UNIVERSITÄT ZU LÜBECK

From the Institute of Experimental Dermatology
of the University of Lübeck
Director: Prof. Dr. Hauke Busch

Development of a general Software Application for the Quantification and Mitigation of Batch Effects in Microbiome Data

Dissertation
for Fulfillment of
Requirements
for the Doctoral Degree
of the University of Lübeck
from the Department of Natural Sciences

Submitted by
Michael Olbrich
from Berlin

Lübeck 2022

First referee: Prof. Dr. Hauke Busch

Second referee: Prof. Dr. Christian Sina

Date of oral examination: 10.11.2022

Approved for printing. Lübeck, 15.11.2022

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne die Benutzung anderer als der angegebenen Hilfsmittel selbständig verfasst habe; die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe des Literaturzitats gekennzeichnet.

Lübeck, den 17. Mai 2022

Abstract

The emergence of unwanted variation resulting from batch processing is a well-researched challenge in next-generation sequencing data. While appropriate measures during the planning and execution of an experiment can limit the emergence and magnitude of batch effects, they are not entirely preventable and thus need to be accounted for before statistical analyses. Despite the availability of batch effect correcting algorithms (BECA), no comprehensive tool that combines batch correction and evaluation of the results exists for microbiome datasets. This work outlines the Microbiome Batch Effects Correction Suite (MBECS) development that integrates several BECAs and evaluation metrics into a software package for the statistical computation framework R.

Kurzzusammenfassung

Das Auftreten unerwünschter Variationen, die sich aus der Stapelverarbeitung ergeben, ist eine gut erforschte Herausforderung bei Sequenzierungsdaten der nächsten Generation. Geeignete Maßnahmen während der Planung und Durchführung eines Experiments können zwar das Auftreten und Ausmaß von Batch-Effekten begrenzen, sind jedoch nicht vollständig vermeidbar und müssen daher vor statistischen Analysen berücksichtigt werden. Trotz der Verfügbarkeit von Batch-Effekt-Korrekturalgorithmen (BECA) existiert für Mikrobiom-Datensätze kein umfassendes Tool, das Batch-Korrektur und Auswertung der Ergebnisse kombiniert. Diese Arbeit skizziert die Entwicklung der Microbiome Batch Effects Correction Suite (MBECS), die mehrere BECAs und Bewertungsmetriken in ein Softwarepaket für das statistische Berechnungsframework R integriert.

Contents

1 Introduction

The term microbiome (MB) was used as early as 1952 by J.L. Mohr to refer to the microorganisms found in a particular environment [101]. The most current definition was developed and published by a panel of experts in 2020 and described the microbiome as a distinctive community of microbial species with innate physiochemical features that inhabit a specific environment [84, 149].

Thus the term microbiome refers to the genome of all the microorganisms that occupy a defined biological habitat. The interplay of microbial communities, including bacteriome, mycobiome, and virome, with their host in symbiotic and pathogenic relationships is significant for comprehending living organisms. Characterization of microbial genomes opened up a new branch of research, making it feasible to characterize microbial species at the functional level and generate snapshots of microbial communities from different environments, e.g., from the gut or the skin microbiome.

As with all signal capturing processes, it is necessary to account for technical variability. This applies to the inherent technical noise of the process, which is commonly modeled with a normally distributed error term and variation that gets introduced through processing in different batches. The batch effect (BE) denotes a class of technical artifacts that emerge from collecting and processing samples in distinct groups.

In 2014 Yue et al. published the results of a comparative analysis of gene expression between human and murine tissues. They reported significant differences in gene expression according to organism rather than tissue type, which would challenge the applicability of murine models as a substitute for human research. An effort to re-analyze the data by Gilad et al. uncovered that the reported significant biological differences were the consequence of batch effects that resulted from processing the samples in batches that reflected the host organisms [61]. This example is a cautionary tale of the significant detrimental ramifications of disregarding potential batch effects.

1.1 Motivation

Bigger cohorts improve statistical power and facilitate the discovery of comparatively small biological effects. However, next-generation sequencing machines are limited by the capacity of samples that can be processed in a single run. Hence, large sample sizes usually result in a batch effect because the samples need to be processed in multiple subsequent runs or on different machines simultaneously. As both the availability of datasets and the cohort sizes grow in correspondence to technological development and cost reduction, the correction of Batch Effects (BE) or, at a minimum, accounting for them in statistical analyses becomes imperative.

In next-generation sequencing applications such as whole-genome sequencing (WGS) or whole-exome sequencing (WES), the development of batch effect correcting algorithms (BECAs) has been a field of research for more than 20 years. Consequently, a wide variety of approaches has been developed over the years, the most prominent of which are 'removeBatchEffects' and the 'ComBat' algorithm incorporated in the limma and sva software packages, respectively [86, 127]. As is the general case in data analysis, the data characteristics justify the choice of processing method to some degree, i.e., non-normal distributed data can be transformed via log transformation to satisfy the requirements of a statistical tool. Similarly, these characteristics modulate the results of batch correction procedures. The compositional nature of microbiome datasets limits the applicability and effectiveness of methods due to the sparsity in the counts. As batch effect correcting algorithms (BECA), for the most part, have been designed for micro-array experiments, their usability for compositional data can be limited [92]. Although technically, all NGS data is compositional, the characteristics are more prevalent in MB data. Other factors like study design, gathered covariate data, or library sizes generate further restrictions and opportunities for BECAs. This results in a wide variety of methods to choose from, requirements and characteristics to consider, and the need to assess the effectiveness of a method concerning the reduction of BE and preservation of the biological effect. The motivation of this work is thus to provide the reader with an in-depth understanding of BE generation and how to limit their emergence during study design and execution and a toolkit, in the form of an R-package, that facilitates the mitigation of BE.

1.2 Scientific Question

The compositional nature of microbiome datasets limits the applicability and effectiveness of methods due to the sparsity in the counts. As batch effect correcting algorithms (BECA), for the most part, have been designed for micro-array experiments, their usability for compositional data can be limited [92], i.e., optimization procedures might not converge on sparse data matrices or exhibit reduced effectiveness. Therefore, the research objective is twofold: first, to investigate and understand existing BECAs, and second, to find appropriate metrics to measure each method's improvement on particular datasets.

1.3 Previous Works

BE mitigation is an established part of bio-informatic data analysis, and a lot of research effort went into developing and implementing computational methods. With the growing accessibility and importance of layers of OMICS, e.g., microbiomics, proteomics, mycobiomics, or metabolomics, the research spread from the classic transcriptome and genome datasets to this new field of investigation as well. Algorithms such as Percentile Normalization [59] have been explicitly developed for microbiome case/control studies. In addition, several software packages and applications implement BEC workflows or incorporate a correction step in their processing pipelines. One example is the BatchQC software that offers an interactive interface for BEC in genomic data [94]. In addition, workflow guides such as [83] provide a valuable resource to learn about BEs, correction methods, and a step-by-step approach to correction and evaluation methods.

There is no comprehensive software tool concerning BE mitigation to the author's knowledge.

1.4 Thesis Outline

This work aims to provide an in-depth understanding of batch effect formation and consequences for analysis in microbiome data sets. To that end, the subsequent Chapter 2 will present basic information on biological and technical concepts. This comprises a discussion of biological and technical aspects of the microbiome followed

by a short overview of developments in NGS technology and their inherent characteristics and an outline of the required software processing steps in MB data analyses. The chapter will close with a comprehensive description of batch effects. The consecutive Chapter 3 is concerned with the used materials and methods chapter will detail the datasets used for the development of the MBECS package, introduce the employed software and explain the respective BECAs and delve into the metrics used for the assessment of BEs. Chapter 4 presents the developmental process and the features of the microbiome batch effect correction suite. It can be considered an extensive user manual. Finally, Chapter 5 will summarize the efforts made and deliberate possible challenges and future improvements.

2 Background on Biology and Technology

In 1665, Robert Hooke improved upon the design of the microscope and was the first to observe cells that are otherwise invisible to the naked eye. He examined a piece of cork and found a porous structure that reminded him of 'cells' in a monastery [70]. This achievement represents an enormous gain in the comprehension of life. It would not have been possible without the progress in the field of optics that ultimately led to the development of the microscope. Similarly, the accumulation of knowledge led to the formulation of the theory of natural selection. Although in one form or another, the concept of heredity was applied throughout human history, e.g., selective breeding was used long before Hooke observed the first cells. Almost 200 years later, Charles Darwin raised the concept of evolution to the state of a theory with his book 'On the Origin of Species' [38]. A significant drawback of Darwinism was the lack of an explanation for the fundamental biologic mechanism that governs heredity. His experiments on inheritance by Mendel did not gain broad attention when he published his findings in the book "Experiments on Plant Hybridization" in 1865 [1]. Mendel bred pea plants, observed and documented inheritance patterns, and was able to formulate mathematical rules that describe the behavior of the observed traits. His results implied that heredity is not acquired but a particulate mechanism in which the parent generation passes on its traits to the children. Deoxyribonucleic Acid, a large molecule in the nucleus of cells, was discovered by Friedrich Miescher in 1869 [35]. Albrecht Kossel managed to isolate the non-protein component of DNA, the nucleic acid, in 1878. Chromosomes were identified as central to the hereditary process in 1880 by Walter Sutton and Theodor Boveri [13]. It was not known yet that DNA carries the genetic information in the form of discrete units, i.e., genes, but this represents the advent of cytology, an effort to link cell theory and genetics together. With the rediscovery of Mendel's work in 1900, scientists focused on establishing the biological process governing inheritance, making him the parent of modern genetics. In 1928 Frederick Griffith published a report that detailed his dis-

covery of a *transforming principle* in what is now known as *Griffith's Experiment*. He observed the transformation of a harmless bacterial strain into a pathogenic one by incorporating cellular material of a heat-killed deadly strain into themselves [65]. This observation was the first piece of knowledge that implied DNA as the carrier for genetic information. Although, by now, building blocks of the DNA molecule had been identified, i.e., the nucleic acids, it was not obvious which structure the molecule had or which, if either, part was responsible for carrying the genetic information. Further developments in scientific methods and optical technology led to the capture of famous photo 51, an x-ray diffraction image of highly hydrated DNA [42, 160]. It turned out to be essential for the efforts of Francis Crick and James Watson in establishing the first correct structural model of the DNA molecule in 1953 [148]. The molecule comprises the nitrogenous nucleobases Adenine, Thymine, Guanine, and Cytosine, a monosaccharide called desoxyribose, and a phosphate group. In an alternating fashion, the phosphate group forms the literal backbone of the polynucleotide chains of the molecule together with the desoxyribose.

Deoxyribonucleic acid is a molecule that exists in every cell. In its natural form, DNA has a double helix structure that was put into the focus of the public by Odile Crick's famous drawing after its discovery in 1952 [111]. In eukaryotic organisms, the genetic information separates into chromosomes; these long strands of DNA coil around so-called histones and are encapsulated into the cell's core to preserve its integrity. When James Watson and Francis Crick published their findings on the structure of DNA in an issue of the journal Nature in 1953, they already acknowledged a possible mechanism for replication.

> It has not escaped our notice that the specific pairing we have postulated immediately suggests a possible copying mechanism for the genetic material [137].

They were referring to the double helix structure of two strains of nucleobases that form covalent bonds, i.e., Adenine can only bind to Thymine and Guanine binds exclusively to Cytosine, thus joining both sides of the double helix. Because the binding partner of any base is restricted to a single option, both separate strains of DNA contain the same information in reverse complement order.

This restriction means that any of the two half-strains can be used to reconstitute a complete DNA molecule by incrementally attaching complementary deoxynucleoside

triphosphate (dNTP) molecules [144]. For cell division, all the chromosomes undergo this process to duplicate the set of genetic information, one set for each cell created. This process of DNA synthesis is called replication and forms the basis for cell division during growth, aging, and the repair of damaged tissues [24].

The sequence of nucleobases on a strand of DNA carries the actual genetic code. Functional units called genes contain the information to generate gene products through gene expression, i.e., the translation of the genetic code into operative products such as proteins. The set of genetic instructions is called the genome. In human beings, the genome consists of the host DNA packaged in 23 chromosome pairs in the center of cell nuclei and an additional chromosome that belongs to cells' mitochondria. Human DNA consists of 3 billion base pairs and, based on current knowledge, less than 2% contain information for generating gene products. To that end, three successive bases form a codon which results in 64 distinct combinations. Of the possible combinations, 60 refer to the 20 standard amino acids (AA) that construct proteins in humans. A complex interplay of the signal encodes the 21st AA during translation. A single codon encodes the starting point of transcription, and the remaining three codons indicate the stopping position for an ongoing transcription. The sequence of base pairs, or codons, determines the sequence of amino acids constructed by a protein [131]. The intricacies of the relationship between AA sequence, three-dimensional structure, and protein function are out of the scope of this work.

An intermediary between DNA and protein synthesis is ribonucleic acid; it is similar to DNA in that it has an alternating sugar-phosphate backbone with bases attached to it. Other than DNA, it consists of a single strand instead of two; the desoxyribose sugar in the backbone is substituted by ribose, and the nucleobase thymine is replaced by uracil. A particular form, the messenger RNA (mRNA), carries the information to the translational apparatus. The term denotes the whole of molecules associated with biological protein synthesis. Part of the translational apparatus is the ribosome, a macromolecular machine that forms polypeptide chains by linking amino acids in the sequence given by a fragment of mRNA. The ribosome is comprised of a small and a large ribosomal subunit. Both subunits are made up of one (large) or multiple (small) ribosomal RNA (rRNA) molecules and a variety of ribosomal proteins (r-protein). Conceptually, the ribosome is a macromolecular machine that synthesizes proteins from protein-coding sequences called genes. Transfer

RNA (tRNA) is the third primary type and links RNA transcription, and protein synthesis [119].

To that end, tRNA comprises an anticodon, i.e., a sequence of three nucleobases that complement a codon that specifies a particular amino acid and an adapter sequence that can bind to an amino acid. After transcription, the mRNA essentially presents a series of codons that determine the order in that AAs need to be linked. Every molecule of tRNA is specific to one type of amino acid. It can attach to the codon that matches their anticodon, which, piece by piece, synthesizes a protein molecule. There are many more different types of RNA, all with varying domains and or functions, e.g., small nuclear RNA (snRNA), small nucleolar RNA (snoRNA), microRNAs (miRNAs), or long non-coding RNA (lncRNA). The synthesis of RNA molecules from DNA is called transcription. The mRNA serves as the template for protein synthesis as the amino acids are aligned in the sequence of the codons of RNA fragments. Only part of all genes is translated into proteins, in any case. For one, part of a gene serves as an instruction set that modulates the translation, e.g., start and stop regions that indicate the beginning and end of the sequences to be translated. Alternatively, transcription factors that mediate the number of gene products produced [4].

This abridged description of the evolution of biology into molecular biology introduced basic concepts and terms that will be fundamental in subsequent sections. It also illuminates a common theme in scientific progress, the relationship between technological progress and fellow scientists' achievements. Advancements in optical technology facilitated the discovery of the cell. For example, the experiments on the inheritance of a monk inspired a whole new generation of scientists more than 30 years after their completion. Furthermore, the inception of X-ray crystallography [8] allowed for determining the actual structure of DNA. The collaborative effort of various technological and scientific domains facilitates progress and increases the resolution of observations. For example, biology developed sub-domains, e.g., microbiology or molecular biology, that opened up whole new fields of investigation and radically transformed the understanding of life.

This chapter will lay out the foundation for the reader by introducing and explaining concepts and ideas concerning the main fields of biology, technology, and computation. Section 2.1 introduces ecological microbiology as field of interest. It will establish heritage, level of organization, and the essential vocabulary. The subse-

quent section outlines the technological aspects of omics data, i.e., the development and characteristics of next-generation sequencing technologies (NGS) in ecological microbiology. Finally, these aspects will be conceptually integrated into the outline of computational procedures required to analyze microbial data.

2.1 Microbiome

The microbiome is defined as a group of microscopic organisms that populate and interact within a specified habitat, creating a microscopic ecosystem [84]. Because microbes are too small to be observed by the naked eye, it required the development of the microscope to catalyze the scientific study of these organisms. Microbiome research developed from microbial biology due to advances in understanding the significance of microscopic organisms.

2.1.1 Microbial biology

Micro-organism encompasses unicellular and multicellular entities in all three domains of life. The domain of eukaryotes contains larger-scale multicellular organisms such as mammals. Bacteria, and Archaea, consist exclusively of micro-organisms. While this definition is relatively new, the idea of minuscule organisms can be traced back to the fifth century BC, when the so-called nigodas were first mentioned in a religious text. Still lacking the technology to observe them, the microbes were implied in spreading disease by the Roman scholar Marcus Terentius Varro in the first century BC [122]. Throughout the centuries, many more speculations were made about tiny, living, disease-causing, and disseminating particles. Finally, in the 17th century, in parallel to the previously outlined evolution of molecular biology, the discovery of lenses and the resulting microscope development also facilitated the emergence of microbial biology.

With a passion for scientific investigation, the Dutch merchant Antonie van Leeuwenhoek developed an interest in lens grinding and started designing and constructing his microscopes. His inventions allowed him to observe the microscopic organisms inhabiting mud, water, and dental plaque samples. A series of letters to the British Royal Society meticulously details his study of the animalcules. Today, van Leeuwenhoek is universally accepted as the father of microbiology for discovering bacteria

[45]. The flowering structures of molds were observed under a microscope by Robert Hooke in 1665 and described as microscopical mushrooms.

Like any other scientific endeavor, biology requires a standardized system that facilitates the stringent description and classification of entities based on shared characteristics and traits to ensure stringent and repeatable experimentation and recording. For example, the Swedish botanist Carl Linnaeus developed the binomial nomenclature in conjunction with the Linnaean taxonomy to classify and name living organisms [21].

A taxonomy is a set of hierarchical taxonomic ranks organized as nested categories with increasing specificity. The categorization begins with selecting an appropriate domain or super-kingdom, i.e., one of the three domains of the tree of life: Archaea, Bacteria, and Eukarya, followed by the more specific kingdom. For all intents and purposes in this work, the three previously introduced micro-organisms categories will be considered kingdoms, i.e., bacteria, fungi, and viruses can be placed in their respective kingdoms. Subsequent taxonomic categories are phylum, class, order, and family; they allow more specific agglomeration and distinction of entities in a specific kingdom. The final levels are genus and species used for naming in the Linnaean binomial nomenclature. This naming scheme uses predominantly Latin grammar and is composed of two parts. The first part identifies the genus, and the second denotes the particular species. In its first iteration, the Linnaean system only discriminated between plant and animal kingdoms. The phylogeny, i.e., the relationship between different taxa, was inferred based on observable taxonomic attributes that served as evidence. At the time of inception, no scientifically accepted theory of heredity or evolution existed. Hence, these attributes were limited to morphology, behavior, geography, and ecology. Moreover, the concept of grouping organisms by traits and nesting them within higher-order categorization began to produce patterns resembling the modern dendrograms of evolutionary taxonomy [97].

