluid from a non-flooded high-temperature petroleum reservoir. J Biosci Bioeng 113 (2):204–210. doi:10.1016/j.jbiosc.2011.09.015
9. Murali Mohan A, Hartsock A, Bibby KJ, Hammack RW, Vidic RD, Gregory KB (2013) Microbial community changes in hydraulic fracturing fluids and produced water from shale gas extraction. Environ Sci Technol 47 (22):13141–13150. doi:10.1021/es402928b
10. Murali Mohan A, Hartsock A, Hammack RW, Vidic RD, Gregory KB (2013) Microbial communities in flowback water impoundments from hydraulic fracturing for recovery of shale gas. FEMS Microbiol Ecol 86(3):567–580. doi:10.1111/1574-6941.12183
11. Valentine DL, Kessler JD, Redmond MC, Mendes SD, Heintz MB, Farwell C, Hu L, Kinnaman FS, Yvon-Lewis S, Du M, Chan EW, Garcia Tigreros F, Villanueva CJ (2010) Propane respiration jump-starts microbial response to a deep oil spill. Science 330 (6001):208–211. doi:10.1126/science.1196830
12. Wang LY, Ke WJ, Sun XB, Liu JF, Gu JD, Mu BZ (2014) Comparison of bacterial community in aqueous and oil phases of water-flooded petroleum reservoirs using pyrosequencing and clone library approaches. Appl Microbiol Biotechnol 98(9):4209–4221. doi:10.1007/s00253-013-5472-y
13. Hayatdavoudi A, Chegenizadeh N, Chistoserdov A, Boukadi F, Bajpai R (2013) Application of new fingerprinting bacteria DNA in crude oil for reservoir characterization – Part II. In: SPE annual technical conference and exhibition, September 2013. Society of Petroleum Engineers
14. Mason OU, Scott NM, Gonzalez A, Robbins-Pianka A, Bælum J, Kimbrel J, Bouskill NJ, Prestat E, Borglin S, Joyner DC, Fortney JL, Jurelevicius D, Stringfellow WT, Alvarez-Cohen L, Hazen TC, Knight R, Gilbert JA, Jansson JK (2014) Metagenomics reveals sediment microbial community response to Deepwater Horizon oil spill. ISME J 8 (7):1464–1475. doi:10.1038/ismej.2013.254
15. Scott NM, Hess M, Bouskill NJ, Mason OU, Jansson JK, Gilbert JA (2014) The microbial nitrogen cycling potential is impacted by polyaromatic hydrocarbon pollution of marine sediments. Front Microbiol 5:108. doi:10.3389/fmicb.2014.00108
16. Valentine DL, Mezic I, Macesic S, Crnjaric-Zic N, Ivic S, Hogan PJ, Fonoberov VA, Loire S (2012) Dynamic autoinoculation and the microbial ecology of a deep water hydrocarbon irruption. Proc Natl Acad Sci U S A 109

(50):20286–20291. doi:10.1073/pnas.1108820109

17. Bælum J, Borglin S, Chakraborty R, Fortney JL, Lamendella R, Mason OU, Auer M, Jansson JK (2012) Deep-sea bacteria enriched by oil and dispersant from the Deepwater Horizon spill. Environ Microbiol 14(9):2405–2416. doi:10.1111/j.1462-2920.2012.02780.x
18. Redmond MC, Valentine DL (2012) Natural gas and temperature structured a microbial community response to the Deepwater Horizon oil spill. Proc Natl Acad Sci U S A 109 (50):20292–20297. doi:10.1073/pnas.1108756108, PMID: 21969552
19. Kessler JD, Valentine DL, Redmond MC, Du M, Chan EW, Mendes SD, Quiroz EW, Villanueva CJ, Shusta SS, Werra LM, Yvon-Lewis SA, Weber TC (2011) A persistent oxygen anomaly reveals the fate of spilled methane in the deep Gulf of Mexico. Science 331 (6015):312–315. doi:10.1126/science.1199697
20. Goecks J, Nekrutenko A, Taylor J, Galaxy Team (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11(8):R86. doi:10.1186/gb-2010-11-8-r86, Epub 2010 Aug 25
21. Knights D, Costello EK, Knight R (2011) Supervised classification of human microbiota. FEMS Microbiol Rev 35(2):343–359. doi:10.1111/j.1574-6976.2010.00251.x, Epub 2010 Oct 7
22. Caporaso JG, Lauber CL, Walters WA, Berg-Lyons D, Lozupone CA, Turnbaugh PJ, Fierer N, Knight R (2011) Global patterns of 16S rRNA diversity at a depth of millions of sequences per sample. Proc Natl Acad Sci U S A 108(Suppl 1):4516–4522. doi:10.1073/pnas.1000080107, Epub 2010 Jun 3
23. Caporaso JG, Lauber CL, Walters WA, Berg-Lyons D, Huntley J, Fierer N, Owens SM, Betley J, Fraser L, Bauer M, Gormley N, Gilbert JA, Smith G, Knight R (2012) Ultra-high-throughput microbial community analysis on the Illumina HiSeq and MiSeq platforms. ISME J 6(8):1621–1624. doi:10.1038/ismej.2012.8, Epub 2012 Mar 8
24. Werner JJ, Zhou D, Caporaso JG, Knight R, Angenent LT (2012) Comparison of Illumina paired-end and single-direction sequencing for microbial 16S rRNA gene amplicon surveys. ISME J 6(7):1273–1276. doi:10.1038/ismej.2011.186x, Epub 2011 Dec 15
25. Aronesty E (2011) ea-utils: Command-line tools for processing biological sequencing data. http://code.google.com/p/ea-utils
26. Bokulich NA, Subramanian S, Faith JJ, Gevers D, Gordon JI, Knight R, Mills DA, Caporaso JG (2013) Quality-filtering vastly improves diversity estimates from Illumina amplicon sequencing. Nat Methods 10(1):57–59. doi:10.1038/nmeth.2276, Epub 2012 Dec 2
27. Edgar RC (2010) Search and clustering orders of magnitude faster than BLAST. Bioinformatics 26(19):2460–2461. doi:10.1093/bioinformatics/btq461, Epub 2010 Aug 12
28. Fu L, Niu B, Zhu Z, Wu S, Li W (2012) CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics 28(23):3150–3152. doi:10.1093/bioinformatics/bts565, Epub 2012 Oct 11
29. Schloss PD, Westcott SL, Ryabin T, Hall JR, Hartmann M, Hollister EB, Lesniewski RA, Oakley BB, Parks DH, Robinson CJ, Sahl JW, Stres B, Thallinger GG, Van Horn DJ, Weber CF (2009) Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl Environ Microbiol 75 (23):7537–7541. doi:10.1128/AEM.01541-09, Epub 2009 Oct 2
30. Kopylova E, Noé L, Touzet H (2012) SortMeRNA: fast and accurate filtering of ribosomal RNAs in metatranscriptomic data. Bioinformatics 28(24):3211–3217. doi:10.1093/bioinformatics/bts611. Epub 2012 Oct 15
31. Mahé F, Rognes T, Quince C, de Vargas C, Dunthorn M (2014) Swarm: robust and fast clustering method for amplicon-based studies. PeerJ 2:e593. doi:10.7717/peerj.593, eCollection 2014
32. McDonald D, Clemente JC, Kuczynski J, Rideout JR, Stombaugh J, Wendel D, Wilke A, Huse S, Hufnagle J, Meyer F, Knight R, Caporaso JG (2012) The Biological Observation Matrix (BIOM) format or: how I learned to stop worrying and love the ome-ome. Gigascience 1(1):7. doi:10.1186/2047-217X-1-7
33. McDonald D, Price MN, Goodrich J, Nawrocki EP, DeSantis TZ, Probst A, Andersen GL, Knight R, Hugenholtz P (2012) An improved Greengenes taxonomy with explicit ranks for ecological and evolutionary analyses of bacteria and archaea. ISME J 6(3):610–618. doi:10.1038/ismej.2011.139, Epub 2011 Dec 1
34. Hamady M, Lozupone C, Knight R (2010) Fast UniFrac: facilitating high-throughput phylogenetic analyses of microbial communities including analysis of pyrosequencing and PhyloChip data. ISME J 4(1):17–27. doi:10.1038/ismej.2009.97, Epub 2009 Aug 27
35. McMurdie PJ, Holmes S (2014) Waste not, want not: why rarefying microbiome data is

inadmissible. PLoS Comput Biol 10(4): e1003531. doi:10.1371/journal.pcbi.1003531, eCollection 2014

36. Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Appl Environ Microbiol 71 (12):8228–8235
37. Vázquez-Baeza Y, Pirrung M, Gonzalez A, Knight R (2013) EMPeror: a tool for visualizing high-throughput microbial community data. Gigascience 2(1):16. doi:10.1186/2047-217X-2-16
38. Jari Oksanen F, Blanchet G, Kindt R, Legendre P, Minchin PR, O'Hara RB, Simpson GL, Solymos P, Henry M, Stevens H, Wagner H (2014) vegan: community ecology package. R package version 2.2-0. http://CRAN.R-project.org/package = vegan
39. Knights D, Kuczynski J, Charlson ES, Zaneveld J, Mozer MC, Collman RG, Bushman FD, Knight R, Kelley ST (2011) Bayesian community-wide culture-independent microbial source tracking. Nat Methods 8 (9):761–763. doi:10.1038/nmeth.1650

Biodegradation Prediction Tools

Florencio Pazos and Víctor de Lorenzo

Abstract

Experimental approaches for determining the environmental fate of new molecules generated by the modern chemical and pharmaceutical industry cannot cope with the pace at which new of these substances are synthesized, thus raising questions on their ultimate fate if released into the environment. This has fostered the development of different web-based and publicly available platforms that deliver an appraisal of the amenability of given chemical species to microbial biodegradation. One major class of such predictors foretell the final destiny of an input chemical formula, either as an end-point state (i.e., degradable or not) or as an estimation of half-life under certain circumstances. These tools are characteristically automated and thus most suitable for screening large collections of compounds without any expert knowledge. A second type of platforms provides information on the possible – often alternative – biodegradation routes that the compounds under examination may go through, with an indication of possible intermediates and enzymatic reactions. Such pathway-based tools require a degree of interactivity by the user and are more suited to analyses of individual target molecules. The protocols detailed below describe the practical usage of one platform of each type, specifically, EAWAG-PPS (formerly UM-PPS) and *BiodegPred*. Both take as an input the formulas of the compounds, but they deliver different and somewhat complementary information on their most likely environmental fate.

Keywords: Biodegradation, Bioremediation, Metabolism, Prediction, Web server

1 Introduction

The large collection of molecules produced in bulk amounts by the chemical and pharmaceutical industries since the onset of industrialization in the nineteenth century is at the basis of our Western societies and economies. Alas, the same industrial activities typically release into the environment vast quantities of chemical products that can be harmful for the ecosystem. Toxic waste is often generated as a side product of the synthesis or utilization of a molecule of interest, and it becomes noxious once it is released to the environment – either accidentally or deliberately. While large loads of harmful contaminants (e.g., oil, heavy metals) can be partially coped with through mere physicochemical methods (e.g., intensive in-source treatment, physical removal, landfilling), the most typical cases of pollution are those in which the levels of the toxic molecules are low enough to make mechanical removal inefficient while being high enough to cause a distinct environmental impact, often on the long run. Some of the compounds at stake are naturally degraded by environmental physicochemical abiotic processes (photolysis, oxidation, etc.) which transform molecules with a given toxicity into less harmful products. Other molecules can be totally or partially metabolized (or co-metabolized) by environmental microorganisms. But many other substances, termed "recalcitrant," are not "removed" by any of these means, and they remain in the afflicted sites for very long periods of time.

Biodegradation is the ability acquired by certain environmental organisms to catabolize compounds that do not form part of the standard central metabolism. The driving forces for the emergence of such abilities include both the advantage of benefiting from unusual carbon sources and the counteracting of their chemical toxicity [1, 2]. The rational exploitation of biodegradative capacities of naturally occurring or recombinant organisms (generally microorganisms) for removing chemicals from the environment (in particular in cases of low-level but extensive pollution) is generically called "bioremediation" [3, 4]. This approach has advantages and drawbacks. In one hand, since biodegradation routes are evolved or designed to use the target compound as a carbon/energy source, it generally gets completely "mineralized" (i.e., transformed into CO_2, H_2O, and inorganic small ions), which is more desirable than a partial transformation such as that generally associated to abiotic degradation. On the other hand, releasing bacteria (eventually with genetic modifications) that may be able to survive in the environment competing with others raises a large number of issues [5].

Determining the environmental fate of a new chemical before releasing it to the external medium is crucial for designing appropriate strategies for its synthesis, handling, and disposal or even avoiding its usage/release at all. Strict normatives at national and supranational levels control the procedures for determining the environmental fate of substances and the criteria for allowing their usage or not depending on the results of these procedures (e.g., Williams et al. [6]). It is easy to grasp that gathering experimentally enough data on the fate of each of many molecules that are produced every day by synthetic chemists is very consuming in terms of time and resources. Typical tests involve releasing the compound in a controlled environment and measure its concentration in forthcoming samples taken over long periods of time to determine the kinetics of its eventual degradation (e.g., the "half-life" time required to reduce the concentration to a half of the original one). An additional problem is that measuring the disappearance of the original product is not enough, since intermediates of the degradative pathway (not targeted by the measurement) can be hazardous too. Meanwhile, new chemicals are designed at a pace that cannot be coped by these *wet* procedures. For these reasons, the in silico prediction of the biodegradative feasibility of a given chemical compound in the environment is of crucial importance since it could help restricting the experimental time/resources devoted to the task [7, 8]. Indeed, the “benign by design” concept, i.e., take into account the (predicted) proneness to degradation as a positive aspect when designing a molecule, is getting more popular in the chemical field [9].

From a methodological point of view, predicting the biodegradative potential of a compound from its chemical structure is, in essence, similar to predicting any other property, such as its melting point, water solubility, or organismal toxicity. The prediction of toxicity is a much more studied issue due to the more direct relationship with human health and the higher difficulty in performing the experiments: i.e., predicting the toxicity of new drugs in humans. On the contrary, predicting biodegradation is a less-explored subject. In principle this is a more difficult task since it depends on many factors apart from the chemical structure of the compound, such as the physicochemical and biological characteristics of a particular environment: water/soil, pH, microbial communities present, etc. Most attempts for predicting toxicity are based, in one way or another, on “quantitative structure-activity relationships” (QSAR) approaches. Some biodegradability predictors also use these general concepts, while also ad hoc strategies were specifically designed to this particular problem.

Taking into account the output they produce, existing biodegradation predictors can be classified in two main classes. In one hand, there are platforms which only predict the final fate of a given compound, either in a quantitative way (to which extent the molecule under examination is going to be degraded, “half-life”, etc.) or in a qualitative manner: whether the chemical species at issue is going to be degraded or not (according to some criteria). The second type of predictors includes methods which, apart from predicting the final fate, provide some information on the biodegradative pathway it goes through and the intermediate/final products of the process. Both approaches have pros and cons. While the second class of methods provides more information on the degradation process, they also require in many cases some interactivity or additional input from the user. On the contrary, the methods within the first group are more automatic and hence amenable for application to large collections of compounds without user intervention or expert knowledge.

The freely available alternatives within the first group include, for example, the BIOWIN system which, together with other predictors of different properties of molecular structures, is incorporated in the *EPI Suite* distributed by the US Environmental Protection Agency (EPA) (*see* **Note 1**). BIOWIN is based on regression models where compounds are described by vectors coding mainly for the occurrence of substructures in the molecule. Several models for predicting biodegradation are contained in BIOWIN, which differ in the criteria used for defining biodegradability (based on different normatives/databases), the scenarios for biodegradation they were designed for (e.g., hydrocarbon degradation, methanogenic anaerobic degradation, etc.), and the output they produce (qualitative or quantitative). Another example of this class of platforms is the BDPServer [10] which uses a machine learning system ("decision trees") fed with a description of the molecules as vectors coding for the frequency of atom triplets plus molecular weight and water solubility if available. For training the system, the environmental fate of the compounds present in the UM-BBD [11] (*see* **Notes 2** and **4**) was in silico inferred based on whether a pathway connecting them with the central metabolism can be found with the information available at that database. This system has been updated to include other molecular descriptors, other machine learning systems and, more importantly, training sets based on "real" experimental biodegradation data (see below). Indeed, this new version (BiodegPred) is conceived as a "multipredictor": the user can run his/her compound against three different biodegradability predictors (based on three different criteria/databases) as well as a toxicity predictor.

Within the second group, the Pathway Prediction System of the UM-BBD (UM-PPS) allows to interactively infer not only the final environmental fate of a compound but the possible route(s) for its degradation and the intermediates involved as well [12]. The system is based on a set of chemical transformations of functional groups frequently observed in biodegradation processes (called "rules"). These rules are applied to the functional groups found in the compound entered by the user, leading to a number of possible virtual products. The process is iterated for these products until the resulting compounds can enter into the central metabolism and/or no additional rules apply for them. The process is also interactive since the user can choose, from the eventual many alternative routes, which ones to go on exploring, defining in this way the complete biodegradation pathway for the initial compound. This system has been recently improved with a machine learning approach that, trained with known examples of biodegradation, allows assigning probabilities to the different pathways [13]. A similar concept is used in the PathPred system [14] of the KEGG metabolic resource [15]. It uses a set of transformations between molecular substructures (called "rpairs") which are less specific than the transformations of functional groups used in UM-PPS, since they involve smaller molecular fragments. Consequently, the main difference is that PathPred generates many more possible compound conversions. Indeed, this approach is generic for "predicting" metabolic transformations and its application to biodegradation involves mainly using it with the "rpair" transformations frequently observed in KEGG's "xenobiotic biodegradation" pathways. Another approach is followed by the CATABOL/CATALOGIC software [16, 17]. In these systems, the biodegradation pathways for an input compound are delineated based on a set of catabolic transformations (extracted from the literature and UM-BBD) "weighted" with experimental data on biodegradative fates extracted from databases (*see* **Note 2**) and with other factors such as the "biological oxygen demand."

There are many other alternatives, including commercial software. For a recent more exhaustive review, *see* [8]. Here we describe in detail the protocols for using two simple predictors of biodegradability which can be freely accessed through web interfaces. These two methods, described in detail above, represent two very different approaches for biodegradation prediction: an interactive, user-aided approach which gives information not only on the biodegradative fate of a compound but details on the biodegradative pathway(s) as well (UM-PPS), vs. a machine learning system (BDPServer/BiodegPred) which only predicts the final fate but can be applied to large collections of compounds since it is fully automatic.

2 Materials

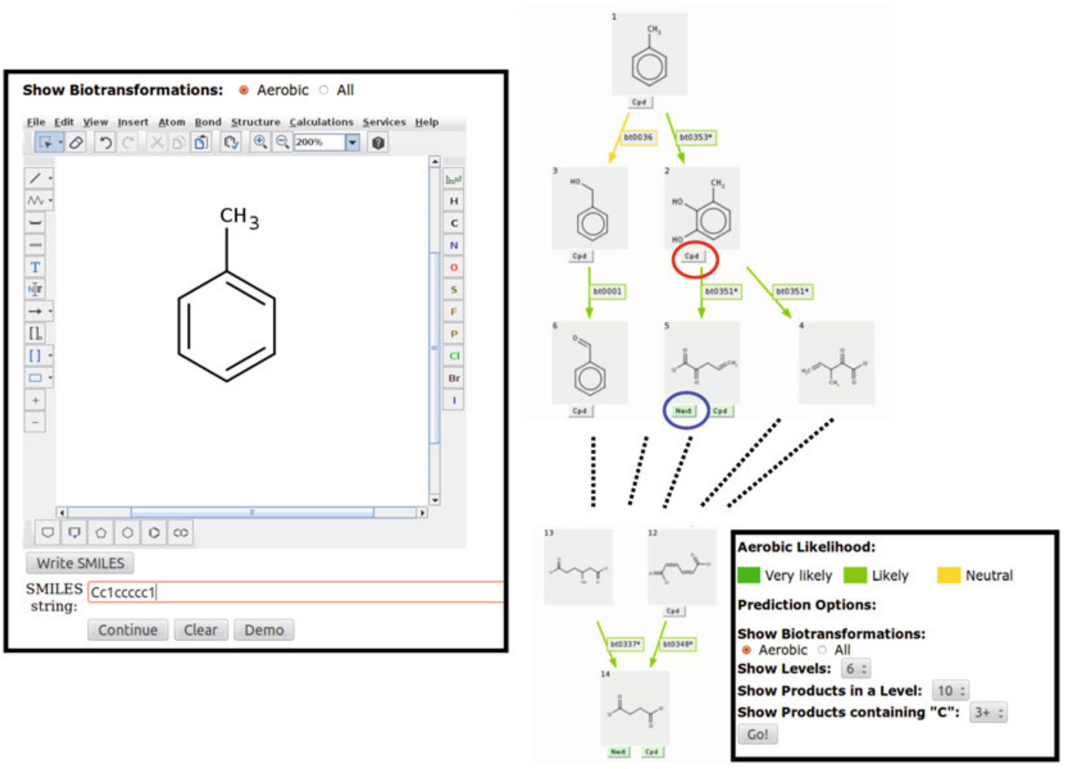
The two resources described in the following protocols can be accessed through a standard web browser. Some of their features are implemented as *Java applets*. You might need to modify your browser's configuration and/or install some additional software for running these applets (*see* **Note 3**).