With scientific progress such as the publication and wide acceptance of an evolutionary theory in 'On the Origin of Species' [38] it became necessary to specialize these categories and expand the set of utilized characteristics. While, at the time, the development of culturing and cloning approaches had been a huge step forward, there remained significant limitations. They are highly labor-intensive and not remarkably accurate identification methods [80, 82]. Technological progress in molec-

ular biology facilitated further refinement due to the newfound access to metabolic processes, immunologic setup, or DNA and RNA sequences. The discovery of the 16S rRNA gene by Carl Woese in 1977 enabled a more straightforward classification of bacterial species and phylogenetic relatedness [52].

Ribosomal ribonucleic acid (rRNA), a form of non-coding RNA, i.e., it will not be translated into a protein that is essential in protein synthesis [30]. The ribosome is present in all organisms and usually organized into a small (SSU) and a large (LSU) ribosomal subunit. In bacterial genomes, these subunits are made up of a single rRNA molecule of roughly 1500 nucleotides in the SSU, the 16S rRNA, and two molecules for the LSU, the 5S rRNA, and the 23S rRNA. Because the ribosome has to fulfill the same tasks, a part of it is highly conserved, i.e., precisely the same regardless of the bacterial species. Figure 2.1 shows a schematic depiction of the different regions in the 16s rRNA gene. The conserved regions shown in blue are intermingled with variable regions depicted in light blue. These variable regions allow for phylogenetic assessments and thus facilitate taxonomic classification.

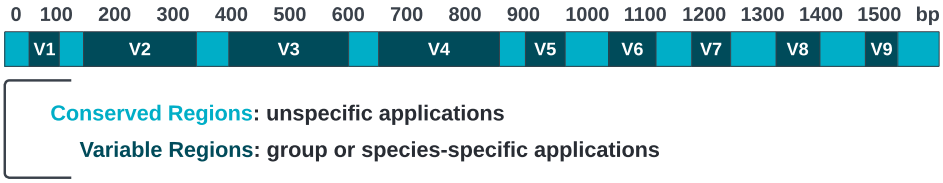


Figure 2.1: Small sub-unit of the 16S rRNA gene showing intermixed conserved and variable regions. The conserved regions can be targeted with primer sequences in order to extract and amplify the variable regions they are bordering. The variable regions allow for taxonomic classification up to species level.

The 16S rRNA gene is used when sequencing bacteria and archaea [72]. Fungal micro-organisms possess a region with similar characteristics, the internal transcribed spacer (ITS) region located between the large and small subunits of ribosomal RNA genes [136]. They also occur in bacterial species but are generally not used for identification. As with the 16S region of bacterial species, the ITS regions serve as phylogenetic DNA markers that allow the classification of fungal taxonomy. These approaches facilitate the identification and characterization of low abundant

and non-cultivable microbial organisms among hundreds of organisms within a single analysis, enabling the faster and more accurate study of complex microbiomes than traditional cloning and culturing approaches. Sample multiplexing allows the processing of multiple samples in a single run, and the semi quantification of abundances enables insights into microbial ecology. Although the processing steps differ slightly between 16S and ITS sequencing approaches, they are similar enough for this work to focus on 16S rRNA sequencing.

The ongoing effort to improve the classification system is perhaps most apparent in fungal taxonomy, as DNA-based investigations tend to overturn classifications based on purely morphological features. However, genetic features conveniently integrate into the existing classification scheme, and the Linnaean approach has proved to be such an elegant solution to the categorization of organisms that it has become well established in science [116]. Microbiology has developed into a large field. By now, it could already be separated by the taxonomic kingdom of the investigated organisms, effectively creating the fields of bacteriology, mycology (fungi), or virology. However, historical development has shown that there will always be an overlap between this categorization and other biological and technical sciences.

2.1.2 Applied Microbiology

Thus far, the main driving factors were medical research and food processing to a lesser degree. Furthermore, microbes were considered singular, unsocial agents with detrimental or beneficial effects in these contexts.

It took until the advent of the 19th century for the new scientific field to pick up the pace. At first, the inquiries focused on single organisms' structure and working principles in a medical and food-related context. Historically, human populations have used microbial organisms long before the advent of microbiology. Fermentation, for example, denotes a biochemical process in which enzymes facilitate a chemical change of organic substrates in an anaerobic environment. Microbes use this naturally occurring procedure to produce energy in the form of adenosine triphosphate (ATP). In food production, this characteristic is used to produce alcohol out of sugars in the manufacturing of wine or beer. Other examples of fermented foods include kimchi, sauerkraut, or yogurt [95]. By linking yeasts to the process of fermentation [5], Louis Pasteur founded the science of studying the mechanics and

applications of fermentation called zymology. The heat treatment used in the food industry to preserve food is named after him. Pasteurization is a thermal processing technique for food in which undesirable micro-organisms and their secreted enzymes are deactivated to prevent or delay spoilage.

In a medical context, the prevailing concept was that of pathogenicity, i.e., specific organisms are responsible for causing and spreading disease [23]. Medical microbiology led to identifying several disease-causing microbial organisms, treatment options, and revolutionary hygiene concepts.

Martinus Beijerinck investigated a contagion of tobacco plants that causes discoloration of the leaves in a mosaic-like pattern. While Dimitri Ivanovsky had attributed the failure to isolate the pathogen to faulty equipment, Beijerinck suggested that it was merely too small to be captured by filtration methods commonly used to capture bacterial species. He called it a contagious living fluid (Contagium vivum fluidum) due to its seemingly liquid properties and is considered the discoverer of viruses [29].

By 1884 Robert Koch had gathered tremendous insights into the development and spread of infectious diseases and is credited as the primary contributor to germ theory and thus the field of medical microbiology. His so-called Koch postulates are a set of criteria based on the principle that a single micro-organism can cause and disseminate a particular disease [146]. In addition, Koch was one of the first scientists to grow bacterial species in a medium of nutrients [81].

Anthrax is a bacterial infection that can occur on the skin, in the lungs, in the intestinal tract, or in the bloodstream through an injection. The symptoms are severe and can lead to the patient's death. Robert Koch was the first to identify the bacterium *Bacillus anthracis* as the causal organism, and Louis Pasteur managed to develop an effective vaccine that enabled the immunization and thus prevented contracting the disease [17, 145]. However, in case of an existing infection, the disease can be treated with antibacterial agents called antibiotics. Incidentally, the development of antibiotic treatments was a welcome side effect of bacterial research. First, Paul Ehrlich identified the synthetic arsenic-derived compound salvarsan that would bind to and destroy exclusively bacterial micro-organisms [57]. Natural remedies of this type were recorded in ancient times as well. For instance, the Greeks documented that the topic application of moldy bread can have beneficial effects on the healing progress of wounds [28]. However, it is unlikely that the responsible

organisms or exact mechanics were understood at the time. A lucky accident led Sir Alexander Flemming to discover the antibacterial properties of the green mold *Penicillium chrysogenum* [14, 124]. He observed how spores of the mold prevented disease-causing bacteria in one of his culture plates from growing and postulated that it secreted an organic antibacterial compound called penicillin. In the field of virology, Edward Jenner managed to immunize a child against smallpox by using inoculations of the cowpox virus [47]. This practice allowed Louis Pasteur to develop a rabies vaccine [33] although, at the time, the knowledge of the structure and operating principles of viruses was still narrow.

Considering the multitude of beneficial developments, it stands to reason that the concept of singular microbial agents was not wrong but merely part of the picture. For example, the conception of chemolithotrophy by Sergei Winogradsky significantly broadened the common understanding of the role of microbes. Chemolithotrophy denotes a metabolic process of micro-organisms in which inorganic chemicals are oxidized to produce energy, which is essential for soil formation of biochemical cycling. This finding raised the question of how microbes react to their environment, and the relationship between structure and function, i.e., microbial physiology, became an increasingly important research subject [43].

2.1.3 Microbial Ecology

With the accumulation of knowledge, it became apparent that micro-organisms existed in every conceivable environment. The understanding shifted from singular unconnected entities to interconnected networks of various microbial organisms [15]. This shift marks the advent of Microbial Ecology (ME), a field of study concerned with the composition of microbial communities and the interactions with living and non-living factors in the environments they inhabit [74].

The focus of research broadened from health and food-related topics. For example, soil and water samples showed an abundance of different microbial species that form symbiotic communities. Interestingly, the organisms did not only exist in antagonistic relationships but also engaged in commensalism, i.e., the end-products of a species' metabolic process are nutrition for another group of micro-organisms [15]. So far, the research was generally concerned with environmental samples, e.g., water from lakes, rivers, swamps, or soil samples. The investigations pushed the

development of culturing techniques and research approaches as usual. However, it turned out that micro-organisms also exist in symbiotic relationships with larger host organisms [77]. For example, the term microbiome (MB) was first used to describe the group of microbes found in any specified environment. Over the years, it has also changed to encompass their metabolic processes and interactions. The most current definition has been developed and published by a panel of experts in 2020 and describes the microbiome as a distinctive community of microbial species with innate physio-chemical features that inhabit a specific environment [84]. The composition of communities in terms of present and absent species is dynamic regarding the environmental conditions of the habitat. As a demarcation, the term microbiota encompasses all living organisms that form the microbiome, which in itself is a difficult distinction to make. It is commonly agreed that bacteria, archaea, fungi, and small protists are to be considered part of the microbiome [118]. The placement is more difficult with algae considering that this informal group contains unicellular and large multicellular organisms. Another controversial topic is the integration of viruses, plasmids, or other free genetic elements. Considering that bacteriophages are essentially an environmental factor that modulates the population of bacterial species, it seems reasonable to include viruses in microbiome studies [89].

The advent of Omics technologies has been essential in understanding the significance of microbial organisms to life in general [50]. Genomic technologies allow for an assembly of whole genomes, which consecutively enables the investigation of familial relationships on a genetic level. The identification of single genes gives insight into cellular processes. Transcriptomics makes it possible to examine the actual composition of gene products in a sample and infer gene function [39]. Moreover, proteomics enables the study of metabolic processes through the expression of proteins. Metabolomics allows the investigation of even smaller products in the metabolome [41, 126]. Culturomics approaches' ability to culture and identify various strains and species provides the knowledge necessary to distinguish bacterial species in high throughput experiments [79].

Genome assemblies and investigation of metabolic features also allow for better taxonomic classification and thus the distinction of species. The distinction between entities became more specific as scientists could rely on similarities in genetic structure and metabolic pathways alike, which transformed the Linnaean system into a reflection of evolutionary relationships among organisms [116].

It was discovered that humans come into contact with a wide variety of microorganisms on a daily basis, and most of them are not disease-causing. In fact, bodies are host to a wide variety of organisms that inhabit the different habitats our body offers, e.g., the skin, as the largest organ, offers dry, sebaceous, and moist sites that facilitate the growth of distinct microbial communities due to their characteristics. For example, the mucosal microbiome is distinct in composition compared to the microbial communities that inhabit a dry area like the skin on the elbow.

Moreover, the microbiome exists in symbiosis with the inhabited organism. For example, the composition of bacterial, fungal, and viral species occupying the gut help with the digestion of food and is modulated by the nutritional content [18, 107]. High sugar intake selects for different species than a high fiber diet [34].

The convolution of humans with their microbial inhabitants is so close that a highly complex nerve cluster evolved in the gut, the vagus nerve, which coordinates the digestion in collaboration with the microbial communities and communicates directly with the brain [53]. The inoculation with microbial communities during vaginal birth is of such high importance to the development of the immune system that the differences due to caesarian section can be tracked over a lifetime [125].

Microbial communities that inhabit the skin enhance its barrier function and prevent pathogenic organisms from colonizing the skin [20].

Concerning abundance, human cells actually make up less than half the cells in the human body; the rest are trillions of bacteria, fungi, and viruses that are spread throughout all tissues. Understanding this relationship gave rise to the holobiont concept, i.e., the understanding that an organism such as a human being or a tree is not only the composition of its inherent genetic composition but also encompasses all the species that live within or around it. The holobiont can be considered in terms of the meta-genome that includes all the genes that are part of this discrete ecological unit [69, 143].

2.2 Data Generation

The ability to capture the base sequences in Deoxyribonucleic Acid (DNA) significantly impacted comprehending biological processes that govern life. The advent of sequencing technologies unleashed a large-scale multidisciplinary effort to decode the human genome with the promise of understanding the code of life. The Human

Genome Project (HGP) was an international research project that set out to map all the base pairs that make up human DNA and ultimately identify and characterize the encoded genes in a functional context [60]. This effort took 13 years to complete and cost about 4.6 billion dollars (value adjusted for 2022) [133].

While the project fell short - in some sense - of its initial goal to attain a complete understanding of the processes that govern life, it yielded the first-ever produced human genome, a reference to compare by, and the first available genotype. The mutual effort spearheaded the development of faster, more reliable, and cheaper methods for sequencing [135]. To understand the significance of technological development, the reader can imagine the hardships of the first sequencing technologies. For example, the wandering spot analysis of Gilbert and Maxam is a painstaking and error-prone first method of sequencing that resulted in the mapping of a 24 base pair sequence [68]. A project on the human genome scale would not have been feasible without the development of Sanger sequencing [132]. The advances also furthered the development of computational approaches in genetics by necessity. The amount of base pairs in the human genome makes analysis without the computational prowess of microprocessors infeasible. Just storing the 3billion base pairs of the genome is challenging in any material form, not to mention actual analyses. Therefore, the processing that is required to produce reliable results comprises error correction methods as well.

The advent of next-generation sequencing technology opened up a new dimension in the investigation and understanding of microbial communities. The high-throughput nature of these new technologies also reinforced the emphasis on stringent experimental procedures. Although the work required to process a single sample had been reduced significantly, the larger number of samples that could be processed now introduced new challenges. This section is concerned with the steps that precede the analysis of microbial communities. The subsequent paragraphs will outline the main aspects that lead to the generation of quantifiable data, from the study's design to the application of modern sequencing technology.

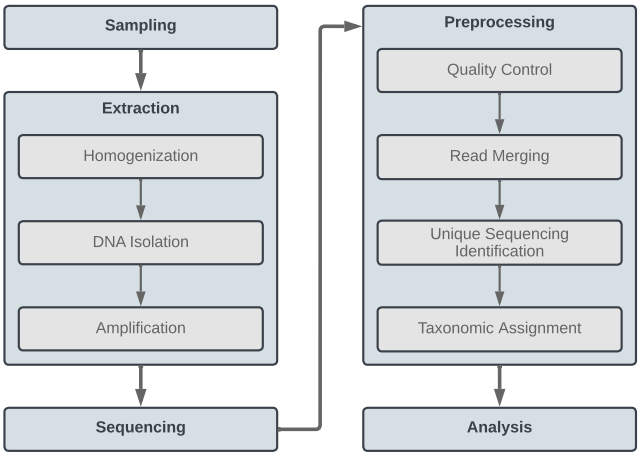


Figure 2.2: The diagram shows the processing steps in microbiome research. After collection, the samples are processed in the laboratory to extract the DNA sequences and prepare for the sequencing procedure. Before the statistical analyses, the resulting raw-data files need to undergo preprocessing for quality control and taxonomic assignment.

2.2.1 Sampling

In a research context, the term sampling refers to the selection of a sample from an individual as well as the collection of multiple samples that are representative of a whole population that is of interest to the particular research effort [16]. Hence the term sample is utilized ambiguously and can either denote a single specimen or a sample of a larger population comprised of many specimens depending on the context.

Sampling a subset of a population makes it feasible in terms of time and money expenditure to perform research assuming that the procedure avoids the introduction of biases and the sample remains representative of the underlying population. Thus it is essential to specify the population that is supposed to be sampled in terms of criteria relevant to the context of the research, e.g., sex, age, nutrition, lifestyle choices, pets, alcohol consumption, and many more. An extreme example of such a sampling bias would be a case-control study in which the diseased group comprises male individuals, and the healthy group is made up predominantly of female subjects.

A variety of strategies that address this issue are available and have been summarized by Elfil et al., or Bhardwaj et al. [16, 46] among others.

The aspects of sampling that relate to this work are of a technical nature. For example, the quality of a particular sample is affected by the environmental conditions at the time of extraction, the used protocol, and the experience of the implementing personnel. The amount of time that passes between extraction and further processing steps or storage of samples will also influence the quality of samples [134].

2.2.2 Extraction

The collected samples need to be processed by a lab technician to prepare the targeted sequencing of variable regions within the 16S ribosomal RNA gene.

The first step in sample processing is homogenization. Just like the homogenization process in food processing, e.g., killing micro-organisms in milk to prevent it from spoiling, one goal of these steps is to kill everything in the sample and stop it from changing from the current state [40]. Furthermore, the particle size of the sample will be reduced and thoroughly mixed so that the content is equally distributed within the sample [2]. This ensures that there will be no heterogeneity if the sample is used for multiple types of analyses. Finally, in the context of molecular biology, homogenization aims to lyse cells, i.e., break down the cell walls and membranes to make the intracellular content accessible to the subsequent processing steps. After this step, samples can be frozen in -80℃ freezer to prevent degradation until further processing [51].

The DNA fragments need to be separated from the fecal content, and other cellular debris that is of no use to the analyses [49, 138]. To that end, the laboratory uses standardized protocols and commercial kits for DNA extraction from microbial samples. An extraction kit comprises several reagents used in a workflow of specific processing steps and yields a cleaned-up solution of a samples' DNA content. There are a wide variety of these extraction kits available, and laboratories might also implement their own protocols to achieve this task.

Adapter sequences specifically designed to attach to the conserved parts of the 16S gene, so-called primers, are added to the purified solution. Primers can be designed and manufactured as part of the local laboratory effort, but they are also available for purchase from commercial manufacturers. These primers are combined with

additional barcode sequences specific to each processed sample and enable the unique identification of each sample. The subsequent step multiplies the sequences that the primers have tagged to create enough genetic material for the sequencing step. This step is called amplification and employs polymerase chain reaction (PCR)[108]. A single PCR cycle duplicates every sequence within the solution tagged with a primer, and after multiple runs, there is enough material to continue with the sequencing procedure. Finally, gel quantification can be used to determine whether or not enough material has been produced [120, 138].

2.2.3 Sequencing

The previously outlined method of sample preparation refers to NGS amplicon sequencing. It remains to acknowledge the other two standard methods for microbiome sequencing for completeness. For one, there is Sanger sequencing which offers precise and accurate sequencing. Due to the labor-intensive process, it is only feasible for a small number of samples. It is a targeted sequencing method that employs the chain termination method to determine sequences. That makes it highly sensitive and comparatively simple to use but also laborious. Hence it is mainly used to identify and confirm variants found with other less reliable methods. Another approach to microbiome research is metagenome sequencing. This is an untargeted NGS approach in which all the genetic material of a sample is sequenced. Subsequent processing steps require separation from host sequences and reassembly of genomes from the massive amount of gene fragments. In principle, this approach provides a better overview of communal structures because it is not limited to a particular type of organism, i.e., bacterial, fungal, or viral, but it will capture everything. Naturally, the generated amount of data is significantly larger than other methods, and thus the processing effort is much greater. Moreover, it might be less effective in detecting rare species and has a questionable impact on final analyses for low abundance rare genera.

The different available technologies of NGS amplicon sequencing, how they work, and which advantages and disadvantages they have are out of the scope of this work. The goal is to provide the reader with a general understanding of the technical principles behind the technology.

Frederick Sanger first modified the mechanism facilitating DNA replication during mitosis to enable sequencing [132]. The challenge in the replication process was to determine which bases were attached to the template strand because the process was too fast for any measurement method to produce accurate results. Sanger resolved this issue by modifying the dNTPs to terminate the synthesis process. These so-called dideoxynucleotide triphosphates (ddNTPs) would attach to a sequence but not allow the attachment of further bases, which effectively stops the replication process.

Assuming that a sequence of interest has already been isolated and amplified, i.e., replicated many times by employing polymerase chain reaction (PCR), and the strands have been broken apart. By adding a primer sequence to the mix, it is possible to select the desired strand and enable the replication process [108]. Now create a solution of free-floating dNTPs and about 1% of ddNTPs and let the replication process commence. Next, filter out the sequences that end in a ddNTP by using gel electrophoresis and measure the weight of these fragments. The weight translates into the length. If this is done multiple times with four solutions containing each type of ddNTP respectively, it is possible to reassemble the template strand sequence.

This measuring method is highly accurate, so much in fact that it will be used today to confirm sequences that have been identified with high throughput approaches. Nevertheless, it is very time-consuming as it only allows to process approximately 3,000 bases per week, which would mean roughly 21K years to sequence the entire human genome for a single research unit. Another limiting approach is the length of sequences that can be processed. Because the method relies on measuring the weight of the molecule in order to infer its length, it becomes unrelieved, i.e., produces too many errors after a certain length. Measuring the weight difference between 1000 and 1001 bases on a molecular level is affected more by the tolerance of measuring tools than the actual difference in weight.

The subsequently developed Next Generation Sequencing methods modified Sangers' approach and thus achieved a much larger throughput [115]. The ddNTPs are now tagged with fluorescent molecules that can emit light according to their type when excited. Hence, after each synthesis cycle, the molecules are stimulated, and the machine will take a picture that can be analyzed to identify the newly attached bases [108]. Secondly, the termination process is reversible, which means that the

attached sequences can be measured in one cycle, and then in the next, the replication process can continue to attach a new base. And third, the process can run a massive amount of template sequences in parallel, which reduces both time and cost of the sequencing process. These approaches can also process sequences from many different samples by attaching primers with barcodes to identify and assign sequences to their respective origin samples.

2.2.4 Sequencing Output

A sequencing run will generate a record of all measured sequences in a data file, usually in FASTQ format. The term FASTQ denotes a text-based file format used to store nucleotide sequences and the quality scores that correspond to each base call. It is the format in which raw data from 16S microbiome sequencing is commonly delivered. Each sequence is represented with four lines; the sequence header in the first line, the sequence of base calls in the second line, a separating third line that usually only includes a plus sign and a fourth line that contains the quality scores corresponding to each base call.

Box 2.2.0 ASCII Code

The American Standard Code for Information Interchange (ASCII) is a bitwise representation of characters. Initially, it was designed to use 7 bits for encoding and the eighth one for error correction. Thus, the standard could represent 128 different states. The first 32 codes are reserved for control characters, and the remaining ones represent printable characters, including numbers (0-9) and the alphabet in lower- and upper-case. The extended ASCII code also uses the eighth bit for encoding, which expands the number of characters to 256. Because the standardized format uniquely represents characters in a single byte, it is helpful to exchange data between different file formats, programs, and operating systems [91].

The sequence ID in the first line is a combination of characters (string) that uniquely identifies every sequence in the file. Everything up to the first space is considered the sequence ID. More information can be incorporated into the first line after the sequence ID, and the concrete format has changed with development

Header 1	Header 2	Meaning
M00967	M00967	Instrument ID
43	43	Run ID
000000000-A3JHG	000000000-A3JHG	Flowcell ID
1	1	Flowcell lane
1101	1101	Tile within flowcell
18327	18954	X-coordinate within tile
1699	2112	Y-coordinate within tile
1	1	Pair ID
N	N	Filter status
0	0	Control bits
188	189	Sample ID

Table 2.1: The table lists the meaning of positions in a FASTQ sequence header comparing two samples that were processed in the same sequencing run.

versions. It will generally contain information about the internal filtering steps of the sequencer as well as the multiplex code or sample IDs. Table 2.1 shows the structure of the sequence ID.

Phred score is an estimate for base calling quality. It is encoded in a single ASCII character and available for every base in a sequence [3, 91, 141]. The metric represents the probability of an incorrect base call on a logarithmic scale. Phred scores range from 0 to 93, i.e., a definitive calling error with a probability of 1 to the improbable event of a single wrong call in more than one billion bases, but commonly do not exceed 60.

$$\begin{aligned} Q &= -10 \log_{10} P \\ P &= 10^{\frac{-Q}{10}} \end{aligned} \tag{2.1}$$

Phred	ASCII	Error probability	Accuracy
10	+	1 in 10	90.0%
20	5	1 in 100	99.0%
30	?	1 in 1,000	99.9%
40	I	1 in 10,000	99.99%
50	S	1 in 100,000	99.999%
60	]	1 in 1,000,000	99.9999%

Table 2.2: According to the Sanger scheme, the table shows an example of Phred scores encoded as ASCII characters. Higher Phred values indicate a reduced probability for an erroneous base call at a sequence position.

2.2.5 Preprocessing

Perhaps the most critical step regarding the quality and replicability of downstream analyses is confirming the correct assignments of sample IDs and the completeness of the data set. The demultiplexed data, i.e., separated by sample ID after the sequencing procedure, will be available in the form of single files. The naming scheme of these files is dependent on the sequencing service provider and the tools that were employed. For example, for paired-end sequencing without replications, two files can be identified by sample ID and some binary marker related to forward and reverse reads, e.g., R1 and R2 for forward and reverse reads, respectively. This information needs to be matched to the sample sheet to determine whether or not the data is complete. This step also includes checking file integrity, i.e., the gold standard is to calculate md5 checksums and compare them to the ones that the service provider usually provides [128]. Depending on the result, this procedure may make it necessary to require some files or exclude specific samples from the analyses. Once this step is concluded, the actual processing of files can begin [22].

Box 2.2.0 FASTQ file infos

Technically, it is possible to contain the complete output of any NGS sequencing run in a single file. Since every sequence can be uniquely identified by its sequence ID, they could be extracted according to sample, lane, or flowcell. However, due to advances in storage and processing memory, the previously

limiting technical factors are also negligible in the modern computational environment. Therefore, it is essentially more convenient to handle multiple single files and deal with errors that might occur during transmission.

Pre-processing steps aim to create a set of consensus sequences in the form of operational taxonomic units (OTUs) or amplicon sequence variants (ASVs), i.e., cleaned up sequences that can undergo taxonomic assignment. To that end, the following steps need to be performed.

Quality Control

Quality control comprises the Phred score-based trimming and filtering of sequences and the removal of chimeric sequences. Several user-defined parameters determine whether a particular sequence remains part of the data set or will be excluded. Stringent protocols will remove sequences containing unidentified bases, i.e., positions with an N instead of a character that refers to a nucleotide. A maximum value for accumulated error probability may be set. For example, in the processing pipeline DADA2, the sum of error probabilities needs to be smaller than the maximally allowed error. Falling below a set minimal length can be an additional exclusion criterion for reads.

The quality profile of a sequence, i.e., the value of Phred scores, deteriorates with reading length. Base-calling becomes less reliable the more sequencing cycles are performed. Hence, the confidence in the correct assignment of bases drops for longer sequences. Therefore, reducing noise for downstream procedures is essential to ensure reliable information, i.e., uncertainty, in the data. Sequences can be trimmed by setting a specific cutoff for forward and reverse reads, respectively. Additionally, it might be advisable to activate automatic truncation if the quality scores drop below a set value for the first time.

An essential consideration for paired-end reads is the required overlap for the merging procedure. For example, depending on the employed primer set, i.e., which region in the 16S rRNA gene has been selected for sequencing, it might be necessary to soften the QC parameters to retain enough sequences to merge the reads.

$$\varepsilon_{max} < 10^{\frac{-Q}{10}} \tag{2.2}$$

Chimeras are erroneous sequences that are created during PCR amplification. The artifacts are formed when an incomplete reaction allows for a free-floating fragment instead of a single dNTP base to be attached to a partial strand. Subsequent cycles will amplify these sequences, which can compose more than 40% of all sequences in a mixed template environmental sample, according to NCBI [121].

Read Merging

An additional merging step is required for paired-end reads to produce a contig that can be used in subsequent steps. It is not helpful to retain both strands of DNA sequence since either can be inferred from the other. Hence, a contig is a consensus sequence that results from merging both read pairs. To that end, a forward read and the reverse complement of its' corresponding reverse read are aligned and then merged. The reverse complement of the reverse read is calculated because it needs to continue the sequence of the forward read [56, 96]. Aligning means that both reads are justified so that the bases in the overlapping part are at equal positions. There are several advantages to paired reads. First, the resulting contig sequences can be longer than the single reads in either direction. Secondly, the merging procedure constitutes an additional filtering step. For the overlapping part, the base-calling quality is calculated twice. In case of diverging calls, the strategy is to either select the better quality or discard both reads. Thus, it is an additional level of filtering for all pairs that can not be matched [104].

Unique Sequence Identification

Considering all the remaining sequences as distinct species contains the risk of falsely identifying sequencing errors as new entities. Clustering approaches were developed to ease this potential by grouping similar sequences and selecting a consensus sequence. The assumption is that closely related species will have similar sequences in targeted genes, and relative specificity can be maintained despite clustering them together. On the other hand, agglomerating these similar sequences to find a consensus sequence will mitigate the effect of sequencing errors because they represent an insignificant amount of bases in these clusters. The resulting reference sequences are operational taxonomic units and can be computed following three different strategies [100].

The de novo clustering approach computes OTUs by aligning and scoring all identified sequences. This operation is computationally expensive, creating potentially long execution times depending on the dataset size. Because all the sequences are part of the clustering model, the model is prone to change once the dataset has been modified. Thus, it is necessary to repeat this step if samples are added or removed or in case multiple sets are to be compared [12, 106]. In reference-based clustering approaches, a database of known taxa is used as a source of targeted gene sequences. The sequences in the data set will be compared to this reference to generate new clusters based on the similarity threshold. The computationally most efficient method, closed-reference clustering, will discard reads that can not be aligned due to errors or because a matching gene sequence is not part of the reference. It is the computationally most efficient method. The unmatched reads will be clustered with a subsequent de novo clustering step when using an open-reference strategy. Hence, this strategy is a trade-off between the computational requirements of de novo clustering and the mapping quality imposed by the comprehensiveness of the reference database [27]. An issue common to all three strategies is the choice of an appropriate threshold for sequence similarity. Using a low cutoff increases the possibility of grouping similar species and losing the individual signals. A high threshold can falsely identify sequencing errors as separate organisms and inflate diversity. A commonly employed threshold for similarity is 97%.

Amplicon sequence variant (ASV) analysis is a relatively new approach that calculates an error model based on all available sequences and associated reads to filter out erroneous sequences based on significance values. Sequences are retained according to the statistical confidence in their correctness. This approach does not require a reference database or repeated computation due to changes in the dataset. Hence, ASV analysis is significantly faster than OTU clustering and enables straightforward comparison of data sets. While the choice for confidence value cutoff in this approach still potentially risks dismissing organisms or inflating diversity, the employed statistical model does not rely on biological assumptions about sequence similarity due to species relatedness. Thus, the ASV approach offers an advantage for the precise characterization of microbiome samples. Regardless of the chosen approach, OTUs and ASV alike will be compared to reference databases in order to assign a taxonomy to the final set of sequences [2, 7].

2.2.6 Analysis

Once the preprocessing procedure is completed, the data can be imported into the framework for statistical analysis. In an initial exploratory step, the dataset will undergo further quality control measures that control for unwanted variation at the analyst's discretion. This usually involves investigating the library sizes of the remaining samples and removing low abundance samples. Furthermore, the data should be tested for potential contaminants. Singleton removal is a measure to reduce noise by removing all features that are not present with minimal abundance in a set number of samples. The assumption is that these sequences have no relation to the biological factor of interest and can be considered artifacts due to their insignificant amount. Thus, these features contribute no helpful information to the subsequent analysis steps, and their removal will reduce random noise. Part of this process should be the assessment of potential batch effects to decide if it is necessary to account for these factors in the subsequent analysis steps [134]. The particulars of this procedure will be outlined subsequently in Section 2.3.

On average, human tissues express about 12000 different gene products simultaneously, including roughly 8000 ubiquitously found products, i.e., they are expressed in every tissue. In statistical analyses on abundance values for gene products, the underlying assumption is that most features will not be significantly different between groups, and only a tiny amount reflects the biological factor of interest. This means that a subset of genes can not be found in either group. Hence the data will contain a broadly similar abundance for all samples. This assumption is not necessarily valid for microbiome data sets. Certain micro-organisms have been associated with the distinct biomes of the human body, i.e., they are reliably found in similar samples, but generally, there exists a higher variability in the actual composition of communities. This is connected to host disease, nutritional and sanitary habits, pets, age, sex, living situation, and environment. This circumstance introduces the challenge of compositionality [92]. Let twenty genera be identified in samples A and B, respectively. Only 10 of those genera match between these two samples, i.e., 10 out of twenty are equal while 20 distinct genera are distributed over two samples. This circumstance creates a 30 by two matrix where one-third of the values are zero because these entities were not found in a sample.

The sparse compositional nature of microbiome data [117] represents a challenge for the analysis of microbiome data. The term compositional data describes a multivariate dataset whose components capture an estimate of some whole. As such, the values are constrained to be positive (including 0) and generally sum up to a fixed value, i.e., the library size for an individual sample. This can also be conceptualized as ratios; by dividing all values by the sum of all values, the sum constraint will be set to 1, and values represent percentages of abundance. Additionally, it means that the values can not vary freely. They are constrained to be positive because something is either part of the whole or not; it cannot be represented negatively. Secondly, this implies at least one correlation coefficient between elements to be negative because the total sum is constrained. For example, in a two-part composition, the correlation between these two elements has to be -1 because a change in one implicitly affects the other in the opposing direction. Essentially, every sample cohort selected to represent a specified population is a composition. Because experiments can never include every existing individual, the goal is to select a group of individuals representing the whole population concerning any conceivable variable. In transcriptomic datasets, each individual sample can be considered an estimate of mRNA composition in the host's tissue at the sampling point. The assumption is that it is an accurate estimate of the actual composition. Naturally, the number of possible transcripts is limited by the number of genes and respective iso-forms they can produce. Another limiting factor is the type of tissue the sample was taken from; the functional pathways determine the genes expressed here. With microbiome data, this characteristic is just more pronounced [62].

2.3 Batch Effects

In electrical engineering, a signal is delineated as a quantity observable in single or multiple dimensions. An intuitive example of a two-dimensional signal is a digital black and white photo. The image represents an observable scene from the real world that has been broken down into a discrete matrix of numerical values. These values correspond to the brightness at the positions within the captured scene when the image was taken. The image results from a signal capturing process and enables storage, reproduction, sharing, and analysis of this scenery at a particular moment

in time. This is commonly referred to as signal processing or, more specifically, image processing.

Sequencing samples to determine the presence and abundance of microbial species at the time of sampling, i.e., taking a snapshot of the microbiome's communal composition, is essentially a signal capturing process. As stated in the previous sections, it is literally an image capturing approach within the sequencing machines. After the necessary pre-processing steps have been performed, the results that undergo statistical analyses are conceptually very similar to a digital image. The data has two dimensions, i.e., the feature space consisting of OTUs (or ASVs) and the sample space made up of all the individual entities that have been sequenced. Moreover, like an image, the sequencing data is subject to unwanted signal modification during capture, storage, conversion, and processing. These undesired alterations of the data are commonly referred to as noise, and since its emergence is an inherent quality of any signal capturing process, there is a long history of dealing with it.

A particular form of noise that is more pertinent to OMICS data is the Batch Effect (BE), a cumulative systemic error that is introduced by processing samples in different batches. The subsequent section will elaborate on the nature and sources of BEs and the problems they raise. Furthermore, it will outline ways to prevent the emergence of BEs or correct the resulting variation if they are present.

In the subsequent chapters, the terms noise, error, and batch effect will be used somewhat interchangeably, and all refer to unwanted variability as a confounding factor that results from processing in distinct batches.

2.3.1 Sources

The notion of processing in different batches virtually extends to every conceivable way of grouping samples and performing one or more steps of the sequencing pipeline in separate groups. Hence the sources that introduce this particular form of noise do not differ from any common source. An example is the technical variation introduced by a particular sequencing machine. Every signal capturing process has an inherent amount of uncertainty that is added to the captured signal. The manufacturers aim to reduce this technical variation to a minimum and guarantee that it will stay under a given threshold for every machine they produce. This variation is assumed

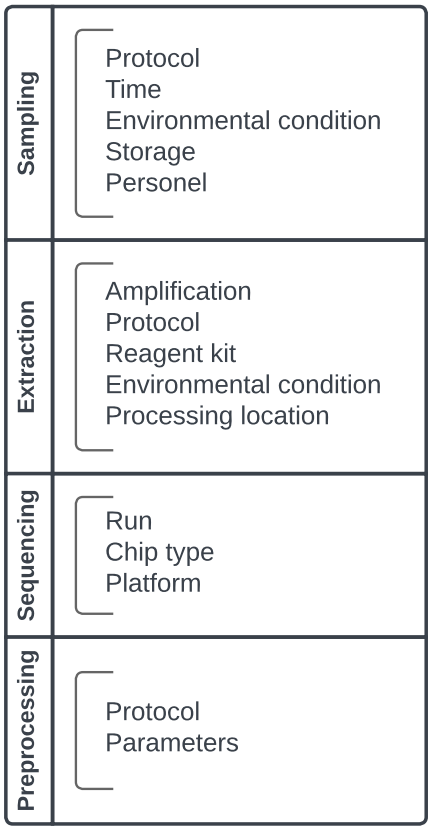


Figure 2.3: The figure shows potential sources of batch effects for the previously outlined data generation steps.

to affect all the samples homogeneously and can thus be represented with a uniform distributed error term in modeling and calculations [99].

However, the variation added to a data set will vary in different runs on the same machine, and it will be different between groups of samples that were processed on different machines of the same type and model. This is a technical factor that will introduce batch effects [147]. However, the amount of variation can be exacerbated by environmental factors at the time of processing, e.g., temperature, humidity, or exposure of the samples to sunlight [114].