3 Methods

The two systems described here for the prediction of the biodegradative fate of chemical compounds are very simple to use. In the simplest case, entering the molecular structure of your target compound as single input and pressing a button is enough for obtaining the final result.

3.1 UM-PPS

1. The EAWAG-PPS (formerly UM-PPS) can be accessed at the following URL http://eawag-bbd.ethz.ch/predict/.
2. The only mandatory input for the system is the molecular structure of the chemical compound for which you want to predict the biodegradative fate (see below). Optionally, you can also specify the aerobic character of the environment in which the putative biodegradative process is going to take place (aerobic or anaerobic). If you know the SMILES string representation of your compound (*see* **Note 4**), enter it in the corresponding checkbox. If not, you can "draw" the molecular


Show Biotransformations: [✓ Aerobic] All

SMILES string: Cc1ccccc1

Aerobic Likelihood: 🟩 Very likely 🟩 Likely 🟨 Neutral

Prediction Options:

Show Biotransformations: [✓ Aerobic] All

Show Levels: 6

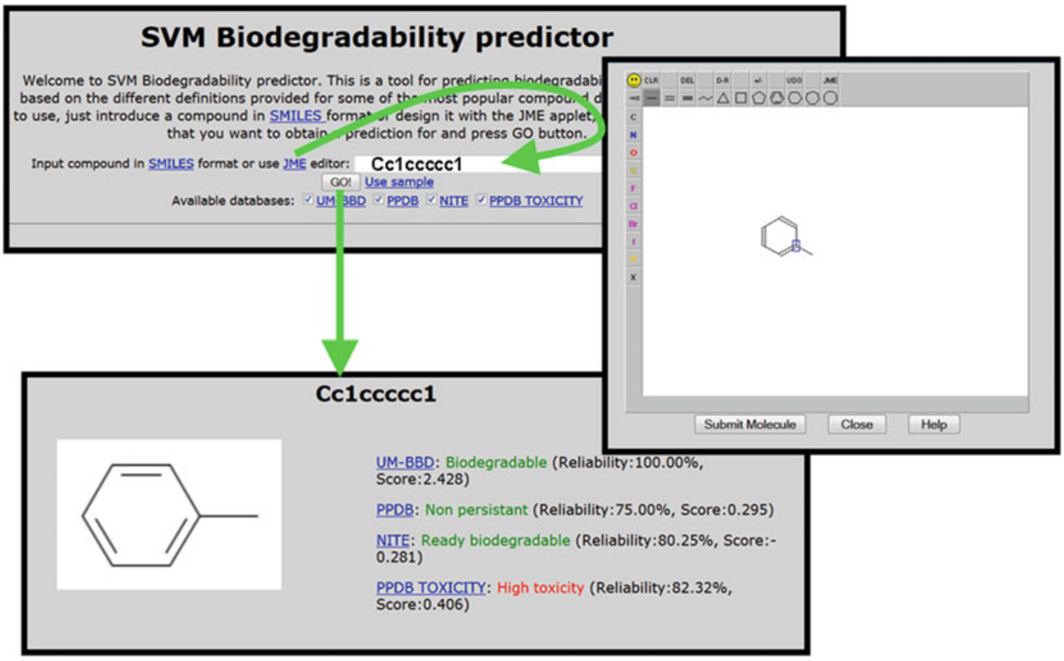
Show Products in a Level: 10

Show Products containing "C": 3+

Fig. 1 Screenshots of the UM-PPS system when predicting the biodegradation routes for toluene (methylbenzene). The input form (*left*) includes a molecular editor to "draw" the structure of the input compound. In the predicted biodegradative routes (a portion of which is shown on the *right*), compounds present in the UM-BBD have a "Cpd" button (*red*) to go to the corresponding pages, while non-end compounds have a "Next" button (blue) to retrieve the downstream biodegradative routes starting with them

structure in the provided molecular editor and press "Write SMILES" afterward to automatically generate the SMILES string for the structure in the editor (Fig. 1).

3. Once the SMILES string of your compound is in place, press "Continue."
4. After a while, a representation of part of the predicted biodegradative pathway for your compound shows up (Fig. 1). In this representation, the aerobic likelihood of the different transformation steps is indicated by a color scale. The "rule" (transformation of functional groups) associated to each putative reaction is also shown (as its UM-BBD code, e.g., "bt0001"). These codes are active links to the UM-BBD pages with detailed information on the rules. Some of the compounds within this pathway (putative intermediate steps of the biodegradative process) might be present in the UM-BBD. In these cases, a "Cpd" label is included, which is an active link to the corresponding UM-BBD pages with detailed compound information.



SVM Biodegradability predictor

Welcome to SVM Biodegradability predictor. This is a tool for predicting biodegradabi
based on the different definitions provided for some of the most popular compou d
to use, just introduce a compound in SMILES format or design it with the JME applet,
that you want to obtain prediction for and press GO button.

Input compound in SMILES format or use JME editor: Cc1ccccc1

GO! Use sample

Available databases: ✓ UM-BBD ✓ PPDB ✓ NITE ✓ PPDB TOXICITY

Cc1ccccc1

UM-BBD: Biodegradable (Reliability:100.00%, Score:2.428)

PPDB: Non persistant (Reliability:75.00%, Score:0.295)

NITE: Ready biodegradable (Reliability:80.25%, Score:-0.281)

PPDB TOXICITY: High toxicity (Reliability:82.32%, Score:0.406)

Fig. 2 Screenshots of the BiodegPred system when used for predicting the environmental fate of toluene (methylbenzene). The input form is at the *top* (including the molecular editor to enter the compound structure), and the results page is at the *bottom*

5. This initial representation does not contain all possible biodegradation routes generated by iteratively applying the rules. Only the first "n" biodegradative steps ("levels") and a given number of compounds per level are shown (see below). The "Next" labels below some of the compounds allow expanding the biodegradative routes starting at these compounds, allowing in this way to interactively explore the whole biodegradation network of your input compound. The idea is to select one route or another based on expert knowledge.
6. Finally, at the bottom of the page, a web form allows you to rerun the system for the same compound but changing the aerobic character or the number of levels and compounds per level shown.

3.2 BiodegPred

1. This system can be accessed at the following URL: http://csbg.cnb.csic.es/BiodegPred/.
2. As in the case of the UM-BBD, the only mandatory input for the system is the molecular structure of the compound. It can also be entered as a SMILES string (*see* **Note 4**) or drawn in the molecular editor following the "SME" link (Fig. 2). There is also a link ("Use sample") for filling the input textbox with an example structure.

3. As commented in the Introduction, this server predicts biodegradability (according with three different criteria) and toxicity. You can choose which of these four predictors you want to use with the provided checkboxes (all are selected by default).
4. To run the selected predictors on your input structure, press "Go."
5. The results page contains a representation of the chemical structure regenerated from the input SMILES (to check that it is correct) and the results of the predictors selected (Fig. 2). As explained earlier, this system only predicts the final "fate" of the compound and does not give information on the pathways used for reaching this final state. For each predictor, the results include the name of the database whose annotated compounds were used for training (UM-BBD, PPDB, NITE, and PPDB toxicity) which are active links to the corresponding resources. Next, you have the prediction for each database: "biodegradable" vs. "non-biodegradable" for UM-BBD (*see* **Note 5**), "persistent" vs. "non-persistent" for PPDB, "ready-biodegradable" vs. "non-ready biodegradable" for NITE, and "low toxicity" vs. "high toxicity" for PPDB toxicity. A color code is used for emphasizing the character of the predictions (green, biodegradable/nontoxic; red, recalcitrant/toxic). The predictor's score and the associated reliability are also indicated. The reliability values associated to each score were obtained from a test set of compounds of known fate, and they represent the fraction of compounds in the test set with that score or higher correctly predicted. Moving the mouse over these data, more information on the criteria used for defining these fates, on the scores, etc., is shown.
6. Note that the criteria used for classifying a compound as "biodegradable" or not in these three resources are different. For example, "persistent" (PPDB) is not exactly the same as "non-ready biodegradable" (NITE). Consequently the same compound could be annotated in different resources with apparently opposite fates, which translates also to the predictions. The user has to interpret these eventual apparent contradictions in view of the exact definition of the criteria.

4 Notes

1. The EPI Suite is available at http://www.epa.gov/opptintr/exposure/pubs/episuite.htm
2. There are many databases with different types of data related to microbial biodegradation of chemical compounds. These are not only useful for the developers of predictors (i.e., to retrieve

datasets for training/testing their systems) but also for the final user. That is because the biodegradation information of our compound of interest (or a similar one) might be already available in these resources. The University of Minnesota Biocatalysis/Biodegradation Database (UM-BBD), now EAWAG Biocatalysis/Biodegradation Database (EAWAG-BD) [11], is the main resource with information on known biodegradation routes (including data on compounds, reactions enzymes, microorganisms, etc.). The main database with general metabolic information, KEGG [15], "mirrors" the UM-BBD data on its "biodegradation of xenobiotics" pathways, so that this information can be queried and used in the same framework as the other KEGG pathways. There are also databases with experimental results on compound biodegradability, such as "half-lives" under different conditions, bioaccumulation, environmental toxicity, etc. For example, the Chemical Risk Information Platform (CHRIP) at the Japanese National Institute of Technology and Evaluation (NITE) (http://www.safe.nite.go.jp/english/) and the UK's Pesticide Properties Database (PPDB) (http://sitem.herts.ac.uk/aeru/projects/ppdb)

3. The molecular editors of the two resources commented are implemented as Java applets embedded in web pages. In recent versions of Java, in order for the embedded applets to work, you have to set the security level to "middle" in the Java configuration panel of your operative system. For example, in MS Windows: control panel > Java > security > security level > middle. Additionally, the first time the applet is run, you will have to accept a number of security warnings and "Allow...?" questions. You also need the *Java Runtime Environment* (JRE) installed on your system for applets to work (e.g., in MS Windows check whether a "Java" item is present in the control panel). You can download and install JRE from https://www.java.com/es/download/

4. The SMILE format (http://www.daylight.com) allows representing any chemical structure as a string of ASCII characters so that it can be stored and handled by computers. Most databases focused on chemical structures include the SMILE representation as a field. Consequently, if your compound is already stored in some database, you can copy/paste the SMILE string from there. If not, most software for converting among chemical formats allows converting from/to SMILES. Finally, there are a number of chemical editors online, such as those used in the two resources described here, which can generate SMILES for the structures entered by the user.

5. The "biodegradable"/"non-biodegradable" definitions for UM-BBD are based, as in the case of the BDPServer [10], on the possibility of finding a biodegradative pathway for the

compounds in the training set in this resource and are not internal annotations of the UM-BBD. Consequently, these annotations are themselves predictions and not experimental outcomes (as in the other three resources).

References

1. Diaz E (2004) Bacterial degradation of aromatic pollutants: a paradigm of metabolic versatility. Int Microbiol 7:173–180
2. Schmid A, Dordick JS, Hauer B, Kiener A, Wubbolts M, Witholt B (2001) Industrial biocatalysis today and tomorrow. Nature 409:258–268
3. Singh A, Ward OP (2004) Biodegradation and bioremediation. Springer, Berlin
4. Dua M, Singh A, Sethunathan N, Johri AK (2002) Biotechnology and bioremediation: successes and limitations. Appl Microbiol Biotechnol 59:143–152
5. Cases I, De Lorenzo V (2005) Genetically modified organisms for the environment: stories of success and failure and what we have learned from them. Int Microbiol 8(3):213–222
6. Williams ES, Panko J, Paustenbach DJ (2009) The European Union's REACH regulation: a review of its history and requirements. Crit Rev Toxicol 39(7):553–575
7. Wackett LP (2004) Prediction of microbial biodegradation. Environ Microbiol 6:313
8. Rücker C, Kümmerer K (2012) Modeling and predicting aquatic aerobic biodegradation - a review from a user's perspective. Green Chem 14:875–887
9. Kümmerer K (2007) Sustainable from the very beginning: rational design of molecules by life cycle engineering as an important approach for green pharmacy and green chemistry. Green Chem 9(8):899–907
10. Gomez MJ, Pazos F, Guijarro FJ, de Lorenzo V, Valencia A (2007) The environmental fate of organic pollutants through the global microbial metabolism. Mol Syst Biol 3:114
11. Gao J, Ellis LBM, Wackett LP (2010) The University of Minnesota biocatalysis/biodegradation database: improving public access. Nucleic Acids Res 38(D):D488–D491
12. Hou BK, Ellis LB, Wackett LP (2004) Encoding microbial metabolic logic: predicting biodegradation. J Ind Microbiol Biotechnol 31 (6):261–272
13. Wicker J, Fenner K, Ellis L, Wackett L, Kramer S (2010) Predicting biodegradation products and pathways: a hybrid knowledge- and machine learning-based approach. Bioinformatics 26(6):814–821
14. Oh M, Yamada T, Hattori M, Goto S, Kanehisa M (2007) Systematic analysis of enzyme-catalyzed reaction patterns and prediction of microbial biodegradation pathways. J Chem Inf Model 47(4):1702–1712
15. Kanehisa M, Goto S, Kawashima S, Okuno Y, Hattori M (2004) The KEGG resource for deciphering the genome. Nucleic Acids Res 32(Database issue):D277–D280
16. Dimitrov S, Kamenska V, Walker JD, Windle W, Purdy R, Lewis M, Mekenyan O (2004) Predicting the biodegradation products of perfluorinated chemicals using CATABOL. SAR QSAR Environ Res 15(1):69–82
17. Dimitrov S, Nedelcheva D, Dimitrova N, Mekenyan O (2010) Development of a biodegradation model for the prediction of metabolites in soil. Sci Total Environ 408 (18):3811–3816

Predicting Protein Interactions

Florencio Pazos and David de Juan

Abstract

Since protein interactions are at the basis of all cellular processes, knowing the interaction network around a protein of interest provides a lot of information on its functioning. It also allows inferring the function of hypothetical proteins, based on those of their interactors. On a larger scale, knowing the whole interactome of a given organism allows studying its biology from a systemic perspective, a tactic which is increasingly more used, for example, to approach diseases. Together with the time-consuming, expensive, and error-prone experimental methods for determining protein interactions, there are a number of computational approaches that are now often used as a complement for the first. They can, for example, target proteins difficult for the experimental techniques or simply provide additional evidences of interaction. These methodologies are mature enough, both in terms of accuracy and easiness of usage to be incorporated into the standard toolboxes of molecular biologists.

The protocols below describe in detail the practical usage of two web-based tools for predicting protein interactions from raw sequence information. These tools provide predictions based on different approaches, whose eventual agreement provides additional support for the predictions.

Keywords: Coevolution, Genomic context, Interactome, Protein function, Protein interaction, Web-based tool

1 Introduction

Interactions between proteins are at the basis of all cellular processes. The molecular functions of individual proteins are like "words" which only acquire a complete meaning (biological function) when linked together in "sentences" (biological processes) via protein-protein interactions. That importance led to the development of a battery of techniques for experimentally determining pairs of interacting proteins [1]. Some of these can be applied in a high-throughput way, in an attempt to uncover whole interactomes, instead of targeting particular interacting pairs. An additional utility of these large interactomes is that they allowed the first studies of molecular systems from a systemic perspective [2, 3], since these systemic approaches take networks (interaction networks in this case) as input. These interactomes, together with other molecular networks are also being used for approaching the diagnostic and treatment of diseases from this systemic point of view [4, 5].

In parallel to the development and application of those experimental techniques, a number of computational methods for predicting pairs of interacting proteins were also devised [6–8]. The most obvious advantage of these in silico methods is that they are faster and cheaper than their experimental partners. But they also have other advantages such as not being influenced by some properties of the target proteins (e.g., concentration, subcellular compartment) known to affect the results of the experimental methods [9]. The obvious drawback of the computational methods, common to most in silico approaches, is their reliability, in principle lower than their experimental counterparts. Nevertheless, current experimental approaches for interaction detection, especially when applied in a high-throughput way, render a considerable number of false positives and negatives [9], making the "reliability gap" between experimental and in silico methods much narrower than in other areas. Altogether, the current view is that the in silico methods complement the experimental approaches, e.g., by targeting proteins problematic for them, restricting the number of pairs to test experimentally, or simply serving as additional evidences of interaction [10]. Indeed, it is becoming standard to incorporate some type of prediction in the recent high-throughput determinations of protein interactions (e.g., [11, 12]).

Computational approaches for predicting protein interaction partners are based on simple sequence and genomic features intuitively related to interaction (Fig. 1). See [6–8] for detailed reviews. For example, two proteins of an organism whose orthologs in another one are "fused" in a single polypeptide are likely to interact [13]. Similarly, two proteins whose coding genes are close to each other in a number of bacterial genomes (an evidence of transcriptional coregulation) are also probably interacting or functionally related in some way [14]. Other approaches look for signatures of coevolution between proteins, under the idea that proteins whose evolutionary histories are entangled are probably interacting and/or working together [15, 16]. An extreme case of evolutionary codependence is represented by proteins that tend to be either present or absent together in the known genomes: if one is not coded in the genome, the other is absent too, so that one never appears without the other. In fact, a number of approaches look for pairs of proteins with similar patterns of presence/absence (a.k.a. "phylogenetic profiles") in a set of genomes as candidate interaction partners [17, 18]. Also of coevolutionary base is the *mirrortree* family of methodologies, which evaluates the similarity of the phylogenetic trees of two proteins (families) as a way of quantifying coevolution [19, 20]. Reading these descriptions, it is easy to see that an important characteristic of these computational methods (also termed "genomic

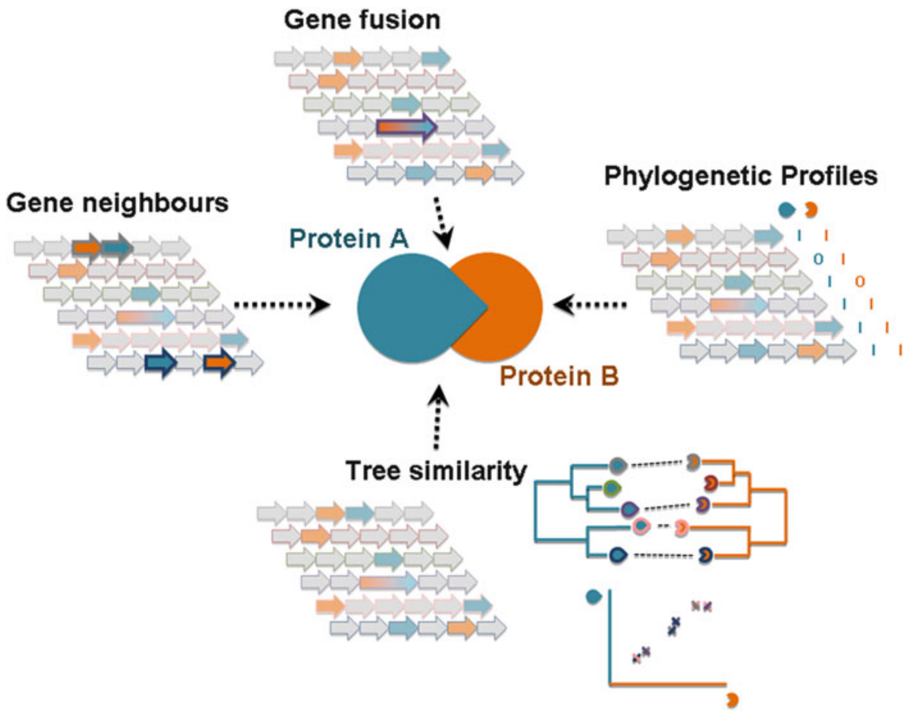


Fig. 1 Context-based methods for inferring protein interactions and functional linkages. The genomic contexts of two proteins (*blue* and *orange*) are represented, highlighting the genomic and sequence features used to infer interaction: a gene fusion event (*top*), conservation of genomic closeness (*left*), similarity of presence/absence patterns (phylogenetic profiles, *right*), and similarity of phylogenetic trees (*bottom*)

context-based methods") is that in most cases they report both physical interactions and other types of functional linkages not necesarily involving direct interactions (e.g., copresence in macromolecular complexes or biological pathways), since the evolutionary landmarks left by all these associations are similar.