The extraction procedure also relies on a variety of reagents for processing. For example, it is conceivable that changing reagent batches due to empty bottles will also introduce differences.

Some amount of variation can also be attributed to the human factor within the processing pipeline. It is a reasonable statement that a lab technician with ten years of work experience delivers work with higher speed and quality than an apprentice.

When implementing a new protocol for the first time, the processing time will presumably increase with every sample. Furthermore, it is also known that the quality of work will decline over the course of a day due to exhaustion. In the same way, it is understood that there will be a difference in quality between two different people that process the same type of samples with the same protocol at the same time.

This factor relates to all processing steps starting with the harvesting of tissues. Technical variation is introduced by a different execution of the sampling protocol, the amount of time the samples are subject to environmental conditions or freeze and thaw cycles for storage and processing [114]. In addition, there might be considerable periods between the acquisition of several samples, e.g., in a time-series study.

All these factors introduce minor variations in the data, and although they are small, they will add up and could obfuscate biological effects.

2.3.2 Challenges

Concerning the biological effect of interest, batch effects have roughly two influences on data sets, masking and modulation. The latter occurs if the biological group, e.g., case and control or wild-type and genetically altered mice, are processed and sequenced separately in their respective groups. The variability that is introduced between the biological groups will be added or subtracted from the actual biological effect. Thus, a statistical analysis can produce over-/under-estimated values for the significance of the examined data, which generates false positives and negatives. Batch Effects are a confounding factor in balanced experiment designs that adds additional noise to the data. This variability can obscure the biological difference between groups and reduce the reproducibility as the discovered effects might result from accumulated technical variability [48].

Similarly, it is a limiting factor for comparing different study results. Integrating multiple studies to gather larger data sets that offer more statistical power is a BE problem. For example, two equivalent studies with the same design that used the same protocols and sequencing technologies are essentially two batches that require adjustment to present a coherent picture of the underlying biology.

For sub-populations, e.g., a proportion of either experimental group with slightly aberrant behavior compared to the rest of the group, the BE can conceivably mask their existence by introducing enough variability to hide them. Classification methods commonly used to identify biological markers are also affected by batch effects, as they introduce a bias that prevents correct classification and model cross-validation efforts.

2.3.3 Rectification

Batch Effects can be categorized into two classes, systematic and non-systematic batch effects. Systematic BEs affect all features (OTUs) in a batch similarly, i.e., the variability introduced by the sequencing machine is likely to be systematic as it relates to the technical process that does not rely on biological interactions. Therefore, it is not as easy as adding a fixed offset to every feature in a batch, in any case. The systematic BE can be conceptualized as a set of values that have been drawn from similar normal distributions so that the values differ for particular features and samples within a batch, but on average, they are comparable.

Non-systematic BEs have a heterogeneous effect on the OTUs. That means that they do not affect all features in the same way. Moreover, some features might remain unaffected by the batches. Others will be affected by BEs to a different degree. Whereas systematic BEs are considered to be drawn from roughly the same uniform distribution, the mean values for non-systematic BEs can be very different [19]. One example of non-systematic variability is the amplification bias introduced in the PCR step. The idea is that the present genetic material in a sample is multiplied for the sequencing step. In principle, this can be conceptualized as increasing every distinct sequence by some factor. In practice, a bias will be introduced throughout multiple PCR steps because more prevalent sequences will react more often while less abundant ones will not. This results in higher proliferation rates for already more abundant sequences, i.e., an effect that is more prevalent in some sequences

than others [19]. Concerning the previously outlined sources of batch effect, e.g., environmental conditions and reagent batches, this can introduce a non-systematic batch effect when samples are processed some time apart or by different personnel.

Independent from the nature of Batch Effects, they need to be controlled and accounted for to facilitate meaningful, reliable, and reproducible scientific results. The two major approaches to mitigate batch effects are prevention and correction. They can be considered complementary tools since neither can remove batch effects.

Prevention

It has been shown in section 2.3.1 that noise is an inherent factor in any signal processing approach. Batch effects differ from common noise only because they correlate with samples processing in particular sub-groups, i.e., processing batches. Unwanted variation will inevitably be created, and therefore two pathways are essential to alleviate the severity.

First is a stringent experimental protocol that minimizes the generation of unwanted variability. This approach relies mainly on instructing and training the personnel at the respective sample processing steps. Instruction means cultivating an appreciation for the detrimental effects of unwanted variation on the utility of the generated data in general. In particular, it means educating about the factors that lead to the emergence of batch effects. It is one thing to be well trained and able to produce qualitative work by strictly adhering to the methods. It is another also to be well informed about the impact of additional factors and make conscious, considerate decisions. This aspect starts with collecting samples and might mean that only a single person is tasked with this step. Furthermore, it holds for the rest of the processing pipeline [134].

It is, for example, not feasible to control and document all environmental variables for all samples. First of all, the overhead for documentation would be enormous, and secondly, it is questionable that this information is helpful in statistical analyses. Every documented environmental variable, e.g., temperature, processing time, humidity, persons who collected the samples, and researchers who worked in the lab, presents a new covariate in the statistical analysis and a potential new batch. Some of these covariates can be combined to reduce the overall number, e.g., personnel responsible for sampling and lab work can be combined into a single variable. Fi-

nally, the newly created covariate will fractionate the data into new batches. This combination can be performed with categorical data but is not suited for continuous numerical data like temperature or humidity. The other limiting factor is that at some point, the newly generated batches will contain very small sample sizes or even fractionate the study into single sample batches. At that point, the collected data becomes useless because it essentially represents that every sample is a distinct entity.

These circumstances led to the second option for BE prevention and informed statistical experimental design [134].

A strategy that needs to be studiously avoided is to confound batches with study groups. From an organizational point of view, it might seem convenient to process the samples in blocks that relate to the experimental grouping, i.e., all the case samples are processed and sequenced in a single run and then follow the healthy controls or other groupings that are part of an experiment. However, in this case, the differences between the means of the groupings are entirely confounded with their batch. Furthermore, since all the samples of one study group are aggregated in their respective batches, there is no approach to include the batches in a modeling approach that aims to control this effect. A severe example of this issue is a published study of the Mouse Encode Consortium [168]. After sequencing and analyzing the expression data of 10 matched tissue types from human and mouse samples, the authors proclaimed that gene expression is more similar among tissues within a species than between corresponding tissues of two species. This finding has significant implications, considering the overwhelming use of mice as model organisms in experimentation because it means that results from this type of experiment are in no way transferable to humans. A re-analysis of the same data-set ascertained that most of the tissues that belong to a particular organism had been processed and sequenced in blocks with only some exceptions in which both species were put into identical batches. It was, however, enough to account for these batches in the statistical modeling and reveal that tissues of the same type are similar to each other regardless of the species in which they originated [87].

One approach is to perform the entire experiment in a single batch. Assuming that that entire experiment can be performed without disruption in a single day and also processed on a single machine in one run, it would be optimal to do so in order to reduce batch effects. However, the differences between the group mean,

i.e., case versus controls or treatments against diseased subjects, will not be affected by batch effects. Concerning reproducibility and comparisons between studies, it has to be noted that the batch-treatment interaction that relates to the performing investigator can neither be eliminated nor calculated in this design. Consequently, the same investigator can reproduce the experiment while another person cannot do so [87].

Another approach is to replicate the complete experimental setup with the batch. Assuming that it is not feasible to process the complete experiment in a single batch due to a large number of samples or some other constraint, it would be best to aim for a balanced distribution of samples from every group under investigation in every batch.

Correction

For existing data sets, the previously presented strategies are unavailable. In these cases, the severity of the batch effect needs to be evaluated and accounted for if necessary. In addition, several factors are likely to affect computational approaches to batch effect correction.

The first factor is whether or not there is knowledge about batches, i.e., does the covariate data include factors that account for the distribution of samples regarding processing in batches. If no such information is provided, does it mean that the samples were not processed in batches, or were batches not within the scope of attention of the executive researchers? In the latter case, is it possible to recover this information? For example, when dealing with raw data, it is feasible to retrieve information about sequencing machines, their lanes, and in which runs the samples were processed. Other factors such as environmental variables, which lab technician processed which sample or freeze-thaw cycles are commonly not recorded and are virtually irretrievable with an acceptable level of certainty. This ties into the next important factor, the number, and fragmentation of batch factors. Even if it was feasible to collect and document all variables related to the formation of batch effects, e.g., environmental variables, reagents, and personnel, it does not necessarily mean that this would lead to a better ability to correct for bath effects. For one, the model becomes more complex with every new variable, and it would be necessary to evaluate and control for the impact of every single variable, which in itself pro-

duces uncertainty. Secondly, assuming that the various collect batch factor does not miraculously describe the same separation of samples into batches, i.e., considering sequencing runs and processing days as separate batches, could lead to this particular situation as it is probable that there exists a strong correlation between the time samples were processed and scheduled for sequencing. Assume three batch factors, wet-lab staff, processing day, and reagent batches, each with two categories, i.e., two lab technicians, T1 and T2, two days D1 and D2, and two batches of reagent B1 and B2. There are already eight possible combinations (batches) of circumstances that potentially introduce variability into the data for these three variables. Each new variable potentially leads to an exponential growth in the number of batches which also implies that the number of samples in a data set is a limiting factor in correcting batch effects.

Next is the degree to which batch effects are confounded with the class effect. The class effect denotes the biological effect of interest, e.g., case and control groups or wild-type and genetically modified mice. For a well-balanced design where samples of both groups are equally distributed over two or more batches, the impact of BEs can be estimated for each batch. Due to the subtraction principle, the batch effect is removed when calculating the differences between groups, i.e., differential abundance. Notably, the complete symmetry of study designs is generally not commensal with reality. Due to lack of quality, missing covariate information, or patient dropout, samples may be excluded. However, even a somewhat imbalanced design is preferable to complete confounding between class and batch factor, i.e., the groups that are implied by class and batch factor consist of precisely the same samples. In this case, there is no way to distinguish the impact of the biological class effect from that of the batch in any significant results [62].

The choice of upstream normalization procedure affects both the need and the type of BECA as well. Compared to complex computational correction methods, less severe BEs can be dealt with by variance stabilizing transformations.

Finally, the type and severity of the BEs are essential factors. However good an algorithm performs in estimating and removing the noise attributed to batches, it will remain an estimate with some degree of uncertainty. Systematic BEs are somewhat insignificant in balanced designs as they cancel each other out within batch comparisons and could, if required, be represented with a high degree of accuracy by a uniform distributed error term in the case of less balanced designs.

Non-systematic BEs are more complex to evaluate, i.e., which features did they impact, are the same features impacted in the same way across all samples, is the magnitude comparable between affected features. For lower magnitudes of impact, it might be appropriate to apply a variance stabilizing transformation and account for the batch by including an additive effect in the model if appropriate. Otherwise, the error that will be introduced into the data by correcting for an uncertain model of weak noise might be more severe than the actual batch effect [25, 110]. This can then mask the class effect when it otherwise would have been measurable or, even worse, exaggerate it and produce false significance.

The following challenges that need to be taken into account are concerned with the characterization of the data set. As for statistical analyses, the dataset size significantly impacts computational methods that aim to correct batch effects. This relates to the number of samples and the number of features to examine. It is generally better to have a large samples-size while investigating as few distinct measured parameters as possible. Furthermore, it increases the statistical power. The same remains true for BE correction. A large sample size implies a large amount of information that can be leveraged to evaluate the severity of batch effects. It probably also indicates the presence of multiple batches seeing that, at the very least, the sequencing machines are somewhat of a bottleneck with their limited array sizes. However, if everything else is stringently controlled for, then it is feasible to handle this systematic bias). Moreover, a small feature space means that only a few variables could have been affected by batch-processing; thus, the confidence in correction methods will be higher.

The importance of good experimental design has been outlined in the previous sections. It is worth reiterating that a good thought-out experimental design is the most critical factor in avoiding batch effects. It is possible to prevent the emergence of batch effects or reduce their impact by blocking nuisance variables accordingly. Equally, the choices made while designing an experiment can make it virtually impossible to retrieve any meaningful results.

Another challenge is the availability of batch factors at the time of computation. While raw data files can be investigated for information about sequencing runs, it is almost impossible to retrieve information about nuisance variables such as temperature, processing researcher, or processing time if these were not documented. Although BEs can be inferred computationally, there remains an inherent biological

heterogeneity that leads to uncertainty. This can result in masking actual biological effects that are mistakenly removed [36]. The final challenge in dealing with BEs is the desired goal, e.g., in integrating multiple studies, each study represents a batch. If the goal is to train classification models, it might be more practical to train on either batch and test on another instead of combining them.

3 Materials and Methods

This chapter is concerned with the computational framework of this work. It has been shown that the conception and implementation of experiments is a significant contributor to the mitigation of batch effects [87]. The aim is to provide a tool to treat datasets that contain batch effects. Naturally, the basis for this process is formed by an analytical conception of the problem. This chapter starts with an introduction to the modeling of data sets and the mathematical concept of variability. Building on this foundation, the subsequent section will outline the used Batch Effect Correcting Algorithms (BECAs) and expand on their underlying assumptions to explain how they work. Following this, the framework for the software development portion of this work is outlined. Finally, the chapter concludes with an introduction to the utilized datasets.

3.1 Representing Variability

The previous chapter discussed the biological characteristics of microbiome samples, how NGS technology can capture this signal, and the pre-processing steps required to create a dataset of feature abundance table and covariate information that reflects a biological system. So far, variability has been considered either as the presumed consequence of the investigated biological factor or as a confounding technical artifact that affects the measured values. In order to allow statistical analyses and computational processing, an analytical framework that describes datasets and the inherent variability is required [98].

3.1.1 Model

In order to explain the system's characteristics and study the effects of different components, a mathematical representation, in other words, a model, is required.

This work will employ linear modeling as it is a standard method and the basis for most of the subsequently explained concepts. Linear models can be considered simple equations; both sides of the equation represent different entities but have equal value. Representing the outlined minimal data set in as a linear model results in the following Equation 3.1.

$$Y_{m\times n} = \beta_0 + \beta_{m\times p} X_{p\times n} + \varepsilon_{m\times n} \tag{3.1}$$

Let matrix Y represent the abundance table with $|m|$-rows for features in $|n|$-columns corresponding to samples. This matrix contains the observed responses, i.e., abundances as captured by the NGS procedure, presumed to result from the equations right-hand side. The matrix X denotes the observed predictive variables that are the factors of interest. The number of rows relates to the number of factors. Hence, in this example, X is a one-dimensional row vector of length $|n|$. The actual relation between Y and X is contained in the coefficients denoted by β. The first coefficient β_0 essentially contains the average abundance for all $|m|$ features, i.e., a baseline value. The second coefficient $\beta_{m\times p}$ measures the impact of the $|p|$ independent variables in X on the observed abundances in Y. Finally, ε is a term used to denote normally distributed noise with zero-mean, i.e., the homogeneous variation inherent to all signal processing procedures. In other words, describing an experimental microbiome dataset as a linear model means creating an equation where measured dependent variables on the left side of the equation result from an average abundance and the effect of a measured independent variable on the right side. The noise term captures any additional unaccounted variability. In this framework, matrix Y is the dependent or response variable, and matrix X is the independent or predictive variable. This naming scheme is appreciable since the differences within the abundance table are considered to be dependent on the effect of the factor of interest within the sample table. Hence the assumption is that the factor of interest can be used to predict the differences in Y. This way of modeling the data lends itself to linear regression, where the goal is to estimate the coefficients.

This matrix notation can be easily extended to incorporate more than one observed factor of interest, i.e., several biological factors that are to be investigated by the particular study because the notation still works for values of p larger than 1. In Section 2.3 the distinction was made between variables of interest and nui-

sance variables. The idea between this distinction is that although these variables might be easy to capture and can have a significant impact on the observed data, e.g., differences in sex or ethnicity are well documented to have a significant impact in biological studies, it is not obvious which of those variables need to be part of the final model. However, conceptually the class of nuisance variables contains all observed additional information that is not the focus of a particular investigation. For example, an observed batch factor will be considered a nuisance variable that might need to be included and accounted for in the final model.

The final distinction relates to observed and unobserved factors. All the variation in a data set that the observed independent factors can not explain is related to one or more unobserved factors. This uncertainty can be captured by the error term ε to some degree, although the goal is to minimize this term as a considerable degree of uncertainty implies unreliable or even faulty models. Moreover, as some of the subsequently described algorithms aim to infer these factors from the data, this outline will contain a specific term related to unobserved sources of unwanted variation. These points result in the extended linear model representation shown in Equation 3.2.

$$Y_{m\times n} = \beta_0 + \beta_{m\times p}X_{p\times n} + \gamma_{m\times q}Z_{q\times n} + \alpha_{m\times k}W_{k\times n} + \varepsilon_{m\times n} \tag{3.2}$$

Additionally, observed covariates or nuisance variables, e.g., sex, age, gender, comprise the matrix Z. The unobserved factors that correspond to unwanted variation are part of matrix W. The distinction between observed and unobserved factors is an important one. If all the matrices were observed, this problem could be considered a standard linear regression that aims to estimate the coefficients β, γ, and α. However, in this case, both W and α are unknown and only specified to the point that their product should create a matrix $\hat{W}_{m\times n}$ that has the exact dimensions as Y and contains an offset for every feature in every sample that corresponds to undesirable variation. The size of k, i.e., the number of unobserved factors that correspond to the unwanted variation, is unknown as well and must be determined before estimating W.

3.1.2 Metrics

So far, this work referred to noise as unwanted variation that is an inevitable consequence of signal capturing procedures. More specifically, in the context of molecular biology, it is the result of the processing steps in NGS technology. The term variation is a common concept in daily life, e.g., the variability in types of cars on the road, haircuts of co-workers, or types of flowers, and can be understood intuitively. A more refined understanding of variation concerning data sets is presented in the subsequent sections.

Variability

Two concepts facilitate the characterization of data sets, central tendency and variability [129]. First, let the data set contain the number of minutes spent with physical fitness activities per week for n university post-doctoral employees. Assuming that the data represents an appropriate population sample, there will be differences between individuals. A metric of central tendency, such as the mean (Equation 3.3), measures how much time on average is spent on fitness by this particular sample of individuals by summing up all data points and dividing by n. Technically, a distinction between sample- and population mean can be made. Since working with a whole population is generally unfeasible, this work will discuss sample-related metrics.

$$\mu = \frac{1}{n-1}\sum_{i=n}^{n} x_i \tag{3.3}$$

Another measure of central tendency is the median shown in Equation 3.4. For an ordered list of values X, the median will return the value in the middle of the list. In the case of an even amount of numbers, the median is calculated as the mean of the two most center values. This approach makes the median a more robust metric to outliers than the mean.

$$Med(X) = \begin{cases} X[\frac{n+1}{2}] & \text{if } n \text{ is odd} \\ \frac{(X[\frac{n}{2}]+X[\frac{n}{2}+1])}{2} & \text{if } n \text{ is even} \end{cases} \tag{3.4}$$

Variability measures how far the data points diverge from the central tendency of the dataset, i.e., variance measures their dispersion around the mean of all values.

While the central tendency can be considered a baseline value for a sample, the variability can be thought of as an indicator of the reliability of a sample, i.e., low variability means less uncertainty when making predictions based on the data.

Metrics for variability relevant to this work include the interquartile range (IQR) as shown in Equation 3.5. For an ordered list, the term quartile denotes one of four sequential blocks that contain 25% of values each. The IQR describes the spread of values as the difference between the maximum in $Q3$ and the minimal value in $Q2$. In other words, the IQR is the difference between the medians of the upper and lower half of values.

$$IQR = Q3 - Q1 \tag{3.5}$$

The next metric is the variance which indicates the degree of spread within the data. Computation of the variance comprises calculating the sum of the squared distance to the mean for all values divided by the number of values as shown in Equation 3.6. Hence, the measure of variance is expressed on a different scale than the underlying data and is, therefore, less intuitive to interpret. However, the variance is the basis for statistical inference as it facilitates the comparison of different groups in terms of dispersion.

$$s^2 = \frac{\sum(X - \mu)^2}{n} \tag{3.6}$$

The final metric is the standard deviation (sd) which indicates how far, on average, each value is away from the mean. Equation 3.7 shows that it is essentially the square root of the variance and is thus given in the same units as the underlying data. The standard deviation is consequently more intuitively interpretable than the measure of variance.

$$s = \sqrt{\frac{\sum(X - \mu)^2}{n}} \tag{3.7}$$

The previously introduced metrics are a measure for the amount of spread in a single set of observations, i.e., one variable with multiple measurements. Let x and y denote two vectors of $|n|$ measurements for two distinct variables, e.g., height and weight. The variance will quantify the degree to which either variable deviates from

its mean value. The covariance described in Equation 3.8 is a quantitative measure of the relationship of these changes between the two variables.

$$\sigma(x, y) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \tag{3.8}$$

The covariance contains information about the linear relationship of the variables. It is reasonable to assume a positive covariance value for the two variables, height and weight, because a taller body relates to more bodyweight. The covariance will have a negative value if the variables are inversely related, i.e., if x gets larger, then y will get smaller. Moreover, in the case of absolutely no linear relationship, the covariance between the two variables will remain at 0. This metric can be computed as a single value for two variables or in the form of a covariance matrix for multidimensional data. The following Equation 3.9 will calculate the square covariance matrix C for the $m \times n$ matrix X.

$$C = \frac{1}{n-1} \sum_{i=i}^{n} (X_i - \hat{X})(X_i - \hat{X})^T \tag{3.9}$$

This formula will also compute the covariance between x_i and itself, which default to the variance. Thus, the matrix C is also called the variance-covariance matrix.

Principal Components Analysis

One of the significant challenges in NGS data sets is the extensive feature space, consisting of several hundred to thousands of operational taxonomic units or ASVs, compared to a relatively small amount of samples. A large number of data points for each sample limits statistical power and representability. Fortunately, the concepts that have been introduced so far give rise to a data-driven approach that can be used to reduce the dimension of a data set while conserving most of the contained information. The Principal Components Analysis (PCA) is an established tool in most computational and statistical fields of study because of this property. The procedure computes a new hierarchical coordinate system by creating new variables through independent linear combinations of existing features in the direction of the most extensive variation in the data. The four consecutive computational steps of Principal Components Analysis are shown in Equation 3.10.

$$
\begin{aligned}
\text{Mean center:} \quad & B = X - \bar{X} \\
\text{Covariance:} \quad & C = Cov(B) \\
\text{Eigendecomposition:} \quad & CV = VD \\
\text{Principal Components:} \quad & T = BV
\end{aligned}
\tag{3.10}
$$

Centering each variable on its mean value is a standardization procedure that makes the variables comparable, and the covariance matrix is a square $n \times n$ matrix that makes the subsequent eigendecomposition possible. The eigendecomposition is a matrix factorization that creates the matrix V of eigenvectors and vector D with corresponding eigenvalues. The specific mathematical properties of each component are out of the scope of this work, but they possess characteristics that are of interest in the context of unwanted variation.

1. The eigenvectors are linear combinations of the original variables and form a new coordinate system, i.e., they form a new variable and reduce the number of variables required to represent the data.

2. The eigenvectors are ordered by the amount of variability in the original data they represent, i.e., the first principal component accounts for most of the variability, followed by the second PC, and so on.

3. The corresponding eigenvalues are proportional to the variability captured in their respective eigenvectors. Hence, the eigenvectors represent the principal variance components present in the data. For $|n|$ number of features exactly $|n-1|$ non-zero eigenvectors, that can form principal components, exist.

In summary, PCA transforms a set of correlated features by representing them in the form of orthogonal unit vectors that correspond to best-fitting regression lines. Thus, the method facilitates feature selection and dimensionality reduction.

Singular Value Decomposition

The singular value decomposition (SVD) is the generalized form of the eigendecomposition introduced with PCA. The general idea behind SVD is that any set of vectors, e.g., a matrix, can be expressed in terms of the lengths of their projections onto orthogonal axes. In contrast to the eigendecomposition, the SVD can be calculated for any $m \times n$ matrix and will result in the factorization shown in Equation 3.11.

$$X_{m \times n} \cdot V_{n \times n} = S_{m \times n} \tag{3.11}$$

In this representation, the data matrix X is multiplied by matrix V of decomposition axes, i.e., an alternative coordinate system to display X in, and S is the projection of X in this new coordinate system. Normalizing the matrix S by dividing the columns by their magnitude will create the following representation of Equation 3.12.

$$S_{m \times n} = U_{m \times m} D_{m \times n} \tag{3.12}$$

Here, the matrix U still contains the projection lengths onto V but is scaled to unit vectors, and matrix D is a diagonal matrix containing the scaling factors. With this concept in mind, it is possible to construct the iconic representation of the singular value decomposition as depicted in Equation 3.13.

$$X_{m \times n} = U_{m \times m} D_{m \times n} V^T_{n \times n} \tag{3.13}$$

This representation matrix U comprises hierarchically ordered columns called left singular vectors. These correspond to the singular values in the diagonal matrix D and the right singular vectors that combine into matrix V. The ordering of these elements relates to their ability to represent the variance contained in the original data matrix X. Three characteristics of the singular value decomposition are relevant to this work.

The complete matrix X can be reconstructed with the product of the SVD. However, when the matrices in the SVD are truncated, i.e., only $r \leq n$ columns or rows from the respective matrices are taken, the product will compute an approximation of the original data matrix. Because the elements are ordered hierarchically

regarding the amount of variability in X that they correspond to, this will remove some amounts of variance depending on the value of r. It will also create the best approximation of rank r for X, stated in the Eckard-Young theorem [44, 64].

It is also possible to set any of the column vectors of matrix U to zero and effectively remove the associated variability in the reconstructed matrix.

Finally, the principal components analysis can be thought of as the statistical interpretation of SVD. For example, when computing the decomposition of the mean-centered matrix B, the principal components $T = BV$ are the same as $T = UD$. The implementation of 'prcomp' in R uses SVD to compute the principal components.

3.1.3 Assessment

The previously outlined metrics and procedures form the basis for ascertaining and quantifying variation in a dataset. However, they are not suitable for distinguishing between the biological variation associated with a factor of interest and unwanted variation resulting from other factors. For example, when investigating the presence of BEs or the success of batch effect correction methods, these metrics do not allow quantifying whether the biological signal was removed as well. This limitation means that these methods can provide strong evidence for the failure of a method but only weak evidence for its success, i.e., in the sense that some amount of variation has been removed.

Therefore more sophisticated approaches that can quantify and distinguish between wanted and unwanted variation are required to evaluate the quality of batch effect correction procedures. In statistics, the coefficient of determination denoted R^2 ("R squared") is the proportion of the variation in the dependent variable that is predictable from the independent variable.

ANOVA

Analysis Of Variance (ANOVA or AOV) is a linear statistical model that allows estimating the amount of variability attributed to known batch effects. The categorical covariates that are incorporated into the model are denoted as factors and their categories as groups or levels. The number of factors used denominates the type of analysis, e.g., a 1-way ANOVA uses a single factor, two factors make a 2-way ANOVA, and so on. For a given model, the approach calculates the ratio of

between-group variability (BGV) and within-group variability (WGV) to generate the F-statistic that facilitates calculating significance values for the factor levels. The employed numerical metric is the sum of squares (Equation 3.14) that will sum the squared distances of each data-point from the groups' mean value $\hat{y}_{group}$ for WGV and similarly calculate the distance to the data-mean $\hat{y}_{data}$ for the BGV. Because this method also accounts for the residual variance, i.e., the remaining variability in the data that can not be attributed to any of the variables, the approach enables to compute the ratio of variation that can be attributed to the used factors by scaling each value by the total amount of variation present.

$$SS = \sum_{i=1}^{n} (y_i - \hat{y})^2 \tag{3.14}$$

If a batch factor is known, this concept makes it possible to create a linear model of all relevant factors and estimate the percentage of variance attributable to batch processing under the given model. For a linear model, this means fitting the factors of interest as fixed additive effects and calculating the explained variance ratio for each factor. This method works particularly well if the groups implied by the batch factor are similar in size so that the estimates are reliably comparable. Contrary, if the batch factor contains multiple levels (≥ 3), i.e., resulting in a smaller number of observations per level, or the observations are disproportionately distributed between levels, the variability estimates can become unreliable when calculated by pooling the levels respectively [66]. A variance estimate based on two data points is potentially not a good representation for a group as it is either an over or under-representation of the actual value. In this situation, it can be helpful to employ a linear mixed model that treats the batch factor as a random effect. In contrast to fixed effects, the assumption of independence between the different levels of a factor is now replaced by presuming that although the levels have distinct effects, they are drawn from the same distribution of levels. This presumption enables a partial pooling or shrinkage approach that combines the mean values for respective levels with the overall mean for the factors. Levels that only contain a small number of observations are affected to a more considerable degree by the overall mean, whereas more extensive groups are less affected. These smaller groups are effectively shrunken towards the factors mean by lending some information from the whole, which is possible due to the assumption that the random effect has a specific but similar impact for every level.

Because the random effects overall variance is calculated as the distance between the level means to the factor means, as opposed to calculating the sum of squares for every single observation, the golden rule is to have at least 5 distinct levels in a random effect factor to produce reliable results [66].

In summary, the linear modeling approach can be used to calculate the proportion of variance that can be attributed to covariates of interest. Based on the number of distinct covariate levels and the distribution of observations on these levels, employing a mixed-effects model that considers some factors as random effects can be beneficial.

Redundancy Analysis

The Redundancy Analysis (RDA), also called PCA with instrumental variables [161], can be used to assess the amount of variation in a set of response variables Y that can be explained by a corresponding set of explanatory variables X. As an extension of multiple linear regression (MLR), the approach performs multiple linear regressions for every response variable y in Y, i.e., for every feature, a model will be fitted to the explanatory variables that can be described as $\hat{y} = \beta * X$. These vectors can be combined into a single fitted matrix $\hat{Y}$ that effectively describes the relationship between the features in Y and the covariates in X, i.e., the amount of variability that can be explained is constrained by the explanatory variables. The remaining unexplained variation is contained in $Y_{res} = Y - \hat{Y}$. The second step comprises principal components analysis on $\hat{Y}$ to reduce dimensionality and procure an ordination in the space of the explanatory variables.

An extension to this approach is the partial RDA. The basic concept is to remove, i.e., partial out, the effects of one or more explanatory variables from the data to retrieve an adjusted estimate for the explained variability for the factors of interest. The procedure compares the full model $Y\tilde{X} + W$ with the matrix W containing explanatory variables to be accounted for to the reduced model $Y\tilde{W}$ and considers the difference as the model $Y\tilde{X}$ cleansed from the impact of W. This method can be utilized to extract variance components for specific variables of interest, i.e., variance partitioning. However, if the variables in X and W interact with each other, i.e., they are strongly correlated, this will bias the results and might produce untrustworthy variance components.

Principal Variance Component Analysis

Principal Variance Components Analysis can be used to quantify batch effects. The hybrid approach incorporates PCA and variance components analysis to assess the magnitude of variability that each source introduces. Therefore it is a valuable tool both prior to, and post-batch-correction procedure [88].

The basic procedure uses PCA to reduce the abundance matrix Y dimensionality while preserving most of the contained variability. In order to compute the amount of variability in the expression matrix that can be attributed to factors of interest, e.g., group and batch, the procedure will fit mixed models to each PC respectively and extract the variance components for each factor. The eigenvalues of the PCs scale these variance components the model was fitted to, i.e., the eigenvalues denote how much their respective principal components account for variability in Y, so they serve as weights for the associated variance components. Finally, a ratio of explained variance can be obtained by averaging the respective variance components.

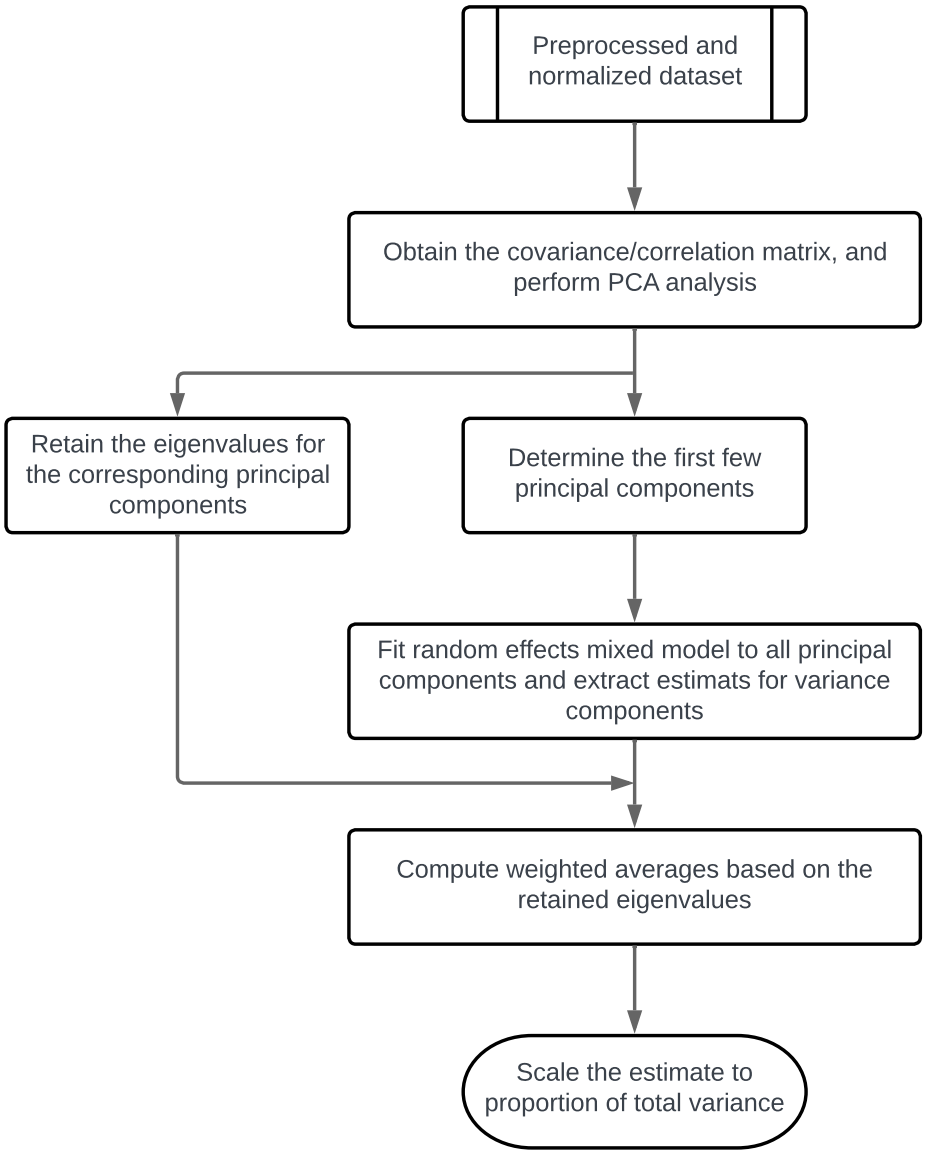


Figure 3.1: The PVCA workflow diagram depicts the computational steps required to estimate variance components via principal variance components analysis. The figure is a reproduction from the book "Batch Effect and Experimental Noise in Microarray Studies" [88].

Silhouette Coefficient

The Silhouette Coefficient (SC) is a metric that allows estimating how well a clustering technique has performed [75]. For n data points that have been assigned to k clusters by, for example, a k-means clustering approach [67, 90], the SC provides a single numerical qualitative score in the interval $I \in [-1, 1]$. A positive value close to one indicates a good fit of data points to the cluster, and negative values mean that the data points have been assigned to the wrong cluster. Coefficient values close to zero imply overlapping clusters.

The computation of silhouette scores requires two metrics, the average within-cluster distance shown in equation 3.15 that calculates the average distance of a datapoint i to all other points in its assigned cluster C_I.

$$a(i) = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i, j) \tag{3.15}$$

Equation 3.16 shows the between cluster distance for data point i, i.e., the dissimilarity to other clusters in the data is defined as the distance to the center of the nearest cluster other than C_I.

$$b(i) = min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d(i, j) \tag{3.16}$$

These two equations can be used to compute the silhouette score for any given point i as shown in Equation 3.17.

$$s(i) = \begin{cases} 1 - a(i)/b(i) & \text{if } a(i) < b(i) \\ 0 & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1 & \text{if } a(i) > b(i) \end{cases} \tag{3.17}$$

Thus, the average silhouette score in a cluster indicates the goodness of fit for this particular cluster, and taking the average scores for all clusters provides a metric that helps assess the goodness of fit for the whole data set.

This approach can also be utilized to assess the relationship between independent and explanatory variables. The assumption is that categorical factors of interest, e.g., batch, treatment, or sex, are responsible for some amount of variability in the data that can be observed in similar responses in samples that belong to the same

category. Hence, each factor and its levels can be considered as clusters assigned to the data, and silhouette scores can be computed to assess how well this clustering fits the data.

3.2 Batch Effect Correcting Algorithms

The algorithms explained in the subsequent sections are divided into two categories, Assessment and Correction. The algorithms in the former category are used to estimate the significance of known batch effects for every feature in the data set and provide significance values for downstream analyses. The latter group will compute the impact of BEs and correct the variance directly within the abundance matrices.