In this chapter, we describe two on-line resources for predicting protein interaction partners starting, in the simplest case, with the raw sequences of our protein(s) of interest as input. The first is the STRING server at the EMBL [21]. STRING integrates, in a common framework, experimentally determined as well as predicted protein interactions and functional associations inferred from diverse sources. These different sources of information related to interaction are termed "evidences." The experimental evidences are mainly extracted from different databases compiling the results of high- and low-throughput detections of interactions. The prediction evidences include sequence and genomic-based features such as those commented above. The similarity of the expression profiles of two genes in a set of experiments ("coexpression") or the co-mentioning of the two genes/proteins in the scientific literature ("text mining") are also used to infer protein linkages (*see* **Note 1**).

Another evidence of interaction between two proteins is the fact that their orthologs in another organism are reported to interact ("homology"). STRING assigns, to every pair of proteins, a single numerical score designed to summarize all these evidences and their relative strengths. STRING provides a simple-to-use web-based graphical interface to navigate all that information, and it is the easiest and fastest way of obtaining a summary of the available interaction information for your protein(s) of interest.

The other web resource we are going to describe is the *mirrortree* web server for the interactive study of protein coevolution as predictor of protein interactions [22]. This server implements the basic *mirrortree* methodology for comparing the phylogenetic trees of two protein families [19]. The user has to provide, as only input, the sequences of a representative member of each family. The server generates phylogenetic trees for them and displays an interactive graphical interface where the user can explore their similarity in a taxonomic context. Different global and local (clade-specific) quantifications of tree similarity are provided, so that the user can infer to which extent the two families are coevolving and in which taxa.

2 Materials

The two resources described in the following protocols can be accessed through a standard web browser.

3 Methods

3.1 STRING Server

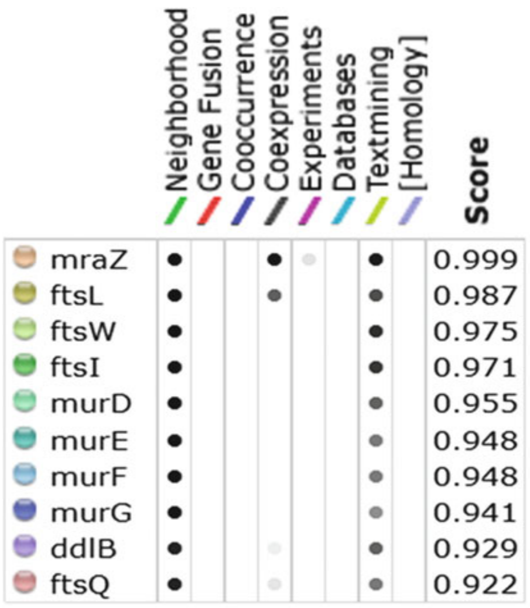
The following protocol describes the basic usage of the STRING web interface (http://string-db.org) for retrieving the interaction context of one or more proteins of interest.

3.2 Input

In the textbox within STRING's main page, enter the identifier (ID) of your protein of interest or its name. IDs of different databases are accepted (e.g., Uniprot, ENSEMBL NCBI, etc.). If you do not find your protein in this way (or it is not in a database) and you want to look for a homolog, you can query STRING by sequence similarity (*see* **Note 2**).

If the input results in several matches (proteins) in STRING's database (e.g., synonymous names/IDs, multiple matches in a sequence search, etc.), a page shows up with a list of the choices for you to select.

It is also possible to use a list of proteins as input for STRING, instead of a single one (*see* **Note 3**).



	Neighborhood	Gene Fusion	Cooccurrence	Coexpression	Experiments	Databases	Textmining	[Homology]	Score
mraZ	●			●	●		●		0.999
ftsL	●			●			●		0.987
ftsW	●						●		0.975
ftsI	●						●		0.971
murD	●						●		0.955
murE	●						●		0.948
murF	●						●		0.948
murG	●						●		0.941
ddlB	●			●			●		0.929
ftsQ	●			●			●		0.922

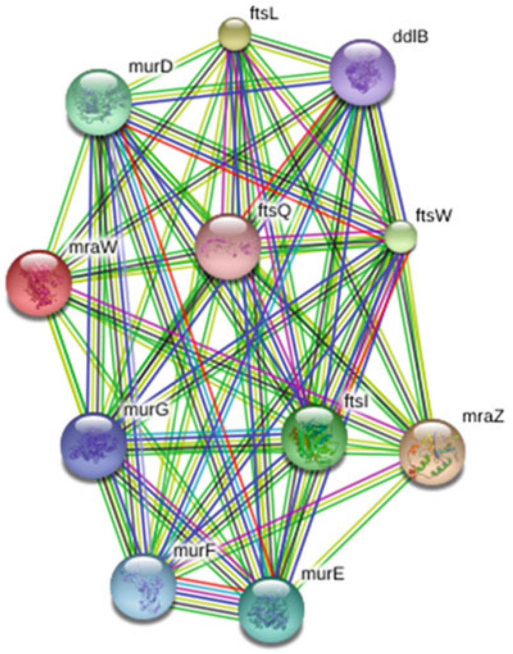


Fig. 2 STRING's network of interactions and functional associations around the *E. coli* protein MraW. The table contains the list of interators, a summary of the evidences supporting each interaction, and the final confidence score

3.3 Main Results Page

At the top of the results page for a given protein, you can find a graphical representation of the interaction network around it. In this network, the nodes are the proteins predicted to interact with yours, and the connections represent the different evidences supporting these interactions, each associated to a different color (Fig. 2). If the three-dimensional (3D) structure of the protein is available or can be modeled by homology, a small figure showing that structure appears within the corresponding node.

The layout of the network can be changed by dragging the nodes with the mouse. On clicking a node, additional information on the corresponding protein shows up. That information includes follow-up links to the corresponding entries for that protein in different databases.

This network not only includes the inferred interactions for your initial input protein but also those among its interactors. The interactions shown are those with a STRING score above a given threshold. You can change that threshold with the "+" and "−" buttons so that the network can be enlarged/reduced by adding/removing interactions. In this way, we can zoom in/out in the interaction context around our protein of interest. With the "confidence" button, you switch to a representation in which the nodes are connected by lines of width proportional to this STRING's score. The "evidence" button changes to the initial representation where the nodes are connected by multiple colored lines representing the different evidences. Obviously, the most reliable interactions are those associated to a large number of evidences and high confidence scores.

Evidences of interaction extracted from the scientific literature ("text mining") might contain additional information on the nature, directionality and characteristics of the interaction, extracted from the textual context where the two proteins are co-mentioned (e.g., "A phosphorylates B," "A is activated by B," etc.). The "actions" button changes the network representation to display that information: nodes become connected by lines colored by type of interaction/action ("bind," "activate," etc.) and with arrows/dots to represent the eventual directionality.

Finally, the "save" button allows exporting this network of interactions to different graphical and text-based formats.

Below this network representation, you find the "Predicted functional partners" table, which contains a detailed list of the predicted interactors shown in the network, indicating the different evidences supporting each interaction and the final score (Fig. 2).

3.4 *Evidence-Specific Pages*

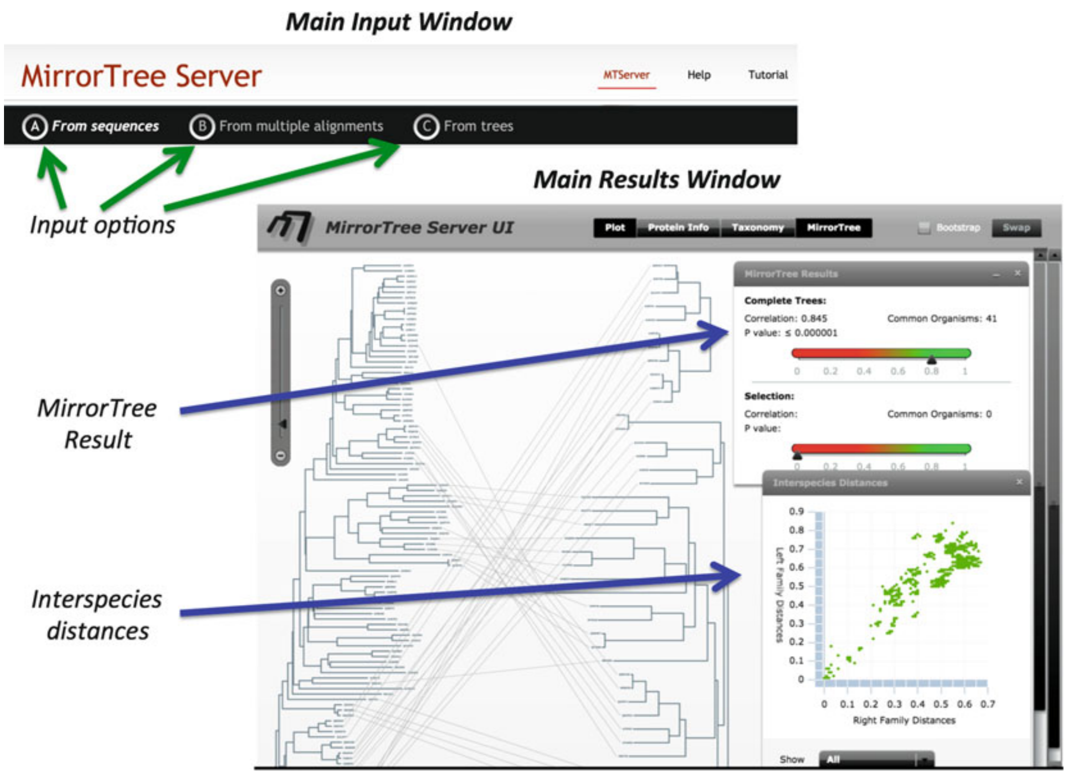
The "View" row contains one button for each type of evidence. These buttons lead to specific pages with detailed information on that particular evidence for the interactions shown in the network. Following them, you can obtain additional (and specific) information regarding a particular type of relationship.

For example, for the "text-mining" evidence, you obtain a list of the publication abstracts co-mentioning the two proteins. The names/IDs of the proteins are highlighted so that you can easily see the textual context where they are co-mentioned and obtain additional information on the type of relationship described. For the "coexpression" evidence, you are taken to a page with a square matrix where the similarity of expression profiles for the proteins in the initial network are indicated with a color scale. For the genomic evidences of interaction, different representations of the genomic contexts of the proteins (Fig. 1) are provided so as to highlight the genomic features from which the interactions were inferred (gene fusion events, conserved operons, similar patterns of genome presence/absence, etc.).

Within these evidence-specific pages, the "summary network" button takes you back to the original network representation of the interactome around your protein.

3.5 Mirrortree *Server*

The following protocol describes the basic usage of the *mirortree* web server for the coevolution-based study of protein interactions, available at http://csbg.cnb.csic.es/mtserver.



Complete Trees:
Correlation: 0.845 Common Organisms: 41
P value: ≤ 0.000001

Selection:
Correlation: Common Organisms: 0
P value:

Fig. 3 Main input and results page of the *mirrortree* web server. The results page shows the mirrored trees on the background and contains a number of floating panels providing different functionalities, such as the taxonomy browser, the interactive of the distance plot, etc.

3.6 Input

The *mirrortree* server accepts three different inputs corresponding to different steps in the process of generating the protein trees whose similarity is going to be evaluated ("A," "B," and "C" in the server's main page – Fig. 3). The "A" input option requires two protein sequences to quantify the coevolution between the corresponding families. This is the default option as it is thought for nonexpert users. These sequences will be the starting point of the process of detecting orthologs in available complete proteomes, building multiple sequence alignments, and calculating the protein trees for each of them. Some advanced options are also available for tuning different parameters related to the homology search and orthologs selection (*see* **Note 4**). While this protocol provides an easy and consistent way of obtaining the protein trees required for a general *mirrotree* analysis, specific research questions might require the intervention of a human expert to generate the adequate trees for them. These specific scenarios might include protein domain analyses (i.e., generate trees for specific domains instead of the whole protein), usage of user curated orthologs, more exhaustive and detailed assessment of the multiple sequence alignments, or exploration of alternative phylogenetic trees. For these and other user-specific needs, the server provides the possibility of introducing either the two multiple sequence alignments (input option "B") or the two final protein trees (option "C"). Options "B" and "C" require certain expertise from the user. In addition to the knowledge required for getting high-quality alignments and trees, some details on the format need to be considered for the server functionalities to work (see the server Help pages for additional info). For the three entry points, input data can be pasted in text boxes or uploaded in separated files for each protein. A brief job description can be also included, and an email address is required in order to receive a notification when the job has finished. That notification includes a link to the results page. This is useful since jobs involving the whole protocol (option "A") and/or families with many orthologs can take a relatively long time to run.

3.7 Results

Mirrortree results are displayed in a multi-panel interface where individual panels can be shown or hidden using the buttons included at the top of the main window (Fig. 3). Above these buttons, links to the intermediate results of the protocol performed by the server ("A" and B" input options) are available (*see* **Note 5**). The main window shows an interactive representation of the two protein trees, with connections between their leaves (proteins) according to the species they belong to (Fig. 3). Selection of a protein in these trees displays its basic information in the "Protein Info panel" (to get this function with "B" and "C" input options, valid protein accession numbers from UniProt must be used in the input files).

The interface allows selecting subtrees in order to restrict the mirrortree calculation to that selection. This can be done manually by selecting tree nodes with the mouse: the whole clade starting at this node becomes selected. As this manual selection can be tedious for large trees, the interface includes a "Taxonomy panel" where proteins from specific taxonomic groups can be selected. In both cases, selection of specific subtrees allows studying the strength of the tree similarity in different groups of species.

The "Results panel" contains the mirrortree results (for the whole trees and the selected clades/taxa). These results include a plot with the protein distances in both trees ("interspecies distances") and their correlation as a quantification of coevolution (main score of mirrortree), together with an associated *p*-value. The distance plot is useful for getting insight into the contribution of different regions of the protein trees to the quantified similarity. For example, the presence of groups of points not following the general correlation trend ("outliers") can indicate taxonomic groups where the two proteins are not coevolving. When a cloud of points (pairs of distances) is selected in this plot, the corresponding species become highlighted in the trees. This, together with the possibility of restricting the calculation to specific tree clades/taxa, allows to interactively delimitate the species where the two proteins are coevolving. When just the two raw protein sequences have been provided ("A" input option), raw and/or aligned orthologous sequences for the whole trees and for selected subtrees can be retrieved, so that users can perform further external refinements in order to carry out better-suited analyses for specific cases.

The system includes a detailed Help page and a guided Tutorial with pre-compiled examples for better understanding the server capabilities.

4 Notes

1. Taking into account the very different nature and origin of STRING's evidences, it becomes clear that the goal of STRING is to store associations between proteins understood in the broadest possible sense: physical interactions, functional relationships, etc. It is important to take this into account when interpreting STRING's results.
2. In this case, STRING will perform a BLAST search of your input sequence against its sequence database and show you the list of hits for you to select.
3. It is also possible to use a list of proteins as input for STRING. This allows inferring whether there are functional associations linking them or how clustered they are according with STRING's evidences. This can be useful, for example, to analyze the functional homogeneity of a set of proteins.
4. These advanced options allow changing, for example, the sequence identity cutoff for considering homology/orthology or the BLAST *e*-value cutoff for the sequence searches. While they should be ok for most situations, they can be changed by advanced users if, for example, distant homologs known to belong to the family are not being picked up.
5. The server generates, as intermediate results, multiple sequence alignments and phylogenetic trees for the two families of interest. These can be useful outside the mirrortree workflow since many bioinformatics approaches with different goals take alignments or trees as input.

References

1. Shoemaker BA, Panchenko AR (2007) Deciphering protein-protein interactions. Part I. Experimental techniques and databases. PLoS Comput Biol 3(3), e42
2. Jeong H, Mason SP, Barabási AL, Oltvai ZN (2001) Lethality and centrality in protein networks. Nature 411:41–42
3. Barabasi AL, Oltvai ZN (2004) Network biology: understanding the cell's functional organization. Nat Rev Genet 5(2):101–113
4. Zanzoni A, Soler-Lopez M, Aloy P (2009) A network medicine approach to human disease. FEBS Lett 583(11):1759–1765
5. Cho DY, Kim YA, Przytycka TM (2012) Network biology approach to complex diseases. PLoS Comp Biol 8(12), e1002820
6. Harrington ED, Jensen LJ, Bork P (2008) Predicting biological networks from genomic data. FEBS Lett 582(8):1251–1258
7. Shoemaker BA, Panchenko AR (2007) Deciphering protein-protein interactions. Part II. Computational methods to predict protein and domain interaction partners. PLoS Comput Biol 3(4), e43
8. Petrey D, Honig B (2014) Structural Bioinformatics of the Interactome. Annu Rev Biophys 43:193–210
9. von Mering C, Krause R, Snel B, Cornell M, Oliver SG, Fields S, Bork P (2002) Comparative assessment of large scale data sets of protein-protein interactions. Nature 417:399–403
10. Lee I, Date SV, Adai AT, Marcotte EM (2004) A probabilistic functional network of yeast genes. Science 306(5701):1555–1558
11. Havugimana PC, Hart GT, Nepusz T, Yang H, Turinsky AL, Li Z, Wang PI, Boutz DR, Fong V, Phanse S, Babu M, Craig SA, Hu P, Wan C, Vlasblom J, Dar VU, Bezginov A, Clark GW, Wu GC, Wodak SJ, Tillier ER, Paccanaro A, Marcotte EM, Emili A (2012) A census of human soluble protein complexes. Cell 150(5):1068–1081
12. Tabach Y, Golan T, Hernandez-Hernandez A, Messer AR, Fukuda T, Kouznetsova A, Liu JG, Lilienthal I, Levy C, Ruvkun G (2013) Human disease locus discovery and mapping to molecular pathways through phylogenetic profiling. Mol Syst Biol 9:692
13. Enright AJ, Iliopoulos I, Kyrpides NC, Ouzounis CA (1999) Protein interaction maps for complete genomes based on gene fusion events. Nature 402:86–90
14. Dandekar T, Snel B, Huynen M, Bork P (1998) Conservation of gene order: a fingerprint of proteins that physically interact. Trends Biochem Sci 23:324–328
15. Pazos F, Valencia A (2008) Protein co-evolution, co-adaptation and interactions. EMBO J 27(20):2648–2655
16. Juan D, Pazos F, Valencia A (2013) Emerging methods in protein co-evolution. Nat Rev Genet 14(4):249–261
17. Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO (1999) Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. Proc Natl Acad Sci USA 96:4285–4288
18. Date SV, Marcotte EM (2003) Discovery of uncharacterized cellular systems by genome-wide analysis of functional linkages. Nat Biotechnol 21(9):1055–1062
19. Pazos F, Valencia A (2001) Similarity of phylogenetic trees as indicator of protein-protein interaction. Protein Eng 14:609–614
20. Juan D, Pazos F, Valencia A (2008) High-confidence prediction of global interactomes based on genome-wide coevolutionary networks. Proc Natl Acad Sci USA 105 (3):934–939
21. von Mering C, Huynen M, Jaeggi D, Schmidt S, Bork P, Snel B (2003) STRING: a database of predicted functional associations between proteins. Nucleic Acids Res 31:258–261
22. Ochoa D, Pazos F (2010) Studying the co-evolution of protein families with the Mirrortree web server. Bioinformatics 26 (10):1370–1371

Syntax and Semantics of Coding in Python

Jeremy Meulemans, Tonya Ward, and Dan Knights

Abstract

Python is a dynamically typed programming language providing a unique blend of power, simplicity and expressiveness that has quickly established itself as a major player in technical fields. The language is built around a very natural syntax, making it an ideal language for both beginners to programming and scientists looking for rapid research implementation. Usage cases from simple file formatting to determining the impact of oil degradation in obtained metagenomic sequence data are facilitated by features of the language. For example, the study of oil breakdown by bacterial communities is likely to involve handling very high-dimensional data, including both the metabolites present in oil and the hundreds of bacterial species often present in environmental communities. The ease of incorporating existing Python modeling packages and the solid frameworks provided by python for scientific computing make this an ideal language for analyzing these types of data.

Keywords: Coding, Introduction, Language, Problem solving, Python

1 Introduction

Python is a rich language with widespread use in education, industry, and research. Python is largely chosen in these fields for its readable syntax, fast development speed, and expressiveness as a general purpose tool for problem solving. From our experience, programs written in languages such as C++ and Java can be written in fewer lines of code, with a higher degree of readability and abstraction, with Python. In comparison to R, Python tends to emerge as a more general use language suited to a wider variety of applications and larger projects, although R does arguably outperform in some areas such as early integration of advanced statistical tools.