3.2.1 BECA LM

The previously outlined concept for modeling a data set describes a linear modeling approach for microbiome data. In linear models, the response variable, e.g., feature abundance, is described as a linear combination of predictor variables, e.g., study grouping and batch. Let X contain the factor of interest denoted by *group*, i.e., study grouping in terms of case and control group membership, and a *batch* vector that denotes affiliation of samples to processing batches. The resulting naive linear model is represented in Equation 3.18.

$$Y_i = \beta_0 + \beta_{group} X_{i,group} + \beta_{batch} X_{i,batch} + \varepsilon_i \tag{3.18}$$

Where β_0 denotes the intercept of the model. It can be seen as the mean value, i.e., the average expression value for a particular gene, that would occur if there is no input from the independent variables. The coefficient β_{group} in this example quantifies the impact of the factor of interest, i.e., the study grouping that, in theory, is connected to the biological effect. In the same way, β_{batch} quantifies the impact of batch-processing on the resulting expression. The error term ε_i essentially absorbs the remaining uncertainty in this model.

This model can now be fitted to every measured gene separately to estimate the coefficients and thereby the magnitude of impact that the modeled factors have on the measured abundance. Every fitted model can undergo hypothesis testing to determine the statistical significance of the factors for each feature. Because this is

done repeatedly for many different features, the resulting values need to be adjusted for multiple testing to prevent exaggeration of significance by testing everything [109]. For this approach to be feasible, the batch factor is required, i.e., a covariate that describes how samples have been processed together in distinct groups needs to be present. If this factor is unknown, it is impossible to attribute any variation to any particular batch.

3.2.2 Surrogate Variable Analysis

Surrogate Variable Analysis is an algorithmic approach to estimating unknown sources of variation from the expression data. The authors [85] of the algorithm denote the challenge of unwanted variation as expression heterogeneity and define its characteristic as affecting subsets of features, i.e., gene expression values, in different ways. This definition is equal to non-systematic batch effects, as introduced previously.

The procedure of SVA is to remove the biological factor of interest from the original data to retrieve the residual matrix that is considered to contain the unwanted variation. A matrix factorization of the residual matrix allows determining the axes, i.e., the signatures of expression heterogeneity, that account for a significant proportion of variability. Because the eigenvectors are constructed as linear combinations of residual gene expression values, it is feasible to calculate the subset of driver genes. For each of the estimated gene subsets, the algorithm will create a new covariate in the original data space. Hence, the subsequent regression analysis can account for the newfound sources of variability and produce cleaned-up statistical values for the biological factors of interest.

3.2.3 Remove Unwanted Variation II

Remove Unwanted variation 2-step (RUV-2) was developed to address the problem of noise in microarray expression studies [54]. This approach does not require the specification of a batch factor, i.e., an observed grouping for processing in batches. Instead, the algorithm requires a set of negative control genes (NCG), i.e., genes that are known to be unaffected by the biological factors of interest, to enable the estimation of unwanted variation. The assumption is that any variation in NCGs can be considered unwanted because it is not related to biological factors. Hence the

algorithm uses these genes to estimate noise and incorporate it into the model. An approach to RUV-2 in the absence of negative control variables is the estimation of pseudo-negative controls. To that end, an L(M)M containing the factor of interest is fitted to each feature, and the significance is calculated. Finally, the features that are not significantly affected by treatment are considered pseudo-negative control variables.

The underlying model of the data is identical to the previously shown model used in SVA. For $|m|$ samples and $|n|$ features, the matrix $Y_{m\times n}$ contains the observed abundances and is modeled as shown in Equation 3.19.

$$Y_{m\times n} = X_{m\times p}\beta_{p\times n} + Z_{m\times q}\gamma_{q\times n} + W_{m\times k}\alpha_{k\times n} + \varepsilon_{m\times n} \tag{3.19}$$

The matrix $X_{m\times p}$ corresponds to $|p|$ observed factors of interest, i.e., one or more biological factors that are to be investigated by the particular study. Further observed covariates or nuisance variables, e.g., sex, age, gender, comprise the optional matrix $Z_{m\times q}$. The unobserved factors that correspond to unwanted variation are contained in matrix $W_{m\times k}$. Finally, $\varepsilon_{m\times n}$ denotes the common term used to account for remaining normally distributed noise.

In order to estimate W, the algorithm uses the established non-association of NGCs and biological factors in X by performing calculations on a subset of the data. Let c denote the number of NGCs in the data and consequently Y_c,β_c,γ_c,α_c, ε_c describe a reduced dataset. Because the NGCs are not associated with the biological factors in X, this means that $\beta_c = 0$, hence the model may be reduced to the following Equation 3.20.

$$Y_c = Z_c\gamma_c + W_c\alpha_c + \varepsilon_c \tag{3.20}$$

The developers of the algorithm take care of eventual covariates in Z_c by multiplying the equation with the residual operator R_Z shown in Equation 3.21. This approach results in Equation 3.22 that allows producing an estimate $\hat{W}$ with factor analysis and substitute for W in the original model, which in turn allows to regress Y and calculate a $\hat{\beta}$ for a model that takes the estimates for unwanted variation into account.

$$R_Z \equiv I - Z(Z'Z)^{-1} \tag{3.21}$$

$$R_Z \equiv I - Z(Z'Z)^{-1} \tag{3.22}$$

The unknown number of latent variation factors, i.e., k-dimension in W, remains to be addressed. In practice, the algorithm should be executed with different choices for k to compare the quality of the output and decide which value to choose. A more detailed explanation of the algorithm can be found in section D of the supplementary material to the corresponding publication [54].

3.2.4 Remove Unwanted Variation IV

Remove Unwanted Variation 4 is a refined version of RUV-2 that addresses two problems with this original approach [54]. For one, the RUV-2 algorithm is sensitive to the choice of k, i.e., the number of unknown factors to estimate, which warrants an iterative approach to evaluate the algorithm's performance for different values of k for a particular dataset.

The new implementation provides a function that can estimate k directly from the data. However, this function is not proven to produce optimal results. For the second challenge, the authors borrow from the SVA algorithm proposed by Leek and Storey [85] and remove the biological factor of interest from the model before starting to estimate the matrix of unwanted residual factors W_0 with factor analysis. Subsequently, the algorithm will employ negative control genes to compute an estimate for the remaining unwanted variation that is confounded with the biological factor of interest and was removed in the first step. According to the authors, the newly proposed algorithm is less sensitive to the choice of k and to violations of the assumptions regarding negative control genes. The last aspect makes the algorithm more applicable to data sets that can only provide pseudo-negative control genes. The output is the same as with RUV-2.

3.2.5 Remove Unwanted Variation III

Remove Unwanted Variation 3 (RUV-3) is an extension of the previously outlined versions RUV-2/4. In contrast to the other implementations, this algorithm will return an abundance matrix that has been corrected for unwanted variation. The procedure requires negative control features, which the developers define as a subset

of features that is not affected by the biological effect of interest, e.g., treatment, disease, or heritage. In the context of microbiome datasets, this requirement can be satisfied by employing well-defined mock communities [26] as part of every batch.

The second requirement is the availability of technical replicates, i.e., a single sample that has been processed twice, as they should, in theory, be the same, and investigation of their dissimilarity helps estimate unwanted variation. Knowledge of batch effects is not a requirement of this algorithm. Instead, the algorithm represents data similar to the previously outlined linear modeling as shown in Equation 3.23.

$$Y_{m\times n} = X_{m\times p}\beta_{p\times n} + W_{m\times k}\alpha_{k\times n} + \varepsilon_{m\times n} \tag{3.23}$$

The difference in the underlying assumptions is that W corresponds to all sources of unwanted variation, but it is unobserved, and k is unknown, as is α. This results in a three-step approach as follows.

First, the residuals between replicate measures can be used to estimate one aspect of unwanted variation. Second, taken in combination with the expression values of negative controls, i.e., supposedly only affected by some degree of variation that is not biological, this serves as the basis to estimate another component of unwanted variation. Third, combine these estimates and subtract them from the original data.

In this approach, the user must specify the number of factors k that should be considered sources of unwanted variation. According to the authors, the only valid strategy to do so is to run multiple analyses with different k and determine the best choice based on the results [102].

3.2.6 Batch Mean Centering

Batch Mean Centering (BMC) calculates the mean values for all features j according to the respective batches k as shown in Equation 3.24 and then adjusts the data by subtracting this value from the respective features (Equation 3.25).

$$\bar{x}_{jk} = \frac{1}{n_k}\sum_{i=1}^{n_k} x_{ijk} \tag{3.24}$$

Consequently, the subgroups corresponding to batches are centered on their respective mean value. This procedure supposedly reduces the variation between different batches, but maintains the intra-batch variation, i.e., the effect of interest.

$$x^*_{ijk} = x_{ijk} - \bar{x}_{jk} \tag{3.25}$$

In the case of balanced group designs, this strategy of zero-centering will eliminate a large proportion of the variability associated with the batch factor while keeping the variability between groups of interest intact. However, for unbalanced designs, the batch effects will be influenced by the uneven distribution of samples, i.e., the group of interest introduces a bias because some predominate the composition of samples within batches. BMC will also lower group differences and reduce statistical power in this situation. The higher the imbalance in the dataset, the higher the chance that the correction method will introduce spurious differences because group and batch factors are too confounded to remove one without affecting the other.

The method has been proven to help integrate NGS gene expression data sets from multiple studies [140]. However, in addition to the short-coming concerning design balance and univariate estimates for variability, this method also presumes uniformly distributed data which is rarely the case for microbiome data [110].

3.2.7 ComBat

The ComBat algorithm was initially developed for microarray expression data and was adopted for other data types due to its ability to handle large feature spaces in studies of small sample size [73]. The procedure employs a maximum marginal likelihood estimator, i.e., Empirical Bayes, that leverages the information of the complete dataset to shrink the parameter estimate for the batch factor in the direction of the overall estimated mean. This method specifically incorporates systematic batch effects, as the underlying assumption is that all features are affected similarly.

3.2.8 Remove Batch Effects

This method was designed to remove BEs from microarray data as part of the `limma` package. The algorithm fits the full model to the data, i.e., all relevant covariates whose effect should not be removed and a model containing only the known BEs. The difference between these models produces a residual matrix that presumably contains only the full-model-effect, e.g., treatment. As of now, the mbecs-correction only uses the first input for batch-effect grouping.

3.2.9 Percentile Normalization

This method was developed specifically to facilitate the integration of microbiome data from different studies/experimental set-ups. This problem is similar to the mitigation of BEs, i.e., when collectively analyzing two or more data sets, every study is effectively a batch on its own. The algorithm iterates over the unique batches and converts the relative abundance of control samples into their percentiles. The relative abundance of case samples within the respective batches is then transformed into percentiles of the associated control distribution. The procedure assumes that the control group is unaffected by any effect of interest, e.g., treatment or sickness, but both groups within a batch are affected by that BE. The switch to percentiles flattens the effective difference in count values due to batch compared to the other batches. This also introduces the two limiting aspects of percentile normalization. It can only be applied to case/control designs because it requires a reference group. In addition, the transformation into percentiles removes information from the data.

3.2.10 Singular Value Decomposition

The underlying assumption of employing SVD to remove batch effects is that they present the most prominent form of variability in the data set. This implies that the first column of the left singular eigenvector matrix U represents the variance in the original data matrix that can be attributed to sources of unwanted variation. Hence, the algorithm computes the decomposition and deflates this vector before reconstructing the cleaned-up matrix. The method does not require knowledge of existing batch effects; it just implements the underlying assumption. It is, however, liable to remove biological variation if not used carefully. In particular, data sets that exhibit minimal technical variation can be negatively affected by this procedure.

3.3 R Framework

R is a framework for statistical computations and analyses that facilitates the fast and coherent translation of ideas into usable software [123]. The framework is composed of the programming language R, a runtime environment, debugging functionality, and the ability to access the functionality of the underlying operating system. The interpreted language does not require a compiler to run. R is the consolidation

of features from two programming languages, S by John Chambers and Scheme by Gerald Sussman [71]. While the resulting language is very similar in appearance and syntax to S, the underlying semantics is based on Scheme.

The name is derived from its creators, Ross Ihaka and Robert Gentleman, who started the project in 1991 and publicized the first version in 1993. The framework is written in C, Fortran, and R itself. It is possible to incorporate a variety of programming languages and execute them at runtime, i.e., extensions may be written in C, C++, and Fortran; furthermore, Java, Python, or .Net can access and manipulate R-objects.

R is no longer developed and maintained by its creators. In 1997 the R Core Group was formed. This group of people has access to the R source code and is responsible for the advancement, documentation, and bug fixes in R. Many contributions, e.g., donating code, helping in creating ports to other systems, or providing expertise in planning further developments, have been made by community members outside the Core Group. A point of access for these individuals or institutions is the R-Foundation, a non-profit organization that members of the core group founded. In addition, the foundation holds the copyright for R and supports innovations in the framework and the field of statistical computing in general. With RStudio, a free and open-source integrated development environment (IDE) is also provided. Over time, the language and RStudio developed and got more functional.

Today, R is widely used, and the fast adaptation in the scientific community, finance applications, and production environments can be attributed to several advantageous qualities of the framework. In contrast to S-Plus, R has been distributed under a GNU license and free to use since 1995. Since it is an interpreted language, R does not require a compiler and is cross-platform compatible, i.e., scripts written on a Windows system can be transferred and used on Unix-like systems without extensive modification of either code or programming environment.

Packages

An additional contributor to its success is the availability of packages that provide a wide variety of single functions, algorithms, and complete workflows to the users. They facilitate the ability to share functionality and data with the respective community within a standardized environment, i.e., package structure needs to follow

specific requirements towards documentation along with directory and code structure. This requirement ensures qualitative implementations and the longevity of the packages.

The elementary functionality of R that enables the essential programming support, IO functionality, arithmetic, and statistical computations is contained in the base packages, e.g., `methods`, `stats`, and `utils`. This group of packages is distributed along with the R installation, and its functions are available in any R environment. Because R is constantly maintained and further developed, well-implemented algorithms ubiquitously used by the community can be incorporated into the base packages in new versions.

Specialized software repositories provide access to additional packages. These centralized and regulated systems can store and distribute software and additional information, e.g., dependencies, Readme, man-pages, and enforce standards for implementation. They are commonly controlled by Package-/Repository-Managers that keep track of correct software versions and allow users to install or update software. The official repository for R is the Comprehensive R Archive Network (CRAN), which the developers initiated. Over 10,000 packages are stored and maintained here, and the R core developer group hosts it. Installing packages from this source is one of R's base functionalities, and it is usually the point of access for new users. In turn, this fosters a vast community, very active in providing and improving packages, i.e., packages provide functionality from data wrangling over graphical representations to the implementations of established and newly developed algorithms.

A variety of other public and private repositories exist. It is, for example, possible to set up local servers in a production environment or intranet to limit access to a curated list of packages. In the context of this work, of particular interest is the Bioconductor open-source network focused on bioinformatics and omics-data analyses. GitHub is not strictly a package repository for R but rather a provider of project hosting and distributed version control functionality for every conceivable programming language. It is a commonly used platform for collaborative software development projects. Additionally, GitHub can be used as a source for implementations in R, i.e., single functions, workflows, and whole R packages can be made available to end-users and complemented with additional explanations and tutorials with github.io web-hosting.

Programming Essentials

This work is based on the object-oriented programming (OOP) paradigm that centers software design on the representation of data rather than the implementation of functions and logic. For example, an object can contain fields for data and is commonly associated with one or more functions that perform computational operations on these fields. The fields and functions that are part of an object are specified in a so-called class that serves as a blueprint for creating data objects. In the context of this work, a class would define the data fields and operations that a required to store and process the information that is part of a microbiome experiment.

Box 3.3.1 Function Example

```
#' Calculate Mean of two variables
#'
#' For two input parameters a and b, this function will
#' return the average of both values.
#'
#' @param a A numerical value.
#' @param b A numerical value.
#' @return mean The average of input parameters a and b.
#' @export
calculateMean <- function(a, b) {

mean <- (a + b) / 2

return(mean)
}
```

Codebox 1 shows an example for the function `calculateMean(a, b)` that calculates and returns the average of two input parameters a and b. The example shows the three blocks common in the declaration of a function. First is the description of the function that explains what it will do and also employs tags to specify input parameters (@param), what value will be returned by the function (@return), and if this function will be visible to a potential user (@export). The function header, `calculateMean <- function(a, b)` defines the name of the function, i.e., what

needs to be written in order to execute it, as well as the name of the input parameters for the function body. The body contains code that will be executed once the function is called.

3.4 Data-sets

Three microbiome datasets are utilized in the subsequent chapter to outline the functionality of the created software package.

Sponge Data

The sponge species *Aplysina aerophoba* produces varying amounts of brominated alkaloids through biosynthesis in its' tissues [130]. Sacristán-Soriano et al. investigated the bacterial community composition in ectosome and choanosome tissues to determine a possible association with the produced amount. They targeted the 16S rRNA gene in two types of gel in denaturing gradient gel electrophoresis, which constitutes two batches. The authors acknowledge a batch effect between both gel-types in the publication, with a significance value of $p = 0.004$ determined via PERMANOVA. The 32 samples are distributed evenly across the sampled tissues and over both batches, and the dataset contains 24 operational taxonomic units. The batch effect is apparent in this data; hence it is used to visualize the package workflow in Chapter 4.

Mouse Data

The mouse dataset was produced specifically for this project by incorporating a deliberate batch factor into the sequencing procedure. The dataset comprises nine unique stool samples of four female and five male B6 mice that were replicated and sequenced at two different laboratories. The samples were taken at the animal facilities in Lübeck, and the material was distributed to the sequencing facilities located in Plön and Lübeck. Both laboratories employed the same protocols and created 16S amplicon datasets of the targetted V3 and V4 regions. After preprocessing and filtering steps as outlined in Section 3.4, the data includes 651 amplicon sequence variants in 18 samples. For the purpose of this work, the sampling locations will be

considered the batch factor, and the sex difference will serve as a biological factor of interest.

Dog Data

The dog dataset comprises 8,390 taxa in 588 amplicon sequenced 16S samples. The samples were taken from 46 dogs and included stool as well as 12 distinct locations on the skin. The skin samples were processed in three batches, and the stool samples were processed in an additional batch. This data set was kindly provided by Dr. Axel Kuenstner and is the basis of an actual publication. The batches result from the large number of processed samples that had to be sequenced in separate runs.

4 Microbiome Batch Effect Correction Suite

The Microbiome Batch Effect Correction Suite (`MBECS`) was conceptualized as a software toolbox that enables batch effect correction as well as the evaluation of the success of correction efforts. The guiding principle for creating the software was to make it as accessible as possible, which imposed several requirements. First of all, it should provide access to a variety of BECAs. Secondly, the assessment of correction success should be integrated into the package's workflow. The last requirement concerns the ease of use facilitated by the convenient import and export of data and largely automated procedures.

The subsequent sections will start with an outline of the general package development procedure, followed by a description of incorporated software packages and their function. This information prepares the delineation of the package's internal structure and functioning before providing an example of the implemented workflow. Finally, the chapter closes with an application of the `MBECS` package to the previously introduced datasets.