For researchers working with hydrocarbons and lipids, a good example of the actual usage of Python is given by Mason et al. [1]. In their analysis of the Deepwater Horizon oil spill's effect on microbial communities, raw DNA sequencing data from sediment samples were processed by a Python package called QIIME [2] to quality filter the reads. The cleaned data was then loaded into a package called USEARCH [3], to perform clustering, and then aligned using another Python package called PyNAST [4]. The results of this Python pipeline were used to reveal that samples exhibiting high levels of polycyclic aromatic hydrocarbons had an abundance of genes involved with denitrification pathways.

As Python is a general language, it would be impossible to consider all possible use cases; however, a strong foundation of syntax and functionality of programming in Python should serve as a basis for advanced usages like that of the previous example. To get the most out of the following overview, it is best to read with a mixture of caution and curiosity. For example, when reading the following explanation of conditional statements, it is natural to wonder if the code executed by each conditional statement will be the same if its operands are swapped, or if statement ordering matters, or if if statements are allowable among elif statements. While covering all such cases is difficult, what is beautiful about Python is the rapidity and ease with which code can be written and experimentally tested. As we will explain, one can simply create a file with the different features of the language that are in question, with the use cases of interest, and run it. The output of the interpreter (or the error messages generated if running fails) will serve to quickly refine and answer many questions.

In the following sections, the basics of Python will be established, followed by a larger example of problem solving using Python. This is intended as quick reference to the important features of the language from the perspective of a biological researcher with little to no computer science background and is therefore by no means complete. The material has been tailored to the elements that are most used in practice, but the depth of Python is far greater than this guide. However, the guide should serve as an adequate starting point. Fortunately, Python's popularity has insured that the Internet is an incredibly rich resource for further information and troubleshooting with the language beyond what can be accomplished or explained by one's own use and experimentation.

2 Installation

Python is available for all major platforms, and in the case of many Linux distributions and OSX, it comes installed on the system. Installation is usually straightforward, and issues that arise can often be readily troubleshooted through a simple web search. In this guide, the version of Python used is version 2.7.6. The print function changes from print to print() in later versions, which is the only feature that should cause deviations from examples given below if a later version is used.

The command line is where most of the functionality of Python lies, so basic familiarity is a prerequisite. For this guide, this includes primarily navigating into and out of directories from the command line. Many tutorials exist on this topic, and again a simple web search should suffice for any questions.

Once Python is successfully installed, try typing python into the command line. An interpreter should come up with information about the current version of Python installed on the system. Try typing 2+2 into this interpreter, or print "hello world". To exit, type quit(). This demonstrates some of the basic interaction with the Python interpreter. In the following examples, >>> will make explicit the input of commands typed into this interpreter.

3 Types, Structures, and Basic Usage

3.1 Numbers and Importing

Integers and floating point numbers are handled in an intuitive way in Python. The following are all integers:

```
1
13
736205637
```

and the following are all floats:

```
1.0
13.45
736205637.0
```

These values can be stored in variables in a predictable way, with math also behaving as expected:

```
>>> int1 = 3
>>> int2 = 5
>>> float1 = 3.0
>>> float2 = 5.67
>>> floatPi = 3.14159
>>> print int1 + int2 + int2 - int1
10
>>> print float1 * float2
17.01
>>> print int2 ** float1
125.0
>>> diam = int1 + int2
>>> area = floatPi * (diam/2.0)**2.0
>>> print area
50.26544
```

Note that if both operands are integers for division in Python, integer division will be performed, so $5/2 = 2$, $5/2.0 = 2.5$, and $5.0/2.0 = 2.5$.

Although the creation of functions will be explored later in this guide, the usage of functions is relatively straightforward. If the function is built-in to Python or user defined, the function is callable with the function(argument1, argument2, ...) syntax.

If the function originated with some module that was imported, as is seen below, the module.function(arg1, arg2, . . .) syntax is used.

In the case of mathematics in Python, further functionality for math is implemented in the provided math module. The functionality of this module is imported into the interpreter, thereby making the functions of the module callable. Functionality for third party packages, once installed, is imported in much the same way (*see* Sect. 4.2):

```
>>> import math
>>> math.sqrt(9.0)
3.0
>>> math.ceil(5.0/2)
3
>>> math.pow(3, 3) == 3**3
True
>>>help(math)
Help on module math: ...
```

The last statement shows how to view the full functionality of a module in Python. The help(module) syntax provides a succinct but comprehensive listing of all the functions that are provided by a module, provided the module creator has implemented such a help listing.

3.2 Booleans

The previous example also revealed another of the basics types of Python, the Boolean type that can have values True and False. For example:

```
>>> 3 == 3.0
True
>>> 3 != 3.0
False
>>> (3 != 4.0) == (4 <> 5.0) == (8 < 9)
True
>>> True and False
False
>>> not (False or True or False)
False
>>> 9 < 8 and ((9/0) == 0)
False
```

Notice in the last example that the and operator does not get evaluated in its entirety. If the first argument to this operator is false, the second argument is never evaluated because the overall and statement can never be true. Because of this, the invalid operation 9/0 is never evaluated. This same short-circuiting principle holds for or as well, in the case that the first operand is true.

3.3 Conditional Statements

Conditional statements are based on if, elif, and else. For some Boolean statements condition1 and condition2 that have some value of True or False, the basic syntax is:

```
if condition1 :
    #do some action
elif condition2 :
    #do some other action
else :
    #do yet another action
```

Thinking about the operation of such conditional code is relatively straightforward. If condition1 is true, then do some action; otherwise if condition2 is true, do some other action instead. If neither condition1 nor condition2 is true, execute yet another action. elif statements and else statements do have the stipulation that a corresponding if statement exists. Note that an arbitrary number of elif statments can be used, as well as multiple if statements.

3.4 Using .py Files

So far, the code that has been run in the interpreter has been single lines, which are typed and then executed. With the branching into conditional statements, code now expands to cross multiple lines. While such code can be typed directly into the interpreter, doing so is difficult, slow, and tedious to edit. A far better approach is the use of a text editor to save and edit the code that is to be executed by the interpreter. To accomplish this, simply create a file named test.py or guide.py or anything_else.py, make sure the .py filetype is present, and enter code into this file. As an introductory example, consider the following code stored in a test.py file, which is itself present on the desktop. *Note that tabs in Python is important; indentations as shown must be present for correct operation.*

It is also worth noting at this point that there is an established coding style for Python called pep8 detailing coding conventions for everything from indenting to naming variables. The examples in this document make an effort to comply with the pep8 standard. Further information on pep8 can be found online.

Continuing on with using .py files, note that everything nested by a tab under an if statement is the code that will get executed if the conditional for that statement is true. As an example of this write the following lines of code to a file called test.py stored on the desktop:

```
import math as mt
condition1 = 1>3
condition2 = mt.sqrt(9.0)<4
if condition1:
    print "condition1 was true."
elif condition2:
    print "condition2 was true."
else:
    print "both conditions were false."
```

To run this code simply navigate to the Desktop directory from the command line (on a Mac- or Linux-based machine, this would be cd ~/Desktop); on a machine running Windows this, would be c:\Users\(username)\Desktop), and then run the command python test.py. The output should be condition2 was true. The code in the file is executed line by line as if it were typed into the interpreter. In this case, it is determined that condition2 is true, and the following print statement is executed.

For a Unix or Mac user familiar with the so-called "shebang" syntax (the use of #! at the start of a file) another option for running a Python file is to add a "shebang" specifying Python as the interpreter at the very top of the file. For this example, the first line then becomes #!/usr/bin/python, above the import math line. The shebang should specify the absolute path to the Python interpreter. If this is unknown, on a UNIX system this can be found by typing which python into the command line. The next step is to mark the file as executable with the command chmod +x test.py. After these steps are completed, the Python script can be executed by typing ./ test.py.

Using either method of running Python code, it is relatively easy to create use cases for Python to see what works simply by typing it into a .py file and running it from the command line. For example, one can find out what happens when running a file containing:

```
x = 3.0
if None:
    x = False
elif not 3 or 4:
    x = 8
else:
    x = "hello"
print x/2
```

The output is 4. A short explanation is that the None value is relatively special; in this particular context it is not True (although interestingly it is also not False), and therefore the assignment of the value False to x never occurs, since a value of True is a prerequisite for the first conditional. As a holdout from earlier programming languages such as C, the integer value 0 is treated as the Boolean condition False, and any value other than 0 is regarded as True. The values of 3 and 4 are therefore regarded as True in the next conditional, thereby evaluating the or to True. This results in the assignment of the value 8 to x and also means that the else will never be executed since an accompanying if evaluated to True. The print statement is then evaluated, performing the division and then printing the result. Although this is an unrealistic example, it serves to show that experimentation with different use cases in Python is a quick and straightforward process.

The above example also demonstrates that the flexibility of Python is a double-edged sword. It is perfectly allowable to have a variable assigned a float value, and in the next line assign that same variable a string. This can be very useful; it is easy to imagine a program that at times handles one output and at others may have a list of outputs, where the reuse of a variable is natural. However, there are some inherent dangers as is illustrated above. If x had been assigned the string "hello", a runtime error would have occurred when the interpreter tried to divide the string by 2, an undefined operation.

3.5 Loops

Further constructs of Python are while and for loops. while loops consist of a single conditional statement. While this statement is true, the body of the loop will be continually executed. To see an example of this, consider the following loop:

```
i = 0
while i < 10:
    print i
    i += 1
```

The numbers 0–9 should be printed when this file is run, stopping at 9. Once the counter is 10, i is no longer less than 10, and the loop terminates. In general, while loops are useful when the number of iterations of a loop is not known. For example, consider finding the first 10 odd numbers in a randomly generated sequence of numbers. It is not known at runtime how many numbers must be iterated over before 10 odd numbers are found. In a case like this, a simple counter for the number of odd numbers found and a while loop with a conditional on the counter that terminates when 10 have been found is an appropriate and natural solution.

In contrast to a while loop, a for loop is used in cases when the number of times the loop will run is known. As a basic example, the counterpart to the while loop above is:

```
for i in range(10):
    print i
```

range(x) simply generates all numbers from 0 – (x−1). For each of these numbers, the body of the loop is executed once. Although this example is basic, for loops in Python are actually quite powerful, with a syntax demonstrated later that lends a high degree of readability to the code.

3.6 Lists

A large part of the power of looping in Python is found in the ability to iterate over lists. Lists in Python are simply collections of integers, floats, strings, or any more complex object. Lists are created with the [] syntax. Some examples include:

```
[1, 2, 3]
["one", "two", "three"]
[[1, 2, 3], [4, 5, 6]]
["one", 2, "three", 4.0]
```

The second to last example above is a list of lists, a list where each object is itself a list, which can be thought of as a matrix when the entries are numeric as above. The last example demonstrates the fact that a list can contain any number and type of object, in this case mixing strings, integers, and floats.

Accessing the elements of a list is straightforward. Singular elements of a list can be accessed and modified, as well as sublists of a larger list. It is important to remember that in Python and many other languages, *the first element of a list is indexed by 0, the second by 1, and so on*. Individual list elements are accessed with the list[index] syntax, whereas slices of lists, or sublists, are called with the list[lower_bound:upper_bound] syntax. Every element of the list from the lower_bound up to one less than the upper_bound will be obtained. Some examples of accessing, slicing, and modifying lists in the interpreter include:

```
>>> L = ["one", 2, "three", 4.0]
>>> L[0]
"one"
>>> L[3]
4.0
>>> L[0:2]
["one", 2]
>>> L[:2]
["one", 2]
>>> L[1:-1]
[2, "three"]
>>> L[:] = [4, 5, 6]
>>> L
[4, 5, 6]
>>> L[2] = 7
>>> L
[4, 5, 7]
>>> L.append(21)
>>> L
[4, 5, 7, 21]
>>> len(L)
4
```

It is important to remember that slicing is up to one less than the upper bound, just as range(x) generates each number from 0 up to (x-1). The negative index specifies an index from the end of the list, where −1 itself specifies the last element. The append function on a list adds an element to the end of a list. The len() function is a built-in function in Python that returns the number of items in an object. The len() function can also be used for many other objects, including tuples and sets.

3.7 Tuples

Tuples in Python are very similar to lists, except in the fact that elements of a tuple cannot be modified once the tuple has been created, meaning tuples are immutable. This is useful to prevent errant modification in a program, which is an issue that can arise with the usage of lists. Tuples are created with the () syntax, similar to the [ ] for a list:

```
>>> t = ("one", 2, "three", 4.0)
>>> t[0]
"one"
>>> t[3]
4.0
>>> t[0:2]
["one", 2]
>>> t[:2]
["one", 2]
>>> t[1:-1]
[2, "three"]
>>> t[0] = 6
TypeError: 'tuple' object does not support item assignment
```

3.8 Sets

Sets in Python are another object that share many similarities with lists. The principle differences between a set and a list are that a set is unordered, it can only contain one instance of any object contained in the set, and access to an element is comparatively fast. Elements of a set must be added individually or converted from a list or a tuple. Sets are specified with the set() syntax, so, for example:

```
>>> s1 = set()
>>> s1.add(3)
>>> s1.add(2)
>>> s1.add(1)
>>> s1
set([1, 2, 3])
>>>s1[0]
TypeError: 'set' object does not support indexing
>>> s2 = set([3, 4, 5])
>>> s2.union(s1)
set([1, 2, 3, 4 ,5])
>>> s2.intersection(s1)
set([3])
>>> help(set)
```

As in the code above, sets can be thought of in much the same way as the mathematical notion of a set, and many operations exist that reinforce this treatment, as is shown by the help(set) output.

3.9 Loops, Continued

With the above-mentioned data structures, the power of the for loop becomes apparent. Consider the following code:

```
numbers = set((1, 2, 3))
total = 0
for number in numbers:
    total += number
print total
```

This code is readable and structured naturally through whitespace, and its purpose is apparent. When run, the output is the expected value 6. For any iterable object, be it a list, tuple, set, or some other object, the for x in y syntax provides a natural way to iterate over every item in the object.

3.10 List Comprehensions

Now that loops have been properly introduced, a more advanced, succinct, and readable way to create lists known as list comprehensions has become accessible. As a basic example, consider the problem of putting the numbers 0. . .9 into a list. An approach may be something like:

```
L = []
for i in range(10):
    L.append(i)
```

The alternative offered by list comprehensions is the notation:

```
L = [i for i in range(10)]
```

Conditionals are also usable in list comprehensions. Suppose there is a set of numeric tags stored as strings, and the goal is to obtain a list containing only the tags that start with some identifier 134. Using a built-in function of strings, a list comprehension could be used as follows for some tag_set:

```
iden_134 = [tag for tag in tag_set if tag.startswith
('134')]
```

List comprehensions can be arbitrarily nested as is needed. This can become a complicated topic to reason about, but as a basic example, consider creating simple array of numbers, a list of lists. For a 110x100 array, this can be accomplished like:

```
array = [[num for num in range(100)] for num in range
(110)]
```

Naturally, each list comprehension can become more complicated with conditional statements and different objects. In practice, although single list comprehensions typically make for more readable and understandable code, nesting list comprehensions tends to accomplish the opposite.

3.11 Dictionaries

At a high level, dictionaries store key-value pairs, where each key and value is an object. The value can be obtained by accessing the key in the dictionary, much like looking up a word in a real dictionary yields the word's definition. Although they may not look it initially, dictionaries in Python are one of the most powerful and natural tools for solving problems.

Creating dictionaries in Python can be done either through direct creation, or through the use of built-in functions, and can be accessed through the [] syntax. For example:

```
>>> leg_count = {}
>>> leg_count["dog"] = 4
>>> leg_count["centipede"] = 100
>>> leg_count["millipede"] = 200
>>> leg_count["centipede"]
100
>>> leg_count["cat"]
... KeyError: 'cat'
>>> leg_count["cat"] = leg_count.get('cat', 4)
>>> leg_count["cat"]
4
>>> leg_count.keys()
['dog', 'centipede', 'millipede', 'cat']
>>> leg_count.values()
[4, 100, 200, 4]
>>> leg_count.iteritems()
<dictionary-itemiterator object at 0x109c49730>
```

As is shown, lists of both the keys and values can be accessed from the dictionary, as well as key-value pairs in the form of the dictionary.iteritems( ). The dictionary.get(value, default) is a means of accessing the value associated with a key, except that instead of an error if no value exists, a predefined default value can be supplied.

3.12 Functions

Now that the basic types have been explored, it is time to define functions, which have been in use throughout this tutorial. Functions are simply snippets of code to perform some task on a variety of inputs, made to be reusable to avoid code duplication. For example, the previous len() function takes as input some object and returns the number of items in that object. The print function simply displays a string on the console. Note that print is unique for the lack of parenthesis required. Creation of functions is prompted through the def keyword, as demonstrated below in a basic implementation of the built-in len(), and values are returned from functions with the return keyword:

```
def length(some_object):
    length = 0
    for item in some_object:
        length += 1
    return length
```

One may try this using the command-line. After importing the above code in the Python interpreter using the from filename import length syntax, the function length( ) should work for objects such as lists and sets, although it certainly isn't as robust as the built-in len function. For a more interesting example, consider the task of counting the number of occurrences of each individual nucleotide in a DNA string and then printing the results. One way to accomplish this task would be:

```
def count_nucleotides(DNA):
    counts = {}
    for letter in DNA:
    if letter == 'A':
       counts['A'] = counts.get('A', 0) + 1
    elif letter == 'C':
       counts['C'] = counts.get('C', 0) + 1
    elif letter == 'T':
       counts['T'] = counts.get('T', 0) + 1
    elif letter == 'G':
       counts['G'] = counts.get('G', 0) + 1
    for nucleotide, count in counts.iteritems():
        print (nucleotide, count)
```

After adding this to a file, import it into a Python session. Once count_nucleotides has been called on some DNA string, it should successfully print the number of occurrences of each individual nucleotide. This function does not have a return value, which is allowable in Python. Functions can perform tasks without returning anything, such as printing items to the screen.

There is also a shorter way to accomplish the above task. In this particular case, a built-in function conveniently called count for strings (and lists) will count the number of occurrences of an input item in the string or list. Using this, and adding support for RNA, the following would be a more reusable function:

```
def count_nucleotides(string, nucleotides):
    for nucleotide in nucleotides:
        count = string.count(nucleotide)
        print (nucleotide, count)
```

This function now requires two inputs, the first a string with the nucleotides to be counted and the second an iterable object containing the nucleotides of interest, such as a list, tuple, or set. Now, to call, simply import into a Python session, and then call something like:

```
count_nucleotides('AACCTTGG', ['A', 'C', 'T', 'G'])
```

Now, suppose that in 90% of use cases, the string of interest is going to be DNA, and only occasionally will RNA or some other string be input. This is an ideal case for the usage of default arguments, where an input to a function has some default value unless otherwise specified. The nucleotide counting function would now be:

```
    def count_nucleotides(string, nucleotides=['A', 'C',
'T', 'G']):
    for nucleotide in nucleotides:
        count = string.count(nucleotide)
        print '('+nucleotide+', '+str(count)+')'
```

The usage would now be:

```
>>> count_nucleotides('AACCUUGG', ['A', 'C', 'U', 'G'])
(A, 2)
(C, 2)
(U, 2)
(G, 2)
>>> count_nucleotides('ACTGGCAT')
(A, 2)
(C, 2)
(T, 2)
(G, 2)
```

3.13 Sorting

Functions and list comprehensions from the previous section become particularly useful in practice in the context of sorting. Basic sorting can be accomplished through the built-in sorted function. For example, after importing the random module which among other things includes a function to shuffle a list in place:

```
>>> import random as rand
>>> l1 = [i for i in range(10)]
>>> l1
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> rand.shuffle(l1)
>>> l1
[0, 6, 1, 2, 9, 3, 7, 5, 8, 4]
>>> l1 = sorted(l1)
>>> l1
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> l1 = sorted(l1, reverse=True)
>>> l1
[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
```

In the last example, the default argument to the reverse argument of sorted is False; to reverse the list this argument must be explicitly defined to be True.

For a more complicated example of the usage of sorted, suppose there is an array of numbers sizes corresponding to sample sizes for some procedure, in which the entry in position 0 of the array is the size of the sample with id 0, the entry in position 1 is that of the sample with id 1, and so on. The goal is to create a function that extracts a list containing the sample ids of the largest samples by sample size. The position in the array is meaningful, so sorting the array will rearrange which sample size is contained in a particular index of the array. To preserve the association of the original array index (the sample id) with the sample size during the rearrangement of sorting, the enumerate function can be used. This function simply provides an association between the index of an array and the array element as a tuple, for example:

```
l = ['hydrocarbon', 'lipid', 'microbiology']
for index, word in enumerate(l):
    print (index, word)
```

Using this tool, to extract the n largest samples, the following function can be used:

```
def top_n(sizes, n):
    return [id for id, size in sorted(enumerate(sizes),
key = lambda
    item: item[1], reverse=True)][:n]
```

This is a complicated one-line function, but it can be broken down into elements from above, and it is worth examining closely as a check for understanding. Within the sorted function, the key argument is a function that is applied to each element of the list before it is sorted. In this case, it is used to designate that it is the second part of the (id, size) tuple that is to be used for comparison. Without this argument, sorting would have been performed based on the ids rather than the sizes of the samples.

The lambda expression is simply an inline function. From the above sorted example, the equivalent function to the given lambda expression would be:

```
def grab_second(some_tuple):
    return some_tuple[1]
def top_n(sizes, n):
    return [id for id, size in sorted(enumerate(sizes),
key = grab_second, reverse=True)][:n]
```

For extremely simple functions that do not need to be reused, a lambda expression provides a means to quickly implement the desired functionality. The second example is useful to note that functions can be passed as parameters to other functions too, which can allow for greater abstraction in code.

3.14 File I/O

File input and output in Python is also closely related to lists. Files in Python are very easy to work with. For a text file file.txt that is stored in the same directory as the Python code, the following will open the file and print line by line:

```
with open('file.txt') as text_file:
    for line in text_file:
      print line
```

As can be seen, an open file is manipulable in Python essentially as a list of strings, where in each line new line character designates the separation between entries in this list. The keyword with prompts the interpreter to properly handle the opening and closing of the file behind the scenes.

Files can be written equally simply, simply by opening them in write ('w') or append ('a') mode. If there is a second file file2.txt that needs to be written, the following will write every other line of file.txt to file2.txt:

```
    with open('file.txt') as read_file,  open('file2.txt',
"w") as write_file:
      to_write = True
      for line in read_file:
          if to_write:
               write_file.write(line)
               to_write = False
          else:
               to_write = True
```

3.15 Testing

Testing is a critical component of the process of writing code. The above examples were relatively simple and straightforward, and so their correctness is readily verifiable through informal testing, such as printing intermediate results and checking output. As a project grows though, it is crucial that tests exist to insure the correctness of code, especially through revisions. Further, as the example in Sect. 3.4 illustrated, Python's flexibility can be as much an asset as a danger, further prompting the need for tests. Basic testing can be accomplished quite simply; consider the grab_second helper function from Sect. 3.13, modified to be a little more robust:

```
import sys
def grab_second(some_tuple):
    if len(some_tuple) >=2:
        return some_tuple[1]
    else:
        return None

def test_two_tuple():
    two_tuple = ('one', 'two')
    return grab_second(two_tuple) == 'two'

def test_one_tuple():
    one_tuple = ('one',)
    return grab_second(one_tuple) == None

    if 'test' in sys.argv:
    if not test_two_tuple():
       print "Error with grab_second on tuple of length 2"
    if not test_one_tuple():
       print "Error with grab_second on tuple of length 1"
    print "Tests completed."
```

sys was imported so that the argv list can be accessed. Argv is simply a list of strings of arguments passed in on the command line. So if this file is test.py, and the command to run it was python test.py, then argv is ['python', 'test.py']. In this way, tests can be run conditionally in the case that one of the arguments passed in on the command line is 'test'. To run the tests with this example then, the command becomes python test.py test. Argv then contains the string 'test', the tests are executed, and any corresponding outputs are printed.

Python also has by default a testing framework called unittest, which can be imported to automate and expand upon the above process. This can be found in the online Python documentation [5].

4 Practical Usage

4.1 File Manipulation

The basic elements of Python have been explained and demonstrated. In some cases the interweaving between pieces is shown; however, it is often best to see the language in use as a whole to understand the implementation of the various components. To demonstrate the functionality of the language in actual usage, we will consider the common task of manipulating a text file to extract and reorder information. In this particular example, there are two input text files. The first contains output presenting a simplified set of sequencing data inspired by the output of the sequencing tool USEARCH [3]. This tool groups gene sequences into the bacterial genome from which they originate. The first column of the data is the gene sequence ID, for say gene1, gene2, gene3, and so on. The second column is the bacterial genome to which that gene was assigned by the sequencer. This file appears something like:

```
gene Sequence ID   Bacteria Genomes
gene1              bacterium1
gene1              bacterium2
gene1              bacterium2
gene3              bacterium2
gene3              bacterium2
gene2              bacterium1
    ...                 ...
```

The second file is simply a file containing all possible gene sequence IDs, since some gene sequence IDs may not have been assigned to a bacterium by the sequencer in the first file. This second file appears like this:

```
gene1
gene2
gene3
gene4
...
```

In the format in which these two files exists, it is unclear how many times each gene appears for each bacteria. It would be useful to use the above data to create a third file that has the data reorganized to display an overview of individual gene counts for each bacterium, such as:

```
Bacteria Genome  gene1  gene2  gene3  gene4 ...
bacterium1       1      1      3      0
bacterium2       5      0      2      5
...
```

To start, it would be useful to have a list in Python of each possible gene. This information is available from the second file, which will be called gene_file. So, a function that reads in a file of this format and outputs a list of the genes is a good starting point. Assuming the input to the function is an already opened gene_file, such a function could be:

```
def get_genes(gene_file):
    return [line.strip() for line in gene_file]
```

All the above function does is, via a list comprehension, go through each line in the now open file, remove any trailing white-space (in this case new line characters), and then save the resulting gene as a string in a list.

Now that the entire listing of genes is obtained, the counts of each gene for each bacterium need to be found and stored in an appropriate data structure. Ideally, this would have each bacterium associated with the counts of genes for that particular bacterium. A dictionary is the perfect data structure for this purpose. The names of the bacteria from the second column of the first file, called bacteria_file, will serve as the key, and the counts of each gene appearing with these bacteria will be the values. In this way, each line of the bacteria_file can be read in, and then the gene count of the appropriate bacterium can be updated, line by line.

As for storing the actual counts, Python once again provides a built-in means of handling this common task, in the form of a counter object. This object must be imported, but once that is accomplished, it provides an easy way to track counts. Queries to a counter object are performed with the [] syntax, much like a key lookup with a dictionary. If the query item exists, the count is returned. If the query item does not exist, the count defaults to return 0. Counts can be added and updated as is necessary in the expected way.

To summarize what now must be done, for each bacterium encountered in the bacteria_file, an entry must be made in the dictionary, where the entry is a counter object. If a counter already exists because the bacterium is in the dictionary, the preexisting counter should not be replaced by a new counter; rather the pre existing counter should be updated. Then, for the gene that is on the same line as the bacterium of interest, the bacterium's count of that gene in its counter object should be incremented by one. At the end, all bacteria in the file will have a counter in the dictionary, and this counter will store the number of times each gene originated with each individual bacterial genome.

Code for the outline above is as follows:

```
from collections import Counter
def get_bacteria_dict(bacteria_file):
    bacteria_dict = {} #create the overarching dictio-
nary to store the bacteria-counter key-value pairs
```

```
    for line in bacteria_file:
        gene, bacterium = line.strip().split('\t')
#remove the whitespace, and then split the tab delimited
line into two pieces
        bacteria_dict[bacterium] = bacteria_dict.get
(bacterium, Counter()) #if the bacterium already has a
value in the dictionary, don't modify it. Otherwise, make
the value a new counter object by specifying a default value.
        bacteria_dict[bacterium][gene] += 1 #increment
the gene count in the counter object (guaranteed to exist
by the previous line) by one. Remember that if the counter
did not have an entry for a gene already, the gene starts at a
count of 0. Therefore, after this line the count will be 1.
    return bacteria_dict
```

Now, a function exists that takes as input the name of a gene file and outputs a list of genes, and a function exists that takes as input a bacteria file and outputs a dictionary containing all the appropriate gene counts. Using these two functions, the third file, output_file, can be written. First, the gene list can serve as the source for the header of the file. Then, for each bacterium in the bacteria dictionary output by the second function, the counter associated with the bacterium can be queried for each gene in the gene list, and this output can be written to the file, in the same order as the genes appear in the gene list. In this way, the table of appropriate counts can be created. To implement this, two functions can be written, the first to write the header and the second to write the rows:

```
  def write_header(gene_list, output_file):
      output_file.write('\t') #provide one tab offset to
align the columns of the output file
      for gene in gene_list:
          output_file.write(gene + '\t') #write each
gene to the top line of the file, in a tab separated format.
      output_file.write('\n') #add a new line so that the next
function will begin the rows at the proper location.

  def write_rows(bacteria_dict, gene_list, output_file):
      for bacterium in bacteria_dict:
          output_file.write(bacterium + '\t') #write the
bacterium name as the first element of the column, with appro-
priate tab placement
      for gene in gene_list:
          gene_count = bacteria_dict[bacterium][gene]
          output_file.write(str(gene_count)+'\t')
#sequentially grab and write the counts of each gene for the
bacterium with appropriate tab separation
          output_file.write('\n') #write a new line before
the next bacterium is queried.
```

Finally, to tie this all together, a driver function can be used to open files and pass the appropriate parameters to functions in the correct order:

```
def main(gene_file_name, bacteria_file_name, output_file_name):
    with open(gene_file_name) as gene_file, open(bacteria_file_name) as bacteria_file, open(output_file_name) as output_file: #open all files as read only, except the output file to be written to.
        gene_list = get_genes(gene_file)
        bacteria_dict = get_bacteria_dict(bacteria_file)
        write_header(gene_list, output_file)
        write_rows(bacteria_dict, gene_list, output_file)
```

To run this, one would include the above functions in one file in the same directory as the bacteria and gene files. Then, after importing the main function to an interpreter session, the code should be runnable.

In this example, the basic types of strings and integers were combined with dictionaries, lists, and file I/O in the creation of several key loops to transform one set of data into another. These pieces ultimately form the backbone of the language. Due to space limitations the treatment of these components is necessarily incomplete. There is more functionality in lists, dictionaries, sets, and related structures that is implemented in Python. It is therefore critical for the interested reader to explore the full functionality that is offered by Python. In the development of python code, we often find that instead of first implementing our own version of a mundane task, a quick web search will reveal that this work would have been redundant. This is a common theme with Python and its rich online support ecosystem; often the applications of the language are unique, but the pieces required to implement the applications are readily available to be fit and molded to the problem at hand.

4.2 Third-Party Packages

This notion is taken even further with third-party packages. These are collections of functionality implemented in Python that essentially expand the installed language. The handling of large arrays of numbers in Python is sometimes clunky and slow, a problem that the NumPy package [6] addresses. Critical loops in a Python file can largely determine the speed at which a file executes; in this case Cython [7] offers a means to incorporate speedy, compiled C code directly into a Python file. Biological data is best handled robustly with third-party packages as well, such as scikit-bio [8], which is itself built on top of the machine learning functionality that is implemented by scikit-learn [9]. The list of these packages goes on and on, and it is well worth the time to search and build upon these packages in one's own work.

These third-party packages can be installed on a machine alongside the current Python distribution, frequently using a package manager called PIP. Once installed, packages can be imported and used in much the same way as the math functionality brought in with the import math statement. If NumPy has been installed on a machine, using the command import numpy will make the functions of the NumPy package callable in the current Python file.

4.3 Testing

Testing is a critical component of the process of writing code. The above examples were relatively simple and straightforward, and so their correctness is readily verifiable through informal testing, such as printing intermediate results and checking output. As a project grows though, it is crucial that tests exist to insure the correctness of code, especially through revisions. As the example in Sect. 3.4 illustrated, Python's flexibility can be as much an asset as a danger, further prompting the need for tests. Formal testing exists in the form of the unittest framework.

5 Conclusion

This overview has described the basic functionality of Python, with the hope that the functionality demonstrated has illustrated the ease with which often complicated tasks can be performed using features of the language. Writing python code often involves solving problems that are very similar to or exactly the same as problems that others have solved in the past. The same is true for debugging erroneous code, because searching the Internet for examples where others have received a similar error message is often the fastest way to identify the source of the error. Therefore as developers we find that searching the internet for helpful examples and code is an integral part of the learning process in this area and can speed up the development process significantly. Time spent learning Python can be invaluable, particularly in the scientific community. It offers quick, understandable ways to solve both the common and the complex tasks that present themselves on a daily basis to a researcher in Hydrocarbon and Lipid Microbiology.

References

1. Mason O, Scott N, Gonzalez A et al (2014) Metagenomics reveals sediment microbial community response to Deepwater Horizon oil spill. ISME J 8:1464–1475
2. QIIME Developers (2015) QIIME: quantitative insights into microbial ecology. http://www.qiime.org. Accessed 23 Mar 2015
3. Edgar R (2015) USEARCH: ultra-fast sequence analysis. http://drive5.com/usearch/. Accessed 23 Mar 2015
4. Caporaso G et al (2010) Python nearest alignment space termination tool. https://github.com/biocore/pynast. Accessed 23 Mar 2015
5. Python Software Foundation (2015) Documentation. https://www.python.org/doc/. Accessed 23 Mar 2015
6. NumPy Developers (2015) NumPy. http://www.numpy.org. Accessed 23 Mar 2015
7. Behnel S et al (2015) Cpython C-extensions for python. http://cython.org. Accessed 23 Mar 2015
8. Scikit-Bio (2015) Scikit-Bio. http://scikit-bio.org. Accessed 23 Mar 2015
9. Scikit-Learn (2015) Scikit-Learn: machine learning in python. http://scikit-learn.org/stable/. Accessed 23 Mar 2015

Protocols for Calculating Reaction Kinetics and Thermodynamics

Jan Dolfing

Abstract

Thermodynamics is central to our understanding of the ecology and physiology of microbial ecosystems. In physiological research, thermodynamics is used to determine the directionality of a reaction or the feasibility of a pathway. In ecological research, thermodynamics is used to determine the feasibility of a process and to rationalize the sequence of redox reactions in both natural environments and engineered biological systems and increasingly also as a basis to describe microbial activities as a function of environmental factors. This protocol provides a detailed annotated description of how to calculate change in Gibbs free energy values and redox potentials and a brief discussion on linking thermodynamics and kinetics for reactions of biological interest.

Keywords: Gibbs free energy, Kinetics, Redox potential, Thermodynamics

1 Introduction

Thermodynamics plays an integrative role in our understanding of the ecology and physiology of microbial ecosystems. Classical examples are (1) the paradigm that the sequence of redox reactions in natural environments follows thermodynamic logic in that the electron acceptor that allows the highest energy yield is used first [1] and (2) the free energy-based rationale for obligate product removal in syntrophy [2]: the organisms that function in these syntrophic associations operate close to the limits of what is thermodynamically possible [3]. Another area where the Gibbs free energy approach can provide pivotal understanding are bioelectrochemical systems, which offer not only promise for a range of biotechnological applications but also as elegant tools for studying the empirical relationship between thermodynamics and kinetics [4, 5]. The existence of a *fundamental* relationship – based on first principles – between thermodynamics and kinetics is still unresolved, but the recently proposed approach via the membrane potential [6] may well point us in the right direction.

In this chapter, we describe approaches to:

1. Calculate change in Gibbs free energy values for reactions of biological interest. Starting with appropriate values for standard conditions of 25°C, a pressure of 1 atm, concentrations of 1 M, and partial pressures of 1 atm, the methodology allows calculations for conditions that differ from these standard conditions.
2. Calculate redox potentials for reactions of biological interest.
3. Link thermodynamics and kinetics for reactions of biological interest.

2 Methods

Change in Gibbs free energy (ΔG) values for reactions of environmental and biotechnological interest [1, 7–9] is calculated in three steps. First the Gibbs free energy change (ΔG°) is calculated for standard conditions, i.e., for a temperature of 25°C, solutes at 1 M concentrations, and gases at partial pressures of 1 atm.

Next a temperature correction is applied if the temperature of interest is not 25°C.

Finally a correction is applied if the actual concentrations are not 1 M and/or the actual partial pressures are not 1 atm, with special attention for speciation and pH.

Redox potentials are linearly related to change in Gibbs free energy values and can thus be calculated directly from change in Gibbs free energy values.

2.1 Calculation of Change in Gibbs Free Energy Values

1. Change in Gibbs free energy values is calculated via

$$\Delta G^{o} = \sum G_{f}^{o}\text{products} - \sum G_{f}^{o}\text{reactants} \tag{1}$$

G_f° values for compounds of biological interest can be found in various sources [7, 10–24] (*see* **Notes** 1 to 6).

2. For temperatures different from 25°C, ΔG values are calculated with the Gibbs–Helmholtz equation:

$$\Delta G^{o}_{T_{\text{act}}} = \Delta G^{o}_{T_{\text{ref}}}(T_{\text{act}}/T_{\text{ref}}) + \Delta H^{o}_{T_{\text{ref}}} \cdot (T_{\text{ref}}T_{\text{act}})/T_{\text{ref}} \tag{2}$$

where T_{ref} is 298.15 K and T_{act} is the temperature of interest.

3. Actual concentrations are generally not 1 M or 1 atm (100 kPa). For a hypothetical reaction aA + bB → cC + dD, this is accounted for via the mass equation:

$$\Delta G = \Delta G^{o} + RT\ln[\text{C}]^{\text{c}} \cdot [\text{D}]^{\text{d}}/[\text{A}]^{\text{a}} \cdot [\text{B}]^{\text{b}} \tag{3}$$

where R is the universal gas constant (8.314 J/K·mol) and T is the temperature in Kelvin (*see* **Notes** 7 to 10).

4. Gibbs free energy values for weak acids and their conjugated bases are calculated by using the G_f value of the acid, with the formula

$$G_f = G_f^o + RT\ln\alpha \tag{4}$$

or the G_f value of the conjugated base with the formula

$$G_f = G_f^o + RT\ln(1-\alpha) \tag{5}$$

where $\alpha = 10\text{-pH}/(10\text{-pH} + 10\text{-p}K_a)$ (*see* **Note** 11).

2.2 Calculation of Redox Potentials

The conversion between change in Gibbs free energy values (ΔG) and redox potential (ΔE) is calculated via the relationship

$$\Delta G = -nF\Delta E \tag{6}$$

where n is the number of electrons involved and F is the Faraday constant (96.48 kJ/V) (*see* **Notes** 12 and 13).

2.3 Calculating Kinetics

The relationship between thermodynamics and kinetics is still poorly constrained. In spite of various recent efforts [25–28], a first principles-based formal coupling between thermodynamics and microbial kinetics remains elusive. Historically these models build on a tradition where thermodynamics is coupled to kinetics by including the equilibrium constant [29, 30]. The latest methodology [6] involves a function that includes the membrane potential as a scaling parameter. In this model, the thermodynamic limiting potential (F_T) is given by $F_T = 1/(e^E + 1)$, where E is calculated using $E = (\Delta G_{\text{reaction}} + \Delta G_{\text{mp}})/RT$. Here $\Delta G_{\text{reaction}}$ is ΔG per electron transferred and ΔG_{mp} is the Gibbs energy of maintaining the membrane potential, i.e., $\Delta G_{\text{mp}} = F\cdot\Delta\Psi$, where F is the Faraday constant and $\Delta\Psi$ is the membrane potential in mV. This F_T is then included as a multiplier in rate expressions, for example, the Michaelis–Menten equation. The issue here is that the actual $\Delta\Psi$ value of living, intact, unperturbed microorganisms is poorly constrained. LaRowe and coworkers [6] mention typical values between 100 and 240 mV. The F_T model is an improvement over previous models [26, 27] in that it uses only one adjustable parameter ($\Delta\Psi$) rather than three. Significantly, the model does not presuppose a biological (ATP-related) energy minimum [31].

2.4 Final Remarks

Calculating change in Gibbs free energy values and redox potentials for biological systems as outlined above is technically straightforward. The only challenge is to grasp that these systems are generally in chemical (but not thermodynamic) equilibrium [32]. Once this is understood, everything else (speciation, interchange between partial pressure in the gas phase, and concentration in the aqueous phase) falls into place, and it becomes clear that the outcome of the calculations will not be affected by the choice of the phase or speciation. Then it becomes also clear that if the system is not in (physico-) chemical equilibrium, e.g., when concentrations of gases in the aqueous phase are in disequilibrium with the gas phase, the calculations should be made for the phase of interest, which is generally the aqueous phase. Performing thermodynamic calculations by hand or with a self-made spreadsheet has the invaluable advantage that it stimulates pondering the logic behind the steps and equations and thus fosters understanding. However, that takes time and effort. An alternative way to do thermodynamic calculations is by adopting a user-friendly computer program such as "eQuilibrator – the biochemical thermodynamics calculator" [33, 34]. The eQuilibrator web interface allows easy calculation of Gibbs free energies of compounds and reactions at arbitrary pH and metabolite concentrations. Specially developed for biochemical calculations, the program has the added advantage that it takes ionic strength into account [35]. Potential disadvantages are that the program is only applicable for a temperature of 25°C and that due to its focus on classical biochemistry, compounds of environmental concern are not always present in the database. eQuilibrator is the heart of a movement to create an open framework for thermodynamics of reactions based on consistent estimation of Gibbs energy using component contributions [34, 36–38].