4.1 Implementation

The `MBECS` project relies on multiple other packages that provide practical means during the development process or facilitate the creation of the package's features. In order to address the particular requirements and the packages that were chosen to provide them, this section is split into five categories. The first category is concerned with the general development procedure and the utilitarian packages that deal with the organizational overhead. The second category introduces the software tools essential for accessing and organizing the data within the package. Thirdly, the computational packages provide single methods or full implementations of concepts

required to assess and correct batch effects. The packages introduced in category four enable the representation of results in plots and reports. Lastly, the makeup of the `MBECS` package will be addressed.

Utility

The elementary steps that were performed in creating the `MBECS` package are depicted in Figure 4.1. They coincide with the general development process for R packages that was chosen as the guideline for approaching this project. This diagram does not account for the organizational effort required to properly manage a package, e.g., generating files and folders and keeping track of all utilized packages. Serendipitously, this organizational overhead is largely taken care of by the utilitarian packages developed to facilitate fast and efficient development.

The `devtools` package provides a collection of functions that allow creating, testing, and interacting with packages in development. It implements the ability to install packages from their GitHub sources or selectively load local packages for testing. Additionally, it offers the `check()` function that performs a suite of tests to validate packages according to the guidelines of the CRAN repository [154].

After the initial setup, the project was configured to use GitHub as a version control system via the interface that RStudio provides. The significant advantage of employing GitHub in this work was the function as an online backup that facilitated the restoration of a working project state in case of a local failure, if a programming mistake made the software unusable, or if code was lost due to hardware failure. Additionally, the online repository facilitated the continued work on a different workstation without the necessity to employ a tedious and error-prone copying process. The inherent benefit of enabling collaborative work is mute in this case because the project was limited to a single developer.

The two packages `utils` and `methods` provide the required tools to design a data structure in R. The `methods` package is part of base R [123] and provides an interface that allows to set up a user-defined class system. The `utils` package offers access to quality of life functions that facilitate development and data handling. It can be used to investigate and configure local packages and variables, e.g., the `head()` function will show the first few lines of a data table and `showMethods("packageName")` can list all available functions in a package. In addition, it provides an interface that

allows the use of progress bars for user-defined functions. The `utils` package is part of base R [123].

After the conclusion of the project setup and design phase, the implementation of the program's logic began. This stage constituted the most considerable effort as every available feature constitutes a single function within the package. The diagram in Figure 4.1 shows that this is a repetitive stage that is comprised of the three steps implementation, documentation, and testing.

In order to implement a new feature, it is broken down into the constituent steps that are then written into code. For example, to create a function that can transform feature abundances into total sum-scaled ratios, it is necessary to consider three major steps. First, retrieve the correct abundance matrix from the input. Second, for every sample in the matrix, divide every feature by the sum of all features for that sample. And third, return the newly created matrix of ratios. These steps are partitioned further until a set of appropriate atomic operations can be written into a function body. During implementation, the recommended working routine is to work consecutively, i.e., complete implementing one feature after another instead of processing multiple issues at once. It is also expedient to conduct documentation and implement tests during this process.

Documentation of the functions is meant to provide users with an outline of how to use the interface, i.e., what are the inputs and parameters that can be controlled and changed. The `roxygen2` package provides a convenient syntax that facilitates the documentation of functions during the process of implementation. Upon completion of the development process, these annotations are translated into a manual that lists and describes all available functions of a software package. These so-called man-pages can be reviewed on the Bioconductor page for the MBECS package (`https://bioconductor.org/packages/release/bioc/manuals/MBECS/man/MBECS.pdf` or in the supplementary material of this work.

It is necessary to ensure the correct mode of operation for every implemented function. That means to test if it accepts the correct input values and creates the expected output and if the function behaves as expected in case of an error. During implementation, this is generally done by the programmer in a trial-and-error fashion. While manual testing has its advantages, it is also error-prone concerning consistent repeatability and the assessment of computational output. These limitations do not apply to automated testing procedures that transfer the human-driven

task of evaluating software into a repeatable automated process. Unit testing refers to an automated procedure that can be developed parallel to implementing the code and used to test correct workings while adding more functionality to the software. This proceeding is a major benefit to developers because it allows the evaluation of code and the process of writing it. For example, often, it becomes necessary to adjust the design of a class or a function due to changes in the design of the software, i.e., the input gathered in manual testing may warrant the implementation of additional functionality. This can profoundly impact the inner workings of the software and change critical factors such as data formats or produced outputs in ways that are not necessarily obvious to the developer. In these cases, the Unit tests will indicate when the newly implemented functions will deviate from the expected behavior and facilitate the adjustment.

The `testthat` package facilitates unit testing to evaluate the correct workings of the implemented functions and class system. It is designed to be used in parallel to the ongoing development process and integrates seamlessly with the quality control function offered by CRAN and Bioconductor [152].

The package's correct function and adherence to a particular software repository guidelines were tested in the final step. Testing is integral to the software development process because it is used to ensure error-free execution, the correctness of the output, and the overall quality of the final product. The task can be separated into two conceptual parts, manual and automated software testing. In manual software testing, the application is evaluated by one or more individuals. The tester will execute single functions and the workflows that are provided to ensure, first of all, that they function correctly. Therefore, it is reasonable for the tester to be familiar with the subject of the developed software, i.e., in the case of analyses of biological samples in a script-like framework such as R, it is sensible to employ a data analyst with intimate knowledge of the subject. This process can also serve to evaluate the quality in terms of accessibility, functionality, and appeal.

BioConductor The package development tools required for the Bioconductor repository are grouped into three packages. `BiocManager` provides the package management interface that allows to install software from Bioconductor [103]. `BiocStyle` offers formatting style for package documentation that complies with the strict requirements imposed by Bioconductor [112].

Table 4.1: The table lists the utility packages that were used during software development.

Package	Description	Version	Repository	Reference
methods	Methods, Classes, and Functions	4.1.2	base	[123]
utils	Development Utility Functions	4.1.2	base	[123]
devtools	Collection of package development tools.	2.4.3	CRAN	[154]
usethis	Automate package and project setup tasks.	2.1.5	CRAN	[157]
testthat	Unit Testing for R	3.1.3	CRAN	[152]
roxygen2	In-Line Documentation for R	7.1.2	CRAN	[156]
BiocManager	Biuoconductor Repository Access	1.30.16	CRAN	[103]
BiocCheck	Bioconductor-specific package checks	1.30.0	Bioconductor	[113]
BiocStyle	Document formatting styles	2.22.0	Bioconductor	[112]

Finally, the `BiocCheck` package performs quality control and compliance checks that are more extensive than for the CRAN repository [113]. The Bioconductor repository requires contributors to adhere to a host of package requirements concerning how other software is referenced, how the packages' code is formatted to improve readability, the completeness of documentation and unit tests, or the adequate description of included datasets. The `BiocCheck` function will evaluate a software package with regards to these requirements and return a report that outlines the package's shortcomings to the developer for correction. The guidelines for package submission, development, and maintenance can be reviewed on the Bioconductor website under `https://bioconductor.org/developers/`.

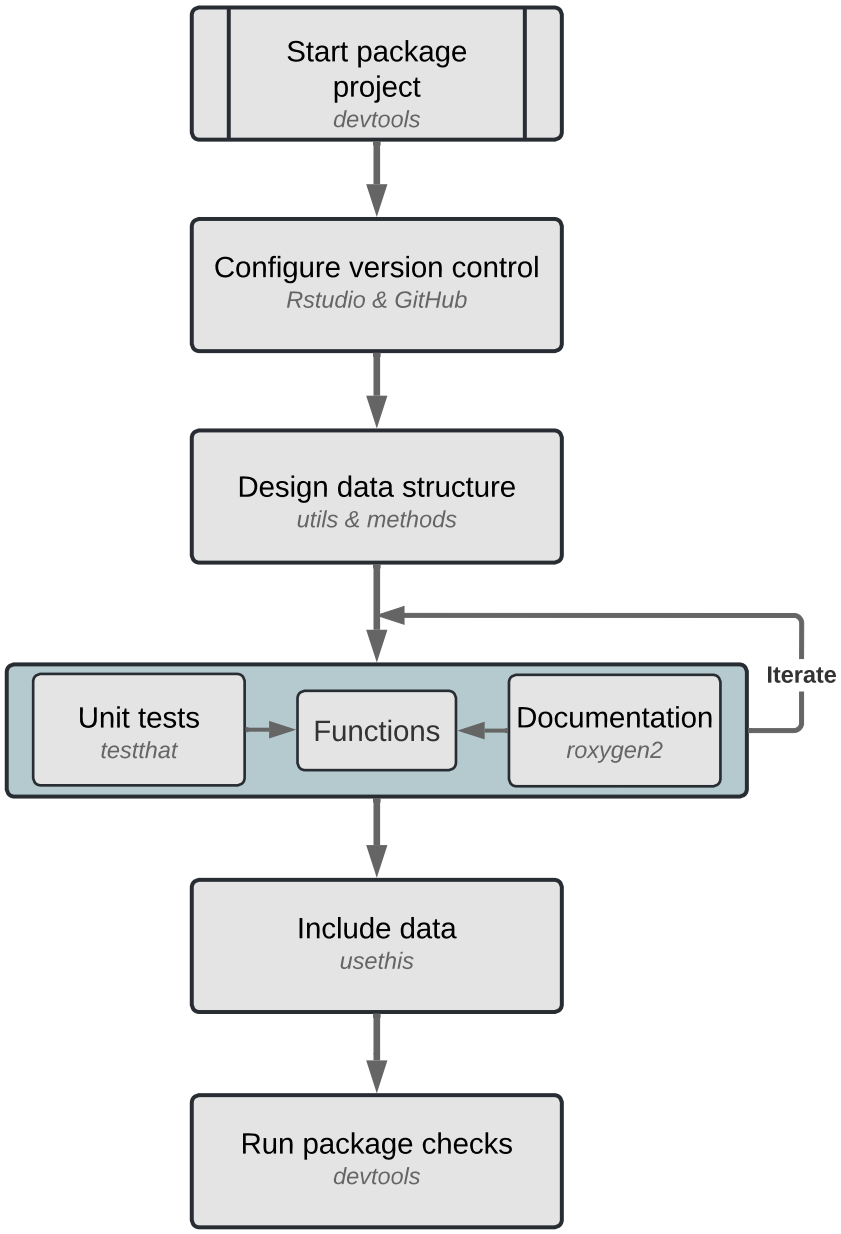


Figure 4.1: Basic steps in R package development procedure and associated tools and software resources.

Data Handling

The procedures implemented in the `MBECS` package need to access, transform and manipulate a dataset, e.g., to calculate average abundance values for all samples and create a bar plot that is colored by affiliation to batches or to perform a batch correction and store the results as part of the data object. Although they are distributed in four different software packages, the subsequently outlined tools complement each other and form a framework for data manipulation and cleaning. For example, the `tibble` package [105] provides a more rigid data structure than the `data.frame` class that is native to R.

`Tibble` implements strict requirements for successful data transformations, which enforces more structured procedures when working with them, and this, in turn, reduces erroneous behavior and thus increases the maintainability and longevity of software.

The `magrittr` package provides the forward-pipe operator to R [10]. This operator facilitates intuitive code writing as it evaluates the output values of an expression and can forward them to the next operation. Writing code is essentially an activity that requires identifying the type and order of tasks required to transform an input into the desired output. Thus, the forward-pipe operator makes it very easy to chain these tasks together in an organized way. This functionality reduces the time spent on developing and implementing functionality. Moreover, it increases code comprehensibility and, consequently, the maintainability of software projects. As for R version 4.1, the pipe operator is also available in base R.

Table 4.2: The table lists the employed packages that facilitate the manipulation and formatting of data.

Package	Description	Version	Repository	Reference
phyloseq	Analysis of microbiome census data	1.38.0	Bioconductor	[98]
tibble	Simple Data Frames	3.1.5	CRAN	[105]
magrittr	Forward-Pipe Operator for R	2.0.1	CRAN	[10]
dplyr	A Grammar of Data Manipulation	1.0.7	CRAN	[155]
tidyr	Tidy Messy Data	1.1.4	CRAN	[153]

Especially covariate information is rarely available in formats that facilitate out-of-the-box analyses. These tables can exhibit missing data points, i.e., NA values that have not been collected or result from dropouts in a study and need to be filtered out. Maybe the desired analysis requires constructing a new variable based on the values of an existing one, e.g., for statistical analyses, it can be helpful to create a binned age variable that allows performing calculations on age ranges instead of continuous values. Alternatively, it may be desirable to reorder a table by some criterion, e.g., perform a grouping by sex prior to creating a heatmap. These tasks are achievable with the R language but generally require long and unwieldy syntax that can not be conveniently applied to various tasks of the same type. The `dplyr` package provides a consistent set of functions that permit and streamline this type of data manipulation. The `tidyr` [159] package is concerned with manipulating the shape and hierarchical structure of a data set. Hence it offers an interface that complements the content-driven functionality of dplyr. The `tidyverse` package comprises, among others, the packages `tidyr`, `dplyr`, `magrittr`, and `ggplot2` [158].

The `phyloseq` class provided by the software package of the same name allows the storage of clustered features, metadata, and phylogenetic information [98]. It is a commonly used package for microbiome analysis within the R framework and was thus chosen as the foundation for the implementation of the `MBECS` data structure.

Statistics

Table 4.3: The table lists the software packages that provide the means for statistical computations and the incorporated batch effect correcting algorithms.

Package	Description	Version	Repository	Reference
stats	R statistical functions	4.1.2	Base	[123]
cluster	Cluster Analysis Extended	2.1.2	CRAN	[93]
limma	Linear Models for Microarray Data	3.50.0	Bioconductor	[127]
lme4	Linear Mixed Effects Model	1.1.27.1	CRAN	[11]
lmerTest	Tests in Linear Mixed Effects Models	3.1.3	CRAN	[78]
permute	Generating Restricted Permutations of Data	0.9.5	CRAN	[139]
ruv	Remove Unwanted Variation using Negative Controls	0.9.7.1	CRAN	[102]
sva	Surrogate Variable Analysis	3.42.0	Bioconductor	[86]

The packages grouped in this section provide the means to build and evaluate statistical models or implement specific computational approaches. For example, the `stats` package is part of the R core distribution that provides a host of methods for statistical evaluation, such as analysis of variance (ANOVA), p-value adjustment for multiple comparisons, or one and two-sample t-tests. In addition, this package comprises random generators for utilitarian distributions and visualization functionality for statistical models.

The `cluster` package implements the computation of the silhouette coefficient that facilitates assessing how well a set of data points fits the grouping described by a common factor [93]. Another valuable collection of functions is the `permute` package that can be used in the construction of a complex statistical model or to provide the possible permutations of any given vector.

The linear model for microarray analysis (`limma`) package is a collection of methods that provide the ability to investigate gene expression micro-array data using linear models and assess differential expression [127]. Due to Bayesian methods, complex designs can be analyzed while providing stable results for small sample sizes. For the `MBECS` package, `limma` provides the `removebachteffects` method, which fits a linear model comprised of batch and an arbitrary number of covariate effects to the data and aims to remove the influence of the known batch effects.

The `lme4` package incorporates the C++ Eigen library and provides an S4 class in R that facilitates the application of linear and linear mixed models to data sets. The `lmerTest` package provides standard statistical analysis methods to evaluate the significance of the fitted designs. To that end, it implements Satterthwaite's degree of freedom and the Kenward-Roger method to provide ANOVA tables and significance values that are otherwise unavailable from `limma`, as well as model selection approaches.

The `ruv` package implements the Remove Unwanted Variation algorithms designed to adjust for unknown sources of variation such as systematic errors in high dimensional data sets [102]. Due to the historical development of such methods, they have been developed for microarray data but are nowadays commonly applied to all types of high dimensional data. The package comprises, among others, the RUV-2, RUV-3, and RUV-4 algorithms, as well as the supporting functions required for stringent execution.

Surrogate Variable Analysis is comprised in the `sva` package and can be used to identify and mitigate sources of unwanted variation in experimental data. The package also contains the ComBat algorithm, a Bayesian approach to removing batch effects.

Graphics and Representation

The packages introduced in this section form the basis for the graphical representation and the reporting function that the microbiome batch effect correction suite provides. One of the most widely used packages is `ggplot2`, implemented by Hadley Wickham [151]. It has existed for more than ten years and has been used to produce millions of graphics. Following the `ggplot2` concept, graphics consist of three key components, the data to be displayed, aesthetic choices, and plot geometry. The category of visual representation, i.e., bar plot, density plot, or histogram, are defined in geometry, while the color and size of annotations or points can be controlled with aesthetic options. The package provides a declarative syntax based on "The Grammar of Graphics" that allows the user to build a visual representation of provided data from a wide variety of graphical primitives. It can be applied to represent data points in almost any conceivable form by declaring aesthetic mapping and annotation and selecting the form of representation.

The low-level graphical functionality that enables the design and structuring of plots in R is contained in the grid package. The package began as an add-on that gave the user more flexibility in managing the layout of produced graphics. Since then, it has become a part of the base R distribution, meaning that it will be installed by default when installing R. The package provides the graphical primitives that form the basis of `ggplot2` and offers classes and functions that enable accessing and representing a graphical object [123]. Additionally, the extension `gridExtra` provides a user-level interface that allows convenient access to grid functionality and enable the arrangement of multiple plot object with a single page as well as drawing of table objects [9].

Although heatmaps could be created by utilizing the `ggplot2` functionality, `pheatmap` provides a convenient wrapping method that integrates matrix display, clustering methods, and depiction of clustering trees into a single function [76].

RStudio received an integration of markdown code, which enables the ability to produce standardized reports that integrate analysis results on the fly, provides the ability to create and use bibliographies for references, and other LaTeX features such as formula notation. For example, an interface for the software version control system Git is part of R, and RStudio offers a graphical user interface for this connection to the user.

Historically, markup refers to the revision process of manuscripts in which reviewers and editors augment the initial content of a document with comments and suggestions for changes and adjustments. In this sense, markup is a syntax that facilitates defining a document's appearance and embedded content in a programmatical way. Much like a programming language, a markup language refers to a set of rules in a specific syntax that allows the definition of extended functionality, are processed automatically and result in a specified output that does not contain the actual language but a combination of plain written text and augmented content, such as graphs, pictures or formatted text. Widely known examples of markup languages are HTML, LaTeX, and XML. They exhibit the main characteristic of a markup language in that they allow to interweave the regular content of a document with commands of the markup language, i.e., plain written text and instructions to segment the text into sections or include graphics at a specific position. Other than, for example, a WORD document, markup files need to be processed to produce the desired output.

`Pandoc` is a Haskell library that can convert documents between various markup formats. It offers no graphical user interface and can only be accessed via the command line. It is advertised as the swiss-army-knife of markup conversion by its developer John McFarlane. It is, for example, able to understand LaTeX style mathematical syntax and provides rendering functions that allow correct conversion into HTML. It also includes capabilities for automated conversion of citations and their respective bibliographies. Although it is not technically a part of the R language, it is distributed with R as it provides the basis for its rendering capabilities. Pandoc enables the creation of interactive HTML documents or LaTeX style pdf documents from the same markdown code.