3 Notes

1. When perusing G_f° and H_f° values from the literature, it is important to pay attention to the units used. Many sources still use non-SI units (cal/mol or kcal/mol); 1 kcal = 4.184 kJ.
2. When selecting G_f° and H_f° values from the literature, it is important to pay attention to the state of the compound (gas, liquid, aqueous solution). Conversion of $G_f^\circ{}_{gas}$ values to $G_f^\circ{}_{aq}$ values can be done with the formula

$$G_f^\circ aq = G_f^\circ gas + RT \ln H$$

   where R is the universal gas constant (8.314 J/K.mol), T is temperature (K), and H is the Henry constant (atm·L·mol^{-1}) [14]. Again, it is important that the Henry constant is in the proper units. As an aside, Amend and Shock [13] list G_f° values for a series of gases for both the gas phase and in solution in the aqueous phase for a series of temperatures. Given the above relationship, the Henry constant can easily be calculated back for a range of temperatures. This is instructive!
3. Conversion of $G_f^\circ{}_{liquid}$ values to $G_f^\circ{}_{aq}$ values can be made with the formula $G_f^\circ aq = G_f^\circ liquid - RT \ln m$ where m is the

aqueous solubility (mol/L) [39]. Similarly, conversion of $G_f^\circ{}_{cryst}$ values to $G_f^\circ{}_{aq}$ values can be made with the formula $G_f^o aq = G_f^o cryst - RT\ln m$ where m is the aqueous solubility (mol/L) [39, 40]. When estimating $G_f^\circ{}_{aq}$ values based on literature data for the crystalline or liquid state as outlined above, it is important to consider the applicable state (solid or liquid) for the pure compound [24].

4. Thauer et al. [12] is a well-respected, widely used source of G_f° values. Its potential drawback is that, unlike most other databases, this database does not give H_f° values. ΔH is needed to make the necessary corrections if the temperature is not 25°C. Amend and Shock [13] do not give H_f° values either, but these authors give G_f° values for a series of temperatures. For most practical purposes, it will be acceptable to interpolate.
5. For those compounds where G_f° values are not readily available in the literature, group contribution methods can be used to obtain an approximate estimate. The group contribution method of Mavrovouniotis [40, 41] is very intuitive but has only values for the more common groups (aliphatics, aromatics, hydroxyl, carboxyl, and the like). Others have built on this framework to add values for, e.g., halogen substituents [14, 42].
6. Gibbs free energy of formation values based on quantum chemical methods are generally more accurate than previous values based on Benson's classical group contribution method. When available, the state-of-the-art quantum chemical-based values should be used [15].
7. The threshold concentration, that is, the minimum (in case of a reactant) or maximum (in case of a product) concentration of a certain compound at which a reaction is in thermodynamic equilibrium ($\Delta G = 0$) [43–45], is calculated in two steps. First the ΔG value for that reaction is calculated, and then the mass action formula is applied. For example, for hydrogenotrophic methanogenesis ($4H_2 + CO_2 \rightarrow CH_4 + 2H_2O$), $\Delta G^\circ = -130.7$ kJ/reaction. Hence, $\Delta G = -130.7 + RT \ln ([CH_4]/[CO_2]\cdot[H_2]^4)$. Under otherwise standard conditions (CH_4 and CO_2 at 1 atm, i.e., $\ln[CH_4] = \ln 1 = 0$ and $\ln [CO_2] = \ln 1 = 0$), $\Delta G = -130.7 - RT\ln[H_2]^4$. Other sample calculations are given in Dolfing et al. [23, 43].
8. The G_f° value for H^+ is 0. This is the value at a concentration of 1 M, i.e., at pH = 0. The correction for pH, for example, for pH = 7, follows from the mass equation (Eq. 3): $G_f\left(H^+{}_{pH}=7\right)=G_f^o\left(H^+{}_{pH}=0\right) + RT\ln 10\text{-}7=0+ 8.314 \times 298.15 \times \ln 10\text{-}7\,J/mol=-39.95\,kJ/mol$
9. The same logic can be applied to OH^-. Under standard conditions (1 M concentration, i.e., pH = 14), $G_f^o=-157.3\,kJ/mol$ [12]. Thus, at pH = 7, $G_f OH^-{}_{pH}=7=G_f^o OH^-{}_{pH}=14 + RT$

$\ln 10^{-7} = -157.3 + 8.314 \times 298.15 \times \ln 10^{-7}$ J/mol$=-197.2$ kJ/mol. Similarly, at pH = 9, that is, at $[OH^-] = 10^{-5}$, $G_f\,OH^-_{pH=9} = G_f\,OH^-_{pH=14} + RT\ln 10^{-5} = -157.3 + 8.314 \times 298.15 \times \ln 10^{-5}$ J/mol $= -185.8$ kJ/mol.

10. It may be insightful to point out that the sum of the ΔG° values for H^+ and OH^- at a given pH equals -237.2 kJ/mol, which is the G_f° value for H_2O. This is in agreement with the notion that $[H^+]\cdot[OH^-] = 10^{-14}$, combined with the fact that the change in Gibbs free energy for a reaction is directly related to the equilibrium constant (K) of that reaction:

$$\Delta G^{o} = -RT\ln K$$

With this relationship [7] in mind, it becomes intuitive that ΔG° provides information how far a reaction is away from equilibrium under standard conditions and that ΔG informs how far the reaction is away from equilibrium under a given condition.

11. For example, G_f° and pK values for acetic acid are -396.6 and 4.8 kJ/mol, respectively, while the G_f° value for acetate is -369.4 kJ/mol (note that [39] the relationship between these G_f° values conforms $G^{o}_{f\,\text{acetate}} = G^{o}_{f\,\text{acetic acid}} + RT\ln K$, i.e., at 25°C, $G^{o}_{f\,\text{acetate}} = G^{o}_{f\,\text{acetic acid}} + 5.70\,\text{p}K$). Calculating $G_{f\ \text{acetic acid}}$ and $G_{f\,\text{acetate}^-}$ by using the relationships $G_{f\,\text{acetic acid}} = G^{o}_{f\,\text{acetic acid}} + RT\ln\alpha$ (Eq. 4) and $G_{f\,\text{acetate}} = G^{o}_{f\,\text{acetate}} + RT\ln(1 - \alpha)$ (Eq. 5), where $\alpha = 10^{-\text{pH}}/\left(10^{-\text{pH}} + 10^{-\text{p}K_a}\right)$, will show (you) that for all pH values between 0 and 14, $G_{f\,\text{acetic acid}} = G_{f\,\text{acetate}^-} + G_{f\,H^+}$, as expected because these species are in thermodynamic equilibrium [46].

12. For example, consider the reaction $2H^+ + 2e^- \rightarrow H_2$. Given that G_f° $(H^+) = 0$; G_f° $(H_2) = 0$; G_f° $(e^-) = 0$ (data from Hanselmann [10]), this gives according to Eq. (1) $\Delta G^{\circ} = 0$ kJ, and consequently (Eq. 6) $\Delta E^{\circ} = 0$ V. Now consider idem at pH = 7. According to Eq. (3), $\Delta G = 0 - (2 \times -39.95) = 79.9$ kJ; therefore, (Eq. 6) $\Delta E = 79.9/-2 \times 96.48 = -0.414$ V. Thus, the redox potential for the H^+/H_2 redox couple is 0 at pH = 0 and -0.414 V at pH = 7.

13. Another example is to consider the reductive dechlorination of hexachlorobenzene to pentachlorobenzene. Based on G_f° values of -138.5 and -133.5 kJ/mol for C_6Cl_6 (aq) and C_6Cl_5H (aq), respectively, and values of -39.95 kJ/mol for H^+ at pH = 7 (see above) and -131.3 kJ/mol for Cl^- [7], reductive dechlorination of hexachlorobenzene to pentachlorobenzene according to C_6Cl_6 (aq) + H_2 (gas) $\rightarrow$ C_6Cl_5H (aq) + H^+ + Cl^- yields -166.2 kJ/mol at pH 7 under otherwise standard conditions (indicated as $\Delta G^{\circ\prime}$, where the prime indicates pH = 7). Using Eq. (6), the reduction potential for

this reaction is 861 mV. This leads up to the question what the redox potential is of redox couple C_6Cl_6/C_6Cl_5. Since, as indicated above, the redox potential for the H^+/H_2 redox couple at pH = 7 ($E^{\circ\prime}$) is −414 mV, it follows that the redox potential in the opposite direction, that is, with H_2 acting as an electron donor rather than an electron acceptor, is 414 mV. Thus, the redox potential of the C_6Cl_6/C_6Cl_5 couple is $861 - 414 = 447$ mV (note that Cl^- is traditionally not mentioned when couples like C_6Cl_6/C_6Cl_5 are discussed). A more direct (and intuitive) approach to calculate the redox potential of the C_6Cl_6/C_6Cl_5 redox couple is to directly consider the reaction $C_6Cl_6 + H^+ + 2e^- \rightarrow C_6Cl_5H + Cl^-$. Given the above G_f° values for the various reactants and products, this gives a $\Delta G^{\circ\prime}$ value of $-133.5 - 131.3 - (138.5 - 39.95) = -86.35$ kJ/mol; using Eq. (6), this indeed translates into $E^{\circ\prime} = 447$ mV.

References

1. Zehnder AJB, Stumm W (1988) Geochemistry and biogeochemistry of anaerobic habitats. In: Zehnder AJB (ed) Biology of anaerobic microorganisms. Wiley-Interscience, New York, pp 1–38
2. Stams AJM, Plugge CM (2009) Electron transfer in syntrophic communities of anaerobic bacteria and archaea. Nat Rev Microbiol 7:568–577
3. Jackson BE, McInerney MJ (2002) Anaerobic microbial metabolism can proceed close to thermodynamic limits. Nature 415:454–456
4. Stams AJM, de Bok FAM, Plugge CM, van Eekert MHA, Dolfing J, Schraa G (2006) Exocellular electron transfer in anaerobic microbial communities. Environ Microbiol 8:371–382
5. Dolfing J (2014) Syntrophy in microbial fuel cells. ISME J 8:4–5
6. LaRowe DE, Dale AW, Amend JP, Van Cappellen P (2012) Thermodynamic limitations on microbially catalyzed reaction rates. Geochim Cosmochim Acta 90:96–109
7. Stumm W, Morgan JJ (1996) Aquatic chemistry, 3rd edn. Wiley, New York
8. Dolfing J (1988) Acetogenesis. In: Zehnder AJB (ed) Biology of anaerobic microorganisms. Wiley-Interscience, New York, pp 417–468
9. Dolfing J (2003) Thermodynamic considerations for dehalogenation. In: Häggblom MM, Bossert ID (eds) Dehalogenation: microbial processes and environmental applications. Kluwer, Boston, pp 89–114
10. Hanselmann KW (1991) Microbial energetics applied to waste repositories. Experientia 47:645–687
11. Speight JG (2005) Lange's handbook of chemistry, 16th edn. McGraw-Hill, New York
12. Thauer RK, Jungermann K, Decker K (1977) Energy conservation in chemotrophic anaerobic bacteria. Bacteriol Rev 41:100–180
13. Amend JP, Shock EL (2001) Energetics of overall metabolic reactions of thermophilic and hyperthermophilic archaea and bacteria. FEMS Microbiol Rev 25:175–243
14. Dolfing J, Janssen DB (1994) Estimates of Gibbs free energies of formation of chlorinated aliphatic compounds. Biodegradation 5:21–28
15. Dolfing J, Novak I (2015) The Gibbs free energy of formation of halogenated benzenes, benzoates and phenols and their potential role as electron acceptors in anaerobic environments. Biodegradation 26:15–27
16. Helgeson HC, Owens CE, Knox AM, Richard L (1998) Calculation of the standard molal thermodynamic properties of crystalline, liquid, and gas organic molecules at high temperatures and pressures. Geochim Cosmochim Acta 62:985–1081
17. Richard L, Helgeson HC (1998) Calculation of the thermodynamic properties at elevated temperatures and pressures of saturated and aromatic high molecular weight solid and liquid hydrocarbons in kerogen, bitumen, petroleum, and other organic matter of biogeochemical interest. Geochim Cosmochim Acta 62:3591–3636

18. Amend JP, Helgeson HC (1997) Group additivity equations of state for calculating the standard molal thermodynamic properties of aqueous organic species at elevated temperatures and pressures. Geochim Cosmochim Acta 61:11–46
19. Richard L (2001) Calculation of the standard molal thermodynamic properties as a function of temperature and pressure of some geochemically important organic sulfur compounds. Geochim Cosmochim Acta 65:3827–3877
20. Holmes DA, Harrison BK, Dolfing J (1993) Estimation of Gibbs free energy of formation for chlorinated biphenyls. Environ Sci Technol 27:725–731
21. Huang C-L, Harrison BK, Madura J, Dolfing J (1996) Gibbs free energy of formation of PCDDs: evaluation of estimation methods and application for predicting dehalogenation pathways. Environ Toxicol Chem 15:824–836
22. Dolfing J, Novak I, Archelas A, Macarie H (2012) Gibbs free energy of formation of chlordecone and potential degradation products: implications for remediation strategies and environmental fate. Environ Sci Technol 46:8131–8139
23. Dolfing J, Xu A, Gray ND, Larter SR, Head IM (2009) The thermodynamic landscape of methanogenic PAH degradation. Microb Biotechnol 2:66–574
24. Dick JM, Evans KA, Holman AI, Jaraula CMB, Grice K (2013) Estimation and application of the thermodynamic properties of aqueous phenanthrene and isomers of methylphenanthrene at high temperature. Geochim Cosmochim Acta 122:247–266
25. Kleerebezem R, Stams AJM (2000) Kinetics of syntrophic cultures: a theoretical treatise on butyrate fermentation. Biotechnol Bioeng 67:529–543
26. Jin Q, Bethke CM (2005) Predicting the rate of microbial respiration in geochemical environments. Geochim Cosmochim Acta 69:1133–1143
27. Jin Q, Bethke CM (2007) The thermodynamics and kinetics of microbial metabolism. Am J Sci 307:643–677
28. Rodríguez J, Lema JM, Kleerebezem R (2008) Energy-based models for environmental biotechnology. Trends Biotechnol 26:366–374
29. Hoh CY, Cord-Ruwisch R (1996) A practical kinetic model that considers endproduct inhibition in anaerobic digestion processes by including the equilibrium constant. Biotechnol Bioeng 51:597–604
30. Fennell DE, Gossett JM (1998) Modeling the production and competition for hydrogen in a dechlorinating culture. Environ Sci Technol 32:2450–2460
31. Schink B (1997) Energetics of syntrophic cooperation in methanogenic degradation. Microbiol Mol Biol Rev 61:262–280
32. Mayumi D, Dolfing J, Sakata S, Maeda H, Miyagawa Y, Ikarashi M, Tamaki H, Takeuchi M, Nakatsu CH, Kamagata Y (2013) Carbon dioxide concentration dictates alternative methanogenic pathways in oil reservoirs. Nat Commun 4:1998. doi:10.1038/ncomms2998
33. Flamholz A, Noor E, Bar-Even A, Milo R (2012) eQuilibrator – the biochemical thermodynamics calculator. Nucleic Acids Res 40 (Database issue):D770–D775. doi:10.1093/nar/gkr874
34. Noor E, Bar-Even A, Flamholz A, Lubling Y, Davidi D, Milo R (2012) An integrated open framework for thermodynamics of reactions that combines accuracy and coverage. Bioinformatics 28:2037–2044
35. Vojinović V, von Stockar U (2009) Influence of uncertainties in pH, pMg, activity coefficients, metabolite concentrations, and other factors on the analysis of the thermodynamic feasibility of metabolic pathways. Biotechnol Bioeng 103:780–795
36. Rother K, Hoffmann S, Bulik S, Hoppe A, Gasteiger J, Holzhütter H-G (2010) IGERS: inferring Gibbs energy changes of biochemical reactions from reaction similarities. Biophys J 98:2478–2486
37. Noor E, Haraldsdóttir HS, Milo R, Fleming RMT (2013) Consistent estimation of Gibbs energy using component contributions. PLoS Comput Biol 9:e1003098
38. Noor E, Bar-Even A, Flamholz A, Reznik E, Liebermeister W, Milo R (2014) Pathway thermodynamics highlights kinetic obstacles in central metabolism. PLoS Comput Biol 10, e1003483
39. Dolfing J, Harrison BK (1992) The Gibbs free energy of formation of halogenated aromatic compounds and their potential role as electron acceptors in anaerobic environments. Environ Sci Technol 26:2213–2218
40. Mavrovouniotis ML (1990) Group contributions for estimating standard Gibbs energies of formation of biochemical compounds in aqueous solution. Biotechnol Bioeng 36:1070–1082
41. Mavrovouniotis ML (1991) Estimation of standard Gibbs energy changes of biotransformations. J Biol Chem 266:14440–14445
42. Jankowski MD, Henry CS, Broadbelt LJ, Hatzimanikatis V (2008) Group contribution

method for thermodynamic analysis of complex metabolic networks. Biophys J 95:1487–1499

43. Dolfing J, Larter SR, Head IM (2008) Thermodynamic constraints on methanogenic crude oil biodegradation. ISME J 2:442–452
44. Dolfing J (2013) Syntrophic propionate oxidation via butyrate: a novel window of opportunity under methanogenic conditions. Appl Environ Microbiol 79:4515–4516
45. Dolfing J (2014) Thermodynamic constraints on syntrophic acetate oxidation. Appl Environ Microbiol 80:15239–15241
46. Dolfing J, Xu A, Head IM (2010) Anomalous energy yields in thermodynamic calculations: importance of accounting for pH dependent organic acid speciation. ISME J 4:463–464

Modelling the Environmental Fate of Petroleum Hydrocarbons During Bioremediation

Guozhong Wu and Frédéric Coulon

Abstract

This chapter provides the key steps and parameters required for three different numerical modelling approaches to predict the environmental fate of petroleum hydrocarbons in contaminated soils during bioremediation. The first approach is the molecular dynamic simulation which is used to characterise the molecular-scale adsorption, the diffusion and the distribution of the saturate, aromatic, resin and asphaltene (SARA) fractions of oil. Such approach provides insights into the microscopic aggregation, the sequestration and the collision mechanisms which are essential for a better understanding of hydrocarbon bioavailability and biodegradation. The second approach is the use of fugacity modelling to compute the equilibrium distribution of the aliphatic and aromatic hydrocarbons in an environmental matrix composed of four compartments: soil, water, air and nonaqueous phase liquid (NAPL). Further to this, the contribution of the biotic and abiotic processes to the loss of petroleum hydrocarbons including (1) biodegradation in soil and NAPL, (2) advection in air, (3) leaching from soil and (4) diffusion at the soil–air, soil–water and soil–air boundaries can be estimated during biopiling experiments. The third approach is the use of machine learning (ML), an assumption-free data mining method, to predict the changes in the bioavailability of polycyclic aromatic hydrocarbons (PAHs) in contaminated soils. The main advantage of ML models is that they are data-based technique allowing computers to learn and recognise the patterns of the empirical data and work well with highly non-linear systems without relying on prior knowledge on bioremediation processes which make their prediction more realistic than conventional statistical methods. ML outputs can be integrated into microbial degradation models to support decision making for the assessment of bioremediation end points.

Keywords: Bioremediation, Fugacity modelling, Machine learning, Molecular simulation, Petroleum hydrocarbons

1 Introduction

Despite the fact that a variety of bioremediation technologies have been successfully applied for petroleum-contaminated sites, the complicated nature of soil and oil chemistry results in a lack of a universal technology that can be the solution for all contamination. Insights into the intricate interactions between oil and soil especially at the molecular scale are essential for a comprehensive appreciation of the fate of petroleum contaminants. Molecular simulation is a versatile tool to handle such issues. It represents a category of methods that convert the microscopic-level information (e.g. the position and the diffusion velocity of atoms) to the macroscopic properties (e.g. concentration profile, diffusion coefficient and thermodynamic properties of molecules) using statistical mechanics [1]. Simulation methods include quantum chemistry, molecular dynamics (MD), Monte Carlo and mesoscale simulation with the timescale ranging from femto- to nanoseconds (Fig. 1). The fast development of these methods during the last decades improved our understanding of the environmental behaviour of petroleum contaminants in the environment. Although it remains difficult to model the chemical interactions between soil and oil using an 'average' molecular formula for the natural soil and oil from various origins, several models have been developed for predicting the influence of the inorganic mineral and the organic matter of soil on the fate of oil. For example, the structural formula of soil minerals such as gibbsite, kaolinite, pyrophyllite and montmorillonites has now all been defined [2, 3]. Since the first three-dimensional model for soil organic matter (SOM) reported by Schulten and Schnitzer [4], an increasing number of SOM models have been developed such as the optimised dissolved organic matter model [5], the Leonardite humic acid model [6] and the Temple–-Northeastern–Birmingham humic acid model [7]. The progress made with these models has allowed subsequently to refine the chemical behaviour prediction of the interlayer structure of organo-clays by considering the molecular structure of both the SOM constituents and the minerals [8, 9]. Similarly, molecular models for individual hydrocarbon compounds and hydrocarbon fractions (group of hydrocarbons such as saturates, aromatic, olefins, resins, asphaltenes) of petroleum-derived products have been

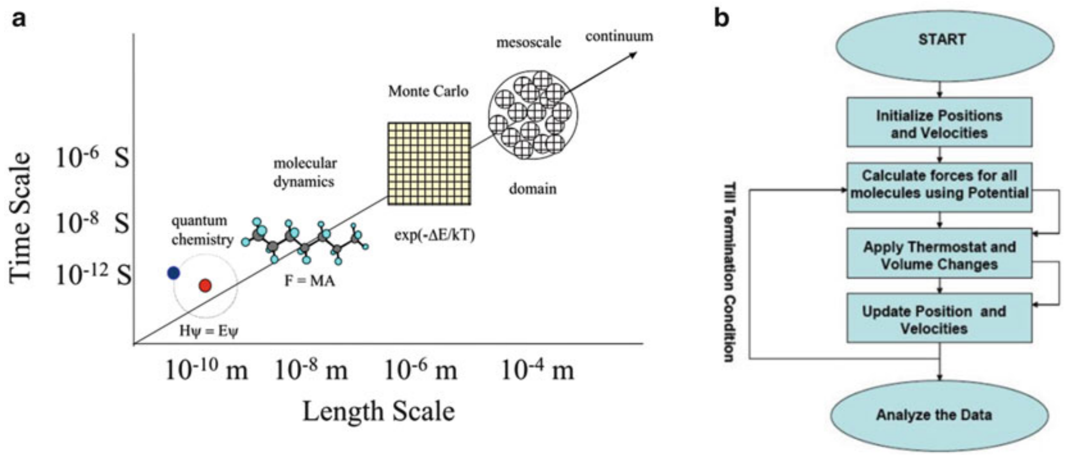


Fig. 1 Timescale (**a**) and steps (**b**) of molecular simulation methods

developed. For instance, the asphaltene molecules can be mimicked by the continental model [10] and the archipelago models [11], which have been validated by instrumental analysis (e.g. nuclear magnetic resonance, X-ray powder diffraction, Fourier-transformed infrared spectroscopy, thermogravimetric analysis) in terms of atom types, functional groups and molecular weight [12]. The development of these models makes possible to characterise the isotherm and thermodynamic parameters beyond the resolution of instrumental analysis such as the contribution of Coulomb electrostatic and van der Waals forces to the adsorption of oil contaminants on the soil mineral surface [1], the binding energy between SOM and oil pollutants [13] and the free energy for the penetration of polycyclic aromatic hydrocarbon through the bacterial membranes [14]. These physicochemical processes are highly important for bioavailability and biodegradation studies [15, 16].

However, the main challenge of using molecular simulation method is the timescale. The phenomena observed in the bioremediation process such as ageing often occurred in months or years; however, it is difficult to model such long timescale process by molecular simulation due to the extremely high computing data requirement. Our preliminary test indicated that it took about 1 month to simulate a 100 ns adsorption process of a system containing 28 asphaltene molecules and 20,000 water molecules using MD simulation on a server with 64 CPU cores (data not published). Accordingly, it would require 10^{12} months (computation time) to simulate such adsorption for only 1 day (real time). An alternative is to use fugacity modelling approach, which is a multimedia environmental fate model based on the thermodynamic theory of fugacity describing the bulk balance of a chemical substance in an ecosystem constituted by compartments [17]. The term 'fugacity' describes the 'fleeing' or 'escaping' tendency of a chemical species from an environmental compartment. Generally, the thermodynamic equilibrium includes chemical potential and fugacity. Chemical potential cannot be measured absolutely which is logarithmically related to the concentration of a compound [18]. By contrast, the major advantages of using fugacity are that (1) fugacity is equal to partial pressure under ideal conditions and is linearly related to concentration; (2) it is relatively convenient to transform the equations for chemical reaction, advective flow and non-diffusive transport rate into fugacity expressions with simple forms; and (3) it is easy to establish and solve sets of fugacity equations describing the complex behaviour of chemicals in multiphase and nonequilibrium environments. Various fugacity-based models have been developed such as generic models, air–water exchange models, soil–air exchange models, sediment–water exchange models, fish bioaccumulation models, sewage treatment plant models, indoor air models, plant uptake models, physiologically based pharmacokinetic models, multiple species bioaccumulation models, the EQuilibrium

Criterion model, the Level III ChemCAN model, the Level III CalTOX model, the QWASI (quantitative water, air, sediment interaction) model and the GloboPOP (chemical fate in the entire global environment) model [19]. These models have been previously applied for directing site remediation decisions [20], quantifying vapour emission from contaminated sites [21] and predicting the fate of organic compounds at landfill sites and constructed biopiles [22–24].

Both molecular simulation and fugacity modelling assume specific forms of mathematical equations, and statistical regression is only used to determine the unknown parameters in the equation. However, in some cases, the lack of knowledge on the bioremediation processes makes it difficult to accurately build these models due to the highly non-linear relationship between multiple variables. For example, there is not a universal form of equation that can be used to capture the relationship between ageing process and the bioavailability changes of different petroleum fractions in contaminated soils during composting processes. Instead machine learning is a data-based and assumption-free approach that distinguishes the data pattern and explores the relative influence of the independent variables on the dependent variables without relying on prior knowledge on the prediction process. The core of machine learning methods is allowing computers to extract general information from empirical data via 'learning' and 'recognising' the patterns of empirical data like a human brain. It has proven to be useful in modelling complex environmental processes, especially highly non-linear dynamic ecosystems such as the biosorption of metals [25–27], emissions of PAHs during combustion process [28] and bioavailability changes of PAHs in compost-amended soils [29].

This chapter provides the key steps and parameters needed when modelling the fate and transport of petroleum hydrocarbons using the above three modelling approaches. Their application can serve several purposes such as (1) improving our understanding of the chemical sources and their environmental fate and transport, (2) understanding the various factors that affect the environmental processes so that researchers can prioritise those factors that most need additional study, (3) providing tools to give a better mechanistic understanding of the degradation pattern and predict remediation end points and (4) enabling better risk-informed decisions during site remediation.

2 Molecular Dynamic Simulation

The protocol described below provides an example of MD simulation of the adsorption, the diffusion and the distribution of the different oil fractions on quartz surface (Fig. 2). The model construction, the energy minimisation and the molecular dynamic simulation can be

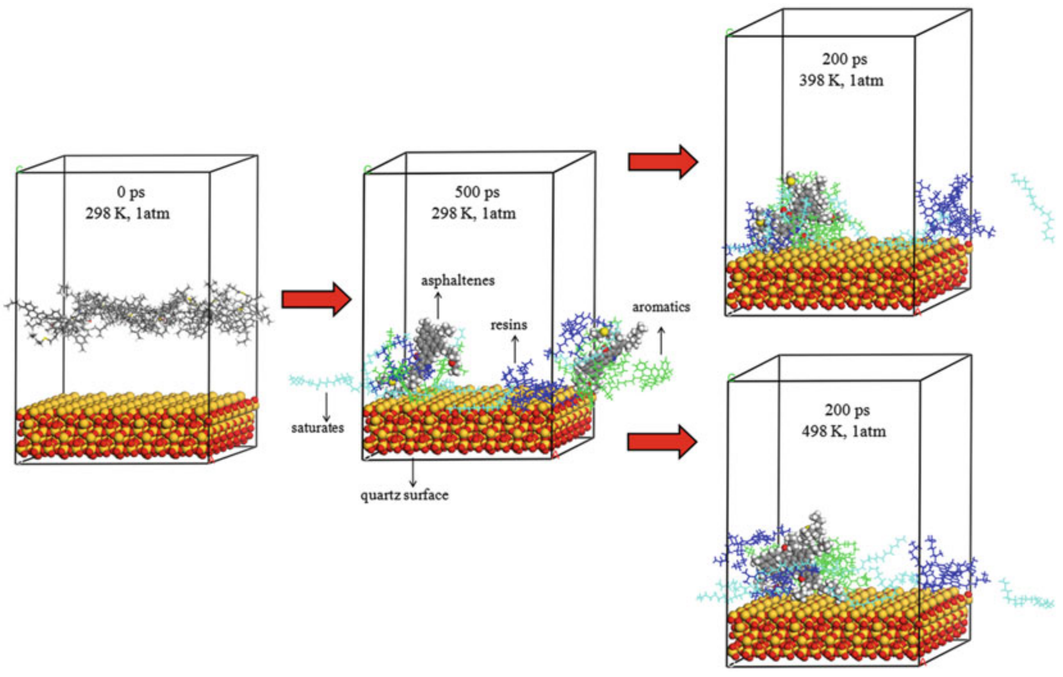


Fig. 2 Molecular modelling of the adsorption of asphaltenes, resins, aromatics and alkanes on quartz surface (adapted from [1])

carried out using a variety of software such as Materials Studio (http://accelrys.com/products/materials-studio/index.html), Gromacs (http://www.gromacs.org/), LAMMPs (http://lammps.sandia.gov/), DL_POLY (https://www.stfc.ac.uk/SCD/44516.aspx) and NAMD (http://www.ks.uiuc.edu/Research/namd/). Materials Studio was used in this protocol. Despite the differences in the program interface and computation performance among these programs, the modelling and simulation procedures are similar to that described in this protocol:

- Force field: select the condensed-phase optimised molecular potential for atomistic simulation studies (COMPASS) force field to describe the bonded and nonbonded potential of inter- and intra-molecule interactions [30] (*see* **Note** 1). Bonded potential includes quartic bond stretch, angle–bend contributions, torsion, out-of-plane angle and cross-coupling terms. Nonbonded potential consists of van der Waals interactions represented by the Lennard–Jones function and electrostatic interaction represented by the Coulombic equation (*see* **Note** 2).
- Oil layer modelling: select the following molecular compositions $C_{54}H_{65}NO_2S$ (MW: 792 g mol^{-1}) [31], $C_{50}H_{80}S$ (MW: 713 g mol^{-1}) [32], $C_{46}H_{50}S$ (MW: 635 g mol^{-1}) [33] and $C_{20}H_{42}$ (MW: 282 g mol^{-1}) to model the asphaltene, resin,

aromatic and saturate fractions of oil, respectively. Create an amorphous cell by packing these molecules with molecular number of 2, 4, 6 and 7, respectively, in a simulation box with 3D periodic boundary condition (*see* **Note** 3).

- Quartz surface modelling: create a quartz (1:1:1) surface using a quartz cell ($a = b = 0.4913$ nm, $c = 0.5405$ nm, $\alpha = \beta = 90°$, $\gamma = 120°$), build a 7 × 7 unit cell replica of the quartz surface and convert it into a 3D cell with 3D periodic boundary condition.
- Sorption system modelling: create a crystal-layered structure by adding the oil layer on the top of the quartz surface.
- Energy minimisation: use the smart minimiser approach to relax the sorption system and ensure that the system has no steric clashes or inappropriate geometry (*see* **Note** 4).
- NVT equilibration: perform an NVT MD simulation with constant number of atoms (N), volume (V) and temperature (T) to bring the system to the desired temperature for the simulation and to ensure the correct configuration of the orientation of the molecules. Assign the dynamic simulation parameters as follows: dynamics time (100 ps), time step (1 fs), temperature (298 K), thermostat (Andersen) and trajectory output (final structure) (*see* **Note** 5).
- NPT equilibration: perform an NPT MD simulation with constant number of atoms (N), pressure (P) and temperature (T) to stabilise the pressure and density of the system. Assign the dynamic simulation parameters as follows: dynamics time (100 ps), time step (1 fs), temperature (298 K), pressure (1 atm), thermostat (Andersen), barostat (Berendsen) and trajectory output (final structure).
- Production MD: perform an NVT MD simulation to the well-equilibrated system at the desired temperature and pressure. Parameters to be assigned include dynamics time (500 ps), time step (1 fs), temperature (298 K), thermostat (Andersen), trajectory output (full) and frequency of trajectory saved (every 5,000 steps).
- Data analysis: use the trajectory from the production MD simulation to analyse the mean square displacement, diffusion coefficient, root mean square deviations in structure, interaction energy, radial distribution functions and concentration profile of oil fractions on the quartz surface (*see* **Note** 6).

3 Fugacity Modelling

This protocol illustrates the application of fugacity models on a typical biopile with a volume of 624 m^3 and a mass of 750 t (Fig. 3).

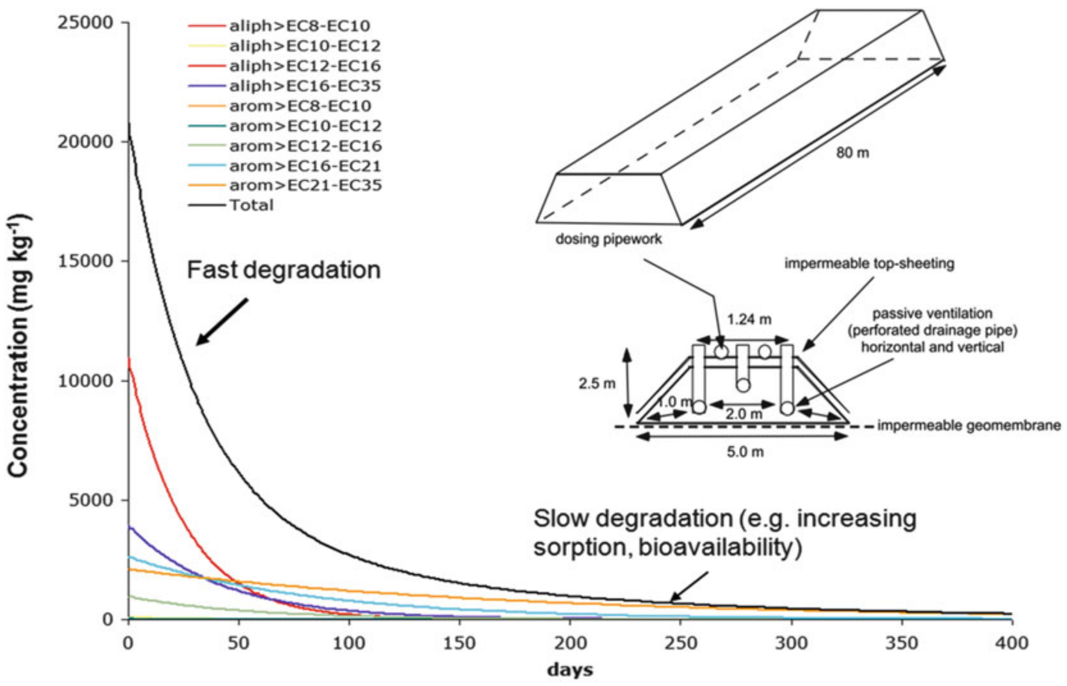


Fig. 3 Predicting the concentration of petroleum hydrocarbon fractions in contaminated soils during biopiling using fugacity modelling (adapted from [22, 23])

Level II fugacity model is used to calculate the steady-state and equilibrium distribution of petroleum hydrocarbon fractions in the contaminated soils. Level IV fugacity model is used to calculate the nonsteady-state chemical emission, advection, reaction, intermedia transport and residence time of petroleum hydrocarbons in the soil–oil matrix during biopiling. Level II and IV fugacity models can be easily coded in Microsoft Visual Basic for Applications (VBA) tool.

- Define four compartments in the biopile as follows: air, water, soil solids and nonaqueous phase liquid (NAPL). Calculate the volume of each compartment (an example of calculation is provided in the previous chapter 'Protocol for Biopile Construction Treating Contaminated Soils with Petroleum Hydrocarbons').
- Measure the physicochemical characteristics of soils (i.e. soil density, mass fraction of soil organic carbon) using standard methods [34]. Measure the total petroleum hydrocarbon (TPH) concentration in the soil (i.e. using the ultrasonic solvent extraction method described by Risdon et al. [35]). Assume that the measured TPH concentration is equal to the NAPL concentration.

- Divide the TPH into aliphatic and aromatic fractions. Divide each fraction into groups according to equivalent carbon number (*see* **Note 7**). Compile the physicochemical properties of petroleum hydrocarbon fractions (i.e. molecular weight, water solubility, vapour pressure, Henry's law constant, log K_{ow}, log K_{oc}, density) from books [19, 36].
- Calculate the half-lives of each petroleum hydrocarbon fraction in air, water and soil using the Estimation Program Interface Suite (http://www.srcinc.com). Assume the half-lives in NAPL to be equal to that in the bulk soil.
- Calculate the fugacity capacity (Z) for each compartment as follows:

Compartment	**Fugacity capacities (mol m^{-3} Pa)**	**Constant definitions and units**
Air	$Z_{AIR} = 1/R \cdot T$	R = the gas constant (8.314 Pa m^3 mol^{-1} K^{-1}) T = temperature (K)
Water	$Z_{WATER} = 1/H$	H = Henry's law constant (Pa m^3 mol^{-1})
NAPL	$Z_{NAPL} = K_{ow}/H$	K_{ow} = octanol–water partition coefficient
Soil solids	$Z_{SOIL} = K_{oc} * f_{OC} * \rho_{SOLIDS} * Z_{WATER}/1{,}000$	K_{oc} = organic carbon partition coefficient (L kg^{-1}) ρ_{SOLIDS} = density of soil solids (kg m^{-3}) f_{OC} = mass fraction of soil organic carbon (g g^{-1})
Bulk soil (without NAPL phase)	$Z_{BULK} = \phi_{AIR} Z_{AIR} + \phi_{WATER} Z_{WATER} + \phi_{SOIL} Z_{SOIL}$	ϕ_{AIR} = volume fraction of air (m^3) ϕ_{WATER} = volume fraction of water (m^3) ϕ_{SOIL} = volume fraction of soil solids (m^3)
Bulk soil	$Z_T = \phi_{AIR} Z_{AIR} + \phi_{WATER} Z_{WATER} + \phi_{SOIL} Z_{SOIL} + \phi_{NAPL} Z_{NAPL}$	ϕ_{NAPL} = volume fraction of NAPL (m^3)

- Steady-state and equilibrium distribution: assume the fugacity is equal and constant in all compartments. Calculate the fugacity (f, Pa) and the concentration of each petroleum hydrocarbon fraction (C, mol m^{-3}) in each compartment using the total mass of TPH (M, mol), the fugacity capacity (Z, mol m^{-3} Pa^{-1}) and the volume of each compartment (V, m^3) as follows: $M = \sum V_iC_i = f\sum V_iZ_i$.

- Define the reaction and advection processes that occurred during biopiling. The former include the biodegradation in bulk soil and NAPL phase, which are simplified as first-order kinetic reaction (*see* **Note 8**). The latter processes include the advection in air, leaching from soil and volatilisation to the surrounding air (i.e. diffusion in soil–air, diffusion in soil–water and diffusion in the soil–air boundary).
- Calculate the D values (a transport parameter similar in principle to rate constant) for each process with the formula and parameters listed as follows (*see* **Note 9**):

Compartment process	Equation
Reaction in bulk soil	$D_R = k_R(V_A + V_S + V_W) \cdot Z_{BULK}$
Reaction in NAPL	$D_{R\_NAPL} = k_{R\_NAPL} \cdot V_{NAPL} \cdot Z_{NAPL}$
Advection in air	$D_A = G_{AIR} \cdot Z_{AIR}$
Leaching from soil	$D_L = G_L \cdot Z_W$
Volatilisation	$D_V = [(1/D_E) + 1/(D_A + D_W)]^{-1}$
Diffusion in soil–air	$D_A = A \cdot B_A \cdot Z_{AIR}/\Upsilon_D$
Diffusion in soil–water	$D_W = A \cdot B_W \cdot Z_{WATER}/\Upsilon_D$
Diffusion across the soil–air boundary	$D_E = A \cdot K_V \cdot Z_{AIR}$
Overall bulk soil D value	$D_T = D_R + D_A + D_V + D_L + D_{R\_NAPL}$

A is the surface area of biopile (m^2)*;* V_i is the volume of compartment i (m^3). A = air; S = soil solids; W = water. k_R is the first-order reaction rate constant for the fraction considered in the bulk soil, calculated as $\ln(2)/T_O$ (h^{-1}); k_{R_NAPL} is the first-order reaction rate constant for the NAPL (h^{-1}); G_{AIR} is the air flow rate through the biopile (6.24 m^3 h^{-1}); G_L is the water flow rate through the biopile (arbitrary rate of 2.1×10^{-3} m^3 h^{-1}); B_A is the effective diffusion coefficient for soil–air, derived from the Millington–Quirk equation (m^2 h^{-1}); B_W is the effective diffusion coefficient for soil–water, derived from the Millington–-Quirk equation (m^2 h^{-1}); Υ_D is the (non-tortuous) diffusion path length, set at the mean depth of the biopile (m); K_V is the partial mass transfer coefficient in the air side of the soil–air interface (m h^{-1}).

- Nonsteady-state dynamic fate: calculate the dynamic changes in fugacity (f_i) by solving the differential equation as follows –

$df_i/dt = -D_{Ti}f_i/V_T Z_{BULKi}$. Then, predict the total concentration of all fractions in the bulk soil at each time step as follows: $C_T(t) = \sum_{i=1}^{N} C_i(t)$.

4 Machine Learning Modelling

The protocol below described the application of ML models to predict the PAH bioavailability changes in contaminated soils during the bioremediation process via compost amendment (Fig. 4). Among the methods developed in ML, the artificial neural network (ANN)-based models, the tree-learning techniques, the rule-learning algorithms and linear regression are the most widely used and therefore are described in this protocol. This protocol provides example of using these models followed by parameterisation, optimisation and validation. It does not elaborate much on the computing theory of these models which is available from previous publications [29, 37]. All calculations were performed using the open-source software WEKA (http://www.cs.waikato.ac.nz/ml/weka/).

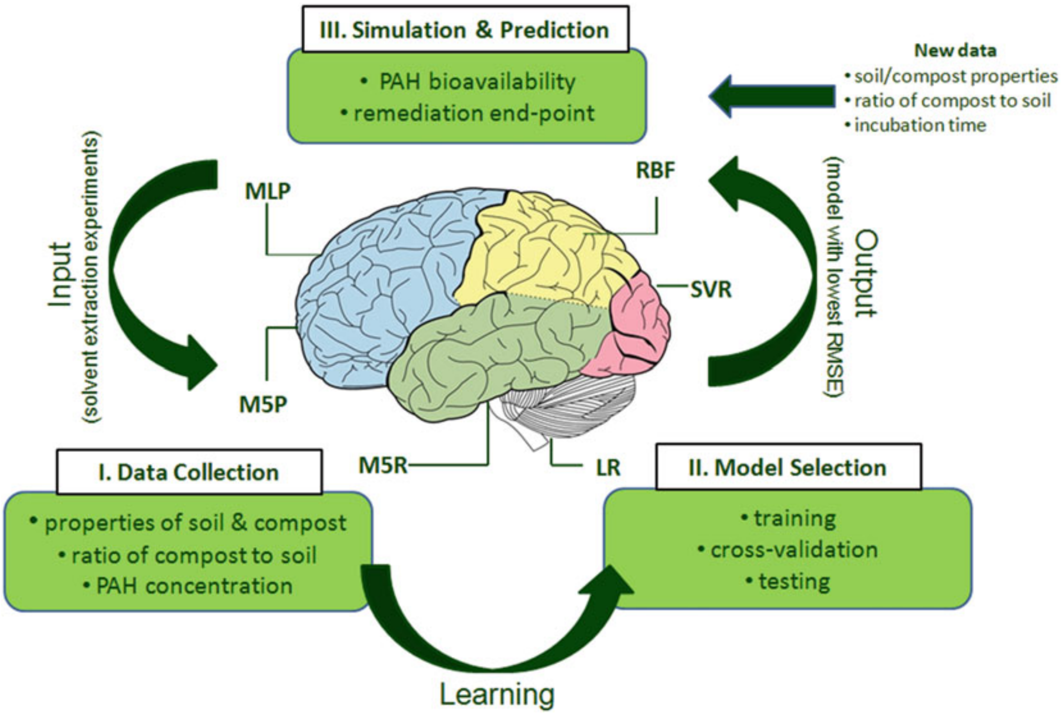


Fig. 4 Machine learning models to predict PAH bioavailability changes in soils after compost amendment. As demonstrated in the figure, the three steps included in machine learning are data collection, model selection and simulation and prediction. The input data was trained using six models including multilayer perceptron (MLP), radial basis function (RBF), support vector regression (SVR), M5 model tree (M5P), M5 model rules (M5R) and linear regression (LR). The model with the lowest root mean square error (RMSE) was finally selected for predicting PAH bioavailability and remediation end point by providing a new data set

- Incubation experiments: the data source for the ML modelling is obtained from laboratory experiments. In this example, two types of mature compost into three contaminated soils were considered, and the change in the bioavailability of the 16 US EPA priority PAHs was monitored [38]. The protocols for the physicochemical characterisation of the soils and the composts, the compost amendment regimes, the incubation time and the chemical analysis and quantification of the PAHs are available from Wu et al. [34].
- Data collection: collect the data on soil and compost properties (i.e. moisture, organic carbon of soil, total nitrogen, total phosphorus, available phosphorus, loss on ignition and soil texture), ratio of compost to soil (250 and 750 t ha^{-1}), initial concentration of total PAH, incubation time (i.e. 0, 3, 6 and 8 months) and the bioavailable concentration of PAHs at different incubation times as input parameters. Define the bioavailable concentration as dependent variable and the others as independent variables.
- Multilayer perceptron (MLP) training: build an MLP network and set the initial number of nodes in the input layer and output layer as 11 and 1, respectively. Fix the value of momentum term as 0.2. Use the CVParameterSelection module in the WEKA software to optimise two model parameters (i.e. the learning rate and the number of nodes in the hidden layer (initial value: 8–23)). The WEKA will output the root mean square errors (RMSE) and the hidden-input and hidden-output connection weights. Use the connection weights to calculate the relative contribution of input variables (i.e. soil and compost properties) to the predictive output (i.e. bioavailable concentration of PAHs) (*see* **Note** 10).
- Radial basis function (RBF) training: choose the k-means clustering algorithm to obtain RBF centres, select symmetric multivariate Gaussian function to fit the data and use CVParameterSelection module to optimise two parameters (i.e. the number of Gaussian radial basis functions (GRBFs) for the k-means algorithm and the minimum of standard deviation for the GRBFs) (*see* **Note** 11).
- Support vector regression (SVR) training: select the sequential minimal optimisation (SMO) algorithm to iteratively train the model and rearrange the non-linear data into linear data. The only parameter to optimise is the complexity parameter that determines the trade-off between the complexity of SMO model and the tolerance of errors. Use a correlation image to visualise the relationship between the input variables and the predictive output.
- M5 model tree (M5P), M5 model rules (M5R) and linear regression (LR) training: use the default parameters in the

WEKA software to train the models, which do not need optimisation.

- Cross validation: split the input data into ten groups randomly. For each model, use the instances from nine groups for model training, while use the remaining one group for model testing. Repeat this process ten times using a different group for testing at each cycle. The above validation method is termed as tenfold cross validation, which should be repeated ten times by re-splitting the data into ten groups. Average the root mean square error (RMSE) from the 100 (10×10) calculation to obtain the overall RMSE. Use RMSE to evaluate the performance of each model in predicting PAH bioavailability. The lower the RMSE, the better the goodness of fit.
- Simulation: overall 96 RMSE (16 PAHs $\times$ 6 models for each PAH) are obtained after training each model for each PAH. Select the best model (with the lowest RMSE) for each PAH as the final model. Use these well-trained models for predicting the bioavailable concentration of each PAH by inputting a new set of data that is not used in the model training or validation. For example, if we want to predict the bioavailable concentration after 12 months which was not measured in the incubation experiment, we just modify the variable of incubation time as 12, input it along with other variables (e.g. soil and compost properties) into the final models and run the models.

5 Notes

1. Force field is one of the most important mathematical functions to be selected prior to molecular dynamic simulation, which should be capable of predicting the structural, vibrational and thermophysical properties for substances in the model system. There are several options for force field such as COMPASS, CVFF, PCFF, GROMOS, CHARMM, AMBER, CLAYFF and OPLS-AA. Before application, the force field should be parameterised (e.g. equilibrium bond distances and angles, bond and angle force constants, dihedral angles, atom charges and Lennard–Jones parameters) via X-ray crystallography, vibrational spectra and quantum mechanics calculations. Alternatively, the force field can be empirically selected based on previous publications. For example, the COMPASS force field has been validated for a number of organic and inorganic molecules such as alkanes and benzene-fused ring compounds [30]. The CLAYFF [39] force field and the CHARMM force field optimised by Lopes et al. [40] can be effectively used for simulation of clay and silica, respectively. The GROMOS force field was initially designed for modelling biomolecules but has

been increasingly employed to model the petroleum-based system (e.g. asphaltenes and polycyclic aromatic hydrocarbons) after modification [10, 31, 41, 42].

2. The nonbonded interactions usually dominate simulations. We recommend using Gromacs (free software) for MD simulation as a lot of algorithmic optimisations have been introduced in the code, and thereby it is extremely fast at calculating the nonbonded interactions. The program is better run under Linux operating system with command line options for input and output files (although there is also a Windows version). For beginners not familiar with the Linux system, we recommend to use Materials Studio (commercial software) which has a more friendly user interface under Windows system.
3. The number of molecules in the MD simulation system should be decided on the molecular weight and concentration ratio of the oil fractions in the laboratory experiments. The periodical boundary condition (PBC) means that if any atoms leave a simulation box in one direction, then they will enter the simulation box from the opposite direction. The application of PBC can avoid problems with boundary effects caused by finite size and make the system more like an infinite one.
4. There are several algorithms to perform energy minimisation such as steepest descent, conjugate gradient and Newton--Raphson. Convergence levels of iterations should be assigned for these algorithms. The Smart Minimizer is a module in the Materials Studio package which executes the above algorithms in sequence according to the changes in the energy potential of the model system. In this protocol, the convergence levels of iterations are set as 1,000, 10 and 0.1 kJ (mol Å)$^{-1}$ for the above three methods, respectively.
5. A full list of the parameters to set before running MD simulation can be found from the online manual of Gromacs software at http://manual.gromacs.org/online/mdp_opt.html.
6. The molecular simulation software offers a large selection of flexible tools for analysing the structural, statistic, dynamic and thermodynamic properties of the model system using the trajectories. A detail description of the data analysis principles can be found from Frenkel and Smit [43] and the online manual of Gromacs (http://www.gromacs.org/).
7. Equivalent carbon number is determined from boiling point or the retention time of the compounds in gas chromatography column, which is more appropriate than the actual carbon number of chemicals [44]. In addition to fractionating the petroleum oil into groups, specific petroleum hydrocarbons can also be selected as indicator compounds for fugacity modelling.

8. All the reaction processes are expressed in first-order form as a function of concentration. If a chemical is susceptible to several reactions in the same phase, the rate constants for the reaction are simply added to make the total rate constant. The strategy is to force first-order kinetics on systems by lumping parameters in the rate constant (i.e. express the second-order rate reactions in the form of pseudo first-order rate reaction to circumvent complex reaction rate equations). Such transformation will make the subsequent calculations much easier.
9. The general form for reaction and advection process is $D_R = k \cdot V \cdot Z$ (k, reaction rate constant, h^{-1}; V, volume, m^3) and $D_A = G \cdot Z$ (G, fluid flow rate, m^3 h^{-1}), respectively.
10. The nodes in the input layer are the input-independent variables, while those in the hidden layer and output layer are processing elements with non-linear activation functions such as sigmoid function that enable the network to solve complex non-linear problems [45]. The number of nodes in the hidden layer should range from ($2n^{1/2}+m$) to ($2n+1$), where n and m are the number of nodes in the input layer and output layer, respectively [46]. Each node in one layer is connected with a certain weight to every node in the following layer. During model training, the predicted results are compared with the experimental results to compute the value of error. The connection weights are adjusted to reduce the error. This process is repeated for a sufficiently large number of cycles until the error is minimised.
11. RBF is another ANN model, which differs from the MLP in that it uses a special class of activation functions known as radial basis functions (e.g. Gaussian function) in the hidden layer.

References

1. Wu G, He L, Chen D (2013) Sorption and distribution of asphaltene, resin, aromatic and saturate fractions of heavy crude oil on quartz surface: molecular dynamic simulation. Chemosphere 92:1465–1471
2. Viani A, Gualtieri AF, Artioli G (2002) The nature of disorder in montmorillonite by simulation of X-ray powder patterns. Am Mineral 87:966–975
3. Teppen BJ, Rasmussen K, Bertsch PM, Miller DM, Schäfer L (1997) Molecular dynamics modeling of clay minerals. 1. Gibbsite, kaolinite, pyrophyllite, and beidellite. J Phys Chem B 101:1579–1587
4. Schulten HR, Schnitzer M (1995) Three-dimensional models for humic acids and soil organic matter. Naturwissenschaften 82:487–498
5. Sutton R, Sposito G, Diallo MS, Schulten HR (2005) Molecular simulation of a model of dissolved organic matter. Environ Toxicol Chem 24:1902–1911
6. Niederer C, Goss KU (2007) Quantum chemical modeling of humic acid/air equilibrium partitioning of organic vapors. Environ Sci Technol 41:3646–3652
7. Sein JLT, Varnum JM, Jansen SA (1999) Conformational modeling of a new building block of humic acid: approaches to the lowest energy conformer. Environ Sci Technol 33:546–552
8. Sutton R, Sposito G (2006) Molecular simulation of humic substance–Ca-montmorillonite complexes. Geochim Cosmochim Acta 70:3566–3581

9. Zeng QH, Yu AB, Lu GQ, Standish RK (2003) Molecular dynamics simulation of organic–inorganic nanocomposites: layering behavior and interlayer structure of organoclays. Chem Mater 15:4732–4738
10. Kuznicki T, Masliyah JH, Bhattacharjee S (2009) Aggregation and partitioning of model asphaltenes at toluene-water interfaces: molecular dynamics simulations. Energy Fuel 23:5027–5035
11. Alshareef AH, Scherer A, Tan X, Azyat K, Stryker JM, Tykwinski RR, Gray MR (2012) Effect of chemical structure on the cracking and coking of archipelago model compounds representative of asphaltenes. Energy Fuel 26:1828–1843
12. Sjöblom J, Simon S, Xu Z (2015) Model molecules mimicking asphaltenes. Adv Colloid Interface Sci 218:1–16
13. Wu G, Zhu X, Ji H, Chen D (2015) Molecular modeling of interactions between heavy crude oil and the soil organic matter coated quartz surface. Chemosphere 119:242–249
14. Ren B, Gao H, Cao Y, Jia L (2015) In silico understanding of the cyclodextrin–phenanthrene hybrid assemblies in both aqueous medium and bacterial membranes. J Hazard Mater 285:148–156
15. Semple KT, Morriss AWJ, Paton GI (2003) Bioavailability of hydrophobic organic contaminants in soils: fundamental concepts and techniques for analysis. Eur J Soil Sci 54:809–818
16. Reid BJ, Jones KC, Semple KT (2000) Bioavailability of persistent organic pollutants in soils and sediments – a perspective on mechanisms, consequences and assessment. Environ Pollut 108:103–112
17. Bru R, Maria Carrasco J, Costa Paraíba L (1998) Unsteady state fugacity model by a dynamic control system. Appl Math Model 22:485–494
18. Lewis GN (1901) The law of physico-chemical change. Proc Am Acad Arts Sci 37:49–69
19. Mackay D (2001) Multimedia environmental models: the fugacity approach. Lewis, Chelsea
20. Pollard SJT, Hoffmann RE, Hrudey SE (1993) Screening of risk management options for abandoned wood-preserving plant sites in Alberta, Canada. Can J Civ Eng 20:787–800
21. Mills WJ, Bennett ER, Schmidt CE, Thibodeaux LJ (2004) Obtaining quantitative vapor emissions estimates of polychlorinated biphenyls and other semivolatile organic compounds from contaminated sites. Environ Toxicol Chem 23:2457–2464
22. Coulon F, Whelan MJ, Paton GI, Semple KT, Villa R, Pollard SJT (2010) Multimedia fate of petroleum hydrocarbons in the soil: oil matrix of constructed biopiles. Chemosphere 81:1454–1462
23. Pollard SJT, Hough RL, Kim KH, Bellarby J, Paton G, Semple KT, Coulon F (2008) Fugacity modelling to predict the distribution of organic contaminants in the soil:oil matrix of constructed biopiles. Chemosphere 71:1432–1439
24. Shafi S, Sweetman A, Hough RL, Smith R, Rosevear A, Pollard SJT (2006) Evaluating fugacity models for trace components in landfill gas. Environ Pollut 144:1013–1023
25. Turan NG, Mesci B, Ozgonenel O (2011) Artificial neural network (ANN) approach for modeling Zn (II) adsorption from leachate using a new biosorbent. Chem Eng J 173:98–105
26. Giri A, Patel R, Mahapatra S (2011) Artificial neural network (ANN) approach for modelling of arsenic (III) biosorption from aqueous solution by living cells of *Bacillus cereus* biomass. Chem Eng J 178:15–25
27. Turan NG, Mesci B, Ozgonenel O (2011) The use of artificial neural networks (ANN) for modeling of adsorption of Cu (II) from industrial leachate by pumice. Chem Eng J 171:1091–1097
28. Inal F (2006) Artificial neural network predictions of polycyclic aromatic hydrocarbon formation in premixed n-heptane flames. Fuel Process Technol 87:1031–1036
29. Wu G, Kechavarzi C, Li X, Wu S, Pollard SJ, Sui H, Coulon F (2013) Machine learning models for predicting PAHs bioavailability in compost amended soils. Chem Eng J 223:747–754
30. Sun H (1998) COMPASS: an ab initio force-field optimized for condensed-phase applications overview with details on alkane and benzene compounds. J Phys Chem B 102:7338–7364
31. Kuznicki T, Masliyah JH, Bhattacharjee S (2008) Molecular dynamics study of model molecules resembling asphaltene-like structures in aqueous organic solvent systems. Energy Fuel 22:2379–2389
32. Murgich J, Rodríguez J, Aray Y (1996) Molecular recognition and molecular mechanics of micelles of some model asphaltenes and resins. Energy Fuel 10:68–76
33. Verstraete J, Schnongs P, Dulot H, Hudebine D (2010) Molecular reconstruction of heavy petroleum residue fractions. Chem Eng Sci 65:304–312
34. Wu G, Kechavarzi C, Li X, Sui H, Pollard SJT, Coulon F (2013) Influence of mature compost

amendment on total and bioavailable polycyclic aromatic hydrocarbons in contaminated soils. Chemosphere 90:2240–2246

35. Risdon GC, Pollard SJT, Brassington KJ, McEwan JN, Paton GI, Semple KT, Coulon F (2008) Development of an analytical procedure for weathered hydrocarbon contaminated soils within a UK risk-based framework. Anal Chem 80:7090–7096
36. TPHCWG (1998) Total petroleum hydrocarbon criteria working group series volume 2: composition of petroleum mixtures. Amherst Scientific, Amherst
37. Mitchell T (1997) Machine learning. McGraw Hill, New York
38. USEPA (1989) Method 610-Polynuclear Aromatic Hydrocarbons, methods for organic chemical analysis of municipal and industrial wastewater. US Environmental Protection Agency, Washington, DC
39. Cygan RT, Liang JJ, Kalinichev AG (2004) Molecular models of hydroxide, oxyhydroxide, and clay phases and the development of a general force field. J Phys Chem B 108:1255–1266
40. Lopes PEM, Murashov V, Tazi M, Demchuk E, MacKerell AD (2006) Development of an empirical force field for silica. Application to the quartz-water interface. J Phys Chem B 110:2782–2792
41. Jian C, Tang T, Bhattacharjee S (2013) Probing the effect of side-chain length on the aggregation of a model asphaltene using molecular dynamics simulations. Energy Fuel 27:2057–2067
42. Bandela A, Chinta JP, Hinge VK, Dikundwar AG, Row TNG, Rao CP (2011) Recognition of polycyclic aromatic hydrocarbons and their derivatives by the 1,3-dinaphthalimide conjugate of calix[4]arene: emission, absorption, crystal structures, and computational studies. J Org Chem 76:1742–1750
43. Frenkel D, Smit B (2002) Understanding molecular simulation: from algorithms to applications. Academic, San Diego
44. Brown DG, Knightes CD, Peters CA (1999) Risk assessment for polycyclic aromatic hydrocarbon NAPLs using component fractions. Environ Sci Technol 33:4357–4363
45. Haykin S (1999) Neural networks: a comprehensive foundation. Prentice Hall, New York
46. Fletcher D, Goss E (1993) Forecasting with neural networks. An application using bankruptcy data. Inf Manag 24:159–167