Markdown for R is an extension of the markdown syntax that facilitates the embedding of R code blocks into a markdown formatted document. These code segments will be executed upon conversion, and the resulting output, e.g., the outcome

Table 4.4: The table lists the software packages that enable the visual representation of results and the generation of report documents.

Package	Description	Version	Repository	Reference
ggplot2	Data Visualisation Using the Grammar of Graphics	3.3.5	CRAN	[150]
gridExtra	Miscellaneous Functions for "Grid" Graphics	2.3	CRAN	[9]
pheatmap	Pretty Heatmaps	1.0.12	CRAN	[76]
rmarkdown	Analyze. Share. Reproduce.	2.11	CRAN	[6, 165, 166]
knitr	Dynamic report generation	1.38	CRAN	[162–164]

of computation or a finished plot, will be integrated into the finished document. The `rmarkdown` package is the library that provides the RMarkdown functionality. It makes use of Pandoc's ability to convert between different formats and thus possible output file types, includes the syntax and rule set defined by the RMarkdown language, and provides the user interface to access these functionalities within R. Whereas the file extension denotes markdown files *.md the rmarkdown files have the *.Rmd extension. Because the underlying system relies on Pandoc, it is also possible to directly use the Pandoc syntax within Rmarkdown, making it more flexible.

The `knitr` package is the final piece in the puzzle that puts everything together, i.e., referring to the process of knitting. It provides three main functions. First, the source document will be parsed to evaluate which parts of the document have which function, i.e., markdown code that can specify code segments or some text formatting, R code that requires execution, or plain text. Second is the code evaluator who executes the identified code segments. Finally, the renderer that puts the different segments together, i.e., performs the knitting and produces the resulting output file.

MBECS Structure

In order to enable the intended automated workflow of batch correction and comparative evaluation, the packages data structure had to be able to store intermediary results and facilitate data access and manipulation without additional user input. The phyloseq class was chosen as starting point for the packages implementation. The class is extended by two additional fields, named `tss` and `clr`, that can store total sum scaled and centered log ratio transformed values respectively. Two additional

lists for the storage of `assessments` and `corrections` complete the structure of the new `MbecData` class. The class diagram in Figure 4.2 shows that the `MbecData` class also defines three additional function interfaces, `MbecData()`, `MbecSetData`, and `MbecGetData`.

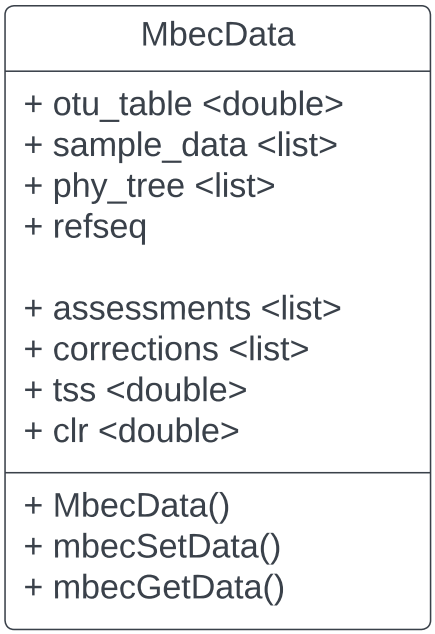


Figure 4.2: The class diagram for the MbecsData class that inherits the data fields otu_table, sample_data, phy_tree, and ref_seq from the phyloseq class and extends it with the additional fields assessments and corrections for feature significance values and batch corrected abundance tables respectively. Additionally, the class defines the two fileds tss and clr to store the respective normalizing transformations. The constructor function `MbecData()` facilitates the creation of new data objects. The ability to access and manipulate an object of class `MbecData` is provided by the functions `mbecSetData()` and `mbecGetData()`.

The so-called constructor `MbecData()` facilitates the generation of new data objects of the type `MbecData`. In order to make the software as user-friendly as possible, the constructor can process inputs of class `phyloseq` and also a simple list that contains an abundance table and some meta-information. The other two functions allow access to and manipulate the data in an object of class `MbecData`. Although it is

technically possible and straightforward to access and change data of objects in R, the use of standardized functions is highly recommended. The reasoning is that these functions provide regular actions that behave pre-determinedly and produce the intended results or an error message in case of invalid input. This characteristic is a highly desirable feature as it ensures the consistent functioning of the software to both developers and end-users.

The public interfaces of `mbecGetData` and `mbecSetData` are shown in Codebox 1. The parameters that are shared between both functions are input.obj, type, and label. The `input.obj` parameter denotes the data object the function will be processing. The argument type determines which slot to access, i.e., the base matrices for un-transformed counts `"otu"`, total sum-scaled counts `"tss"`, cumulative log-ratio transformed counts `"clr"` and batch effect corrected counts `"cor"` and assessment vectors `"ass"`. The later two additionally require the use of the argument 'label' that specifies the name within the respective lists of corrections and assessments.

For the `mbecSetData` function, the additional parameter `new.cnts` is meant for a new data matrix that is stored under the name that is given by the label parameter, i.e., for OTU, TSS, and CLR table, the entry of the appropriate type parameter is sufficient.

The `mbecGetData` function is used to retrieve any of the tables that are stored in a `MbecData` object. It will always return an object containing the selected table and the metadata, i.e., it may be used with default parameters to retrieve features and metadata. The additional parameter 'orientation' determines if the output has features as columns ($s \times f$) or if the columns contain samples ($f \times s$). This is mainly used to retrieve correctly oriented matrices for the different analysis and correction functions. The parameter `required.col` is a vector of column names in the metadata that is required for the analysis at hand. The function only validates that they are present in the data, but it will return the whole meta-frame. This data structure is the backbone of the `MBECS` package that serves as the central access and storage point for all the implemented functionality.

Box 4.1.1 MbecData get and set functions

```
mbecGetData(
  input.obj,
  orientation = "fxs",
  required.col = NULL,
  type = c("otu", "ass", "cor", "clr", "tss"),
  label = character()
)

mbecSetData(
  input.obj,
  new.cnts = NULL,
  type = c("otu", "ass", "cor", "clr", "tss"),
  label = character()
)
```

The current stable version of the `MBECS` package is publicly available in the Bioconductor 3.15 release that was published on April 27th, 2022 [58]. A development version of the package that reflects the ongoing development can be installed from the GitHub repository under `https://github.com/rmolbrich/MBECS`.

In the released version 1.0.0, the package comprises 4787 lines of code that make up a total of 51 functions. The interface and mechanics of these functions are delineated in the 64-page manual that is part of the supplementary material. The test cases contain an additional 1619 lines of code that ensure the consistent and correct functioning of the package.

4.2 Application

The Microbiome Batch-Effect Correction Suite provides a toolkit for stringent assessment and correction of batch effects in microbiome data sets. The package implements a minimal workflow that offers a preliminary report to evaluate the necessity for batch correction, the ability to apply multiple corrections methods, and a post-correction report that enables the comparative evaluation of effectiveness for the employed BECAs.

4.2.1 Workflow

Installation

The MBECS package is part of the Bioconductor repository and can thus be acquired using the installation function provided by the BiocManager package.

Box 4.2.1 Bioconductor installation

```
if(!"BiocManager" %in% rownames(installed.packages()))
     install.packages("BiocManager")

BiocManager::install("MBECS")
```

A package version that incorporates the latest changes can be acquired from the MBECS GitHub repository. Although the repository will generally contain the most current package version, it is not guaranteed to run stable. Intermediary changes made during the development process can produce unexpected issues and are generally fixed and acknowledged with a version bump when updating the repository code. The `devtools` package provides a function that facilitates installing software directly from GitHub. The package can be installed via the CRAN package manager available in R by default. Then the MBECS package can be installed from GitHub as shown in Codebox 2.

Box 4.2.2 GitHub installation

```
# Install from GitHub via the devtools package.
install.packages("devtools")

# This will install the MBECS package from GitHub.
devtools::install_github("rmolbrich/MBECS")
```

The main application of this package is microbiome data. It is common practice to use the phyloseq [98] package for analyses of this type of data. To that end, the MBECS package extends the phyloseq class to provide its functionality. The user can utilize objects of class phyloseq or a list object that contains an abundance table

and metadata. The package contains a dummy data set of artificially generated data to illustrate this process. Loading the package with the `library(MBECS)` command will make all provided functions available to the user.

Import Data

The `data()` function can be used to load the provided mockup data-sets. The dummy data-sets comprise a list that contains an abundance table and metadata `data(dummy.list)`, a phyloseq object `data(dummy.ps)`, and an object of class MbecData `data(dummy.mbec)`. The data contains artificial data for the sole purpose of running examples and showing the package workflow.

To start an analysis, the user requires the `mbecProcessInput()` function. For an input that consists of an abundance table and meta-data, both tables require sample names as either row or column names. They must be passed in a list object with the abundance matrix as the first element. The processing function will handle the correct orientation and return an object of class MbecData. Because the MBECS package inherits its data structure from phyloseq, a correctly formatted phyloseq object can be imported without further considerations. The optional argument `required.col` may be used to ensure that all covariate columns that should be there are available. For the dummy-data these are `"group"`, `"batch"` and `"replicate"`.

Box 4.2.3 infodummy

```
mbec.obj <- mbecProcessInput(dummy.list,
        required.col = c("group", "batch", "replicate"))

mbec.obj <- mbecProcessInput(dummy.ps,
        required.col = c("group", "batch", "replicate"))
```

Apply transformations

The most common normalizing transformations in microbiome analysis are total sum scaling (TSS) and centered log-ratio transformation (CLR). Hence, the MBECS package offers these two methods. The resulting matrices will be stored in their respective slots (tss, clr) in the MbecData object, while the original abundance table

will remain unchanged. Due to the sparse nature of compositional microbiome data, the parameter offset may be used to add a small offset to the abundance matrix to facilitate the CLR transformation. The `mbecTransform()` function enables to apply total sum scaling and centered log-ratio transformations to a data-set. The type of transformation can be chosen with the `method` parameter by using `"tss"` or `"clr"`.

Box 4.2.4 Normalizing transformations

```
mbec.obj <- mbecTransform(mbec.obj, method = "tss")

mbec.obj <- mbecTransform(mbec.obj, method = "clr",
        offset = 0.0001)
```

Preliminary report

The function `mbecReportPrelim()`, shown in Codebox 5, will provide the user with an overview of experimental setup and the significance of the batch effect. To that end, it is required to declare the covariates related to batch effect and group effect, respectively. In addition, it provides the option to select the abundance table to use here. The CLR transformed abundances are the default, and the function will calculate them if they are not present in the input. Technically, the user can start the analysis at this point because the function incorporates the functionality of the aforementioned processing functions.

The parameter `model.vars` is a character vector with two elements. The first denotes the covariate column that describes the batch effect and the second one should be used for the presumed biological effect of interest, e.g., the group effect in case/control studies. The `type` parameter selects which abundance table is to be used `"otu"`, `"clr"` and `"tss"`.

Box 4.2.5 infodummy

```
mbecReportPrelim(input.obj=mbec.obj,
        model.vars=c("batch", "group"), type="clr")
```

Run corrections

The user can choose between assessment and correction functions according to personal preference. The assessment functions incorporate batch effects and user-defined models to control for unwanted variability during the computation of significance values. The abundance values will not be changed in this approach, and the functions will return the statistical analysis results according to the selected method. The correction methods will estimate the variability that can be attributed to batch effects and remove it from the abundance values. This will create new data slots that contain the corrected values for every executed method. These tables can be evaluated for the quality of batch correction in the post-report pipeline and retrieved by the user for further downstream analyses.

The function `mbecCorrection()` will apply a single correction algorithm selected by the parameter method and return an object that contains the resulting corrected abundance matrix in its cor slot with the respective name. The function `mbecRunCorrections()` will apply all correction algorithms selected by the parameter `method` and return an object that contains all respective corrected abundance matrices in the `cor` slot. In the example shown in Codeblock 6 all five available methods are selected for execution. The resulting object will contain five new entries named after the respective methods that created them, i.e., `"ruv3"`, `"rbe"`, `"bmc"`, `"pn"`, and `"svd"`.

Box 4.2.6 BECA

```
mbec.obj <- mbecCorrection(mbec.obj,
        model.vars=c("batch","group"),
        method = "bat", type = "clr")

mbec.obj <- mbecRunCorrections(mbec.obj,
        model.vars=c("batch","group"),
        method=c("ruv3","rbe","bmc","pn","svd"),
        type = "clr")
```

Post report

The post-correction pipeline provides the user with a comparative report for all applied correction methods. The reported sections are equal to the preliminary report. The key difference is that the post report provides paneled plots that illustrate the differences between uncorrected data and other correction methods. The pipeline can be run with the `mbecReportPost()` function as shown in Codebox 7. The parameter `model.vars` is a character vector with two elements. The first denotes the covariate column that describes the batch effect. The second should be used for the biological factor of interest. The `type` parameter selects which abundance table is to be used `"otu"`, `"clr"`, and `"tss"` for comparisons to uncorrected data.

Box 4.2.7 Run post report

```
mbecReportPost(input.obj=mbec.obj,
        model.vars=c("batch","group"),
        type="clr")
```

Data retrieval

Because the `MbecData` class extends the phyloseq class, all functions from phyloseq can also be used. They do, however, only apply to the `otu_table` slot and will return an object of class `phyloseq`, i.e., any transformations or corrections will be lost. For example, to retrieve an object of class `phyloseq` that contains the abundance table of corrected counts, for downstream analyses, the user can employ the `mbecGetPhyloseq()` function. As before, the arguments `type` and `label` are used to specify which abundance table should be used in the returned object.

To retrieve the CLR transformed counts, set the `type` parameter as it is shown in Codebox 8. For corrected abundance matrices the `type` parameter needs to be set to `"cor"` and the `label` parameter selects the chosen matrix, i.e., `"bmc"` for batch mean-centered values as shown in Codebox 8. This concludes the processing pipeline of the `MBECS` package.

Box 4.2.8 Data retrieval

```
ps.clr <- mbecGetPhyloseq(mbec.obj, type="clr")

ps.bmc <- mbecGetPhyloseq(mbec.obj, type="cor", label="bmc")
```

4.2.2 Personalized Usage

Although the package provides a mostly automated processing pipeline that users of all experience levels may use, the functions have been designed to be adjustable to user preferences. For example, the functions used to perform analyses and produce the plots that are part of preliminary and post-correction reports are available to the user. A concise description of each function can be found in the packages' documentation which can be found as part of the supplementary material.

A detailed description of the `mbecHeat` will be subsequently given in order to provide an example of the additional functionality. The creation of heatmaps is part of both reporting pipelines. It will include the top 4 most variable features of the CLR transformed abundance table and compare them with their counterparts in the corrected abundance tables. The function header in Codebox 9 shows the default configuration of this function. The parameters `input.obj`, `model.vars`, `type`, and `label` have previously been explained and fulfill the exact same purpose in this function. Centering and re-scaling of the data prior to the generation of a heatmap can be de-/activated via the parameters `center` and `scale` respectively. The number of features to select for display in the plot is set with the parameter `n` and the `return.data` parameter can be set to `TRUE` to circumvent the plotting function and create a data frame that the user can utilize in their own plotting function. All the exploratory functions, i.e., RLE, PCA, Box, Heatmap, and Mosaic, provide the option to retrieve the data instead of a plot.

The functions for heatmap and differential expression have the additional `method` parameter that allows changing the function's behavior in three different ways. Using `"TOP"` as input will generate a heatmap of top n most variable features based on interquartile range. Setting the parameter to `"ALL"` will select all the features present in the dataset. If this parameter is supplied with a vector of feature names, e.g., `method = c("ASV1", "ASV5", "ASV101")` the function will forgo any selection by

variability or limits imposed by choice of the number of features to display and instead use the selected feature for the plot. This functionality is available for the `mbecBox()` function as well.

Box 4.2.9 Heatmap

```
mbecHeat(
  input.obj,
  model.vars = c("batch", "group"),
  center = TRUE,
  scale = TRUE,
  method = "TOP",
  n = 10,
  type = "clr",
  label = character(),
  return.data = FALSE
)
```

Another notable example of enhanced functionality is the function that governs the variance computation metrics. In this case, the `method` parameter allows choosing between the different approaches for variance estimation, e.g., linear and linear mixed model, RDA, PVCA, and silhouette coefficient. During default operation, this function will generate the appropriate model formulas based on the variables that have been specified with the `model.vars` parameter. This procedure has to follow a somewhat generic scheme to produce reliable deterministic results and is consequently limited in the complexity of the produced models. For example, if a more complex model should be used for analysis, it can be supplied via the `model.form` parameter. This will override the generic model creation and employ the specified model for analysis. More information concerning optional inputs can be found in the documentation of this software package.

Box 4.2.10 Variance Estimation

```
mbecModelVariance(
  input.obj,
```

```
  model.vars = character(),
  method = c("lm", "lmm", "rda", "pvca", "s.coef"),
  model.form = NULL,
  type = c("otu", "clr", "tss", "ass", "cor"),
  label = character(),
  no.warning = TRUE,
  na.action = NULL
)
```

4.2.3 Reporting

A tedious and time-consuming task in data analyses is keeping track of the analysis results. An experienced analyst will have an established routine that facilitates this task, i.e., a specific structure of directories, a naming scheme for the analyses, and a general sequence of typical analyses. However, it is still a lot of programming overhead. The MBECS package provides two automated reporting pipelines that help explore a dataset, assess the magnitude of batch effects, and provide comparative metrics to evaluate the success of different BECAs. To that end, the reports a structured into three main parts a summary of the study, the visualization of selected features, and a statistical assessment of the variance of the dataset.

Both available reports comprise the same three conceptional blocks. The preliminary report is intended to provide an overview of a dataset, decide if batch correction is required, and inform the selection of appropriate correction methods. The preliminary report will perform all the analyses on clr-transformed values unless otherwise specified. After one or more BECAs have been applied to the dataset, a comparative analysis between all applied methods and the uncorrected abundance values will be produced by the post-correction report pipeline. The grouping and types of analyses remain the same as in the preliminary reports, but now the results will be displayed side-by-side with the different correction methods.

Study Summary

The summary section comprises a synopsis of the available metadata, i.e., collected covariate information regarding data type, range, or the number of levels. For example, a mosaic plot that depicts the distribution of samples concerning the biological

factor of interest and the batch factor as shown in Figure 4.3 and an ordination plot shown in Figure 4.4 depicts the sample clustering according to selected principal components.

A particular form of heatmap is the mosaic plot, as shown in figure 4.3, which is used to represent two-way tables, e.g., how samples are distributed over study groups and batches, respectively. Although the cells are commonly color-coded as well, the represented values are indicated by varying height and width of the individual cells.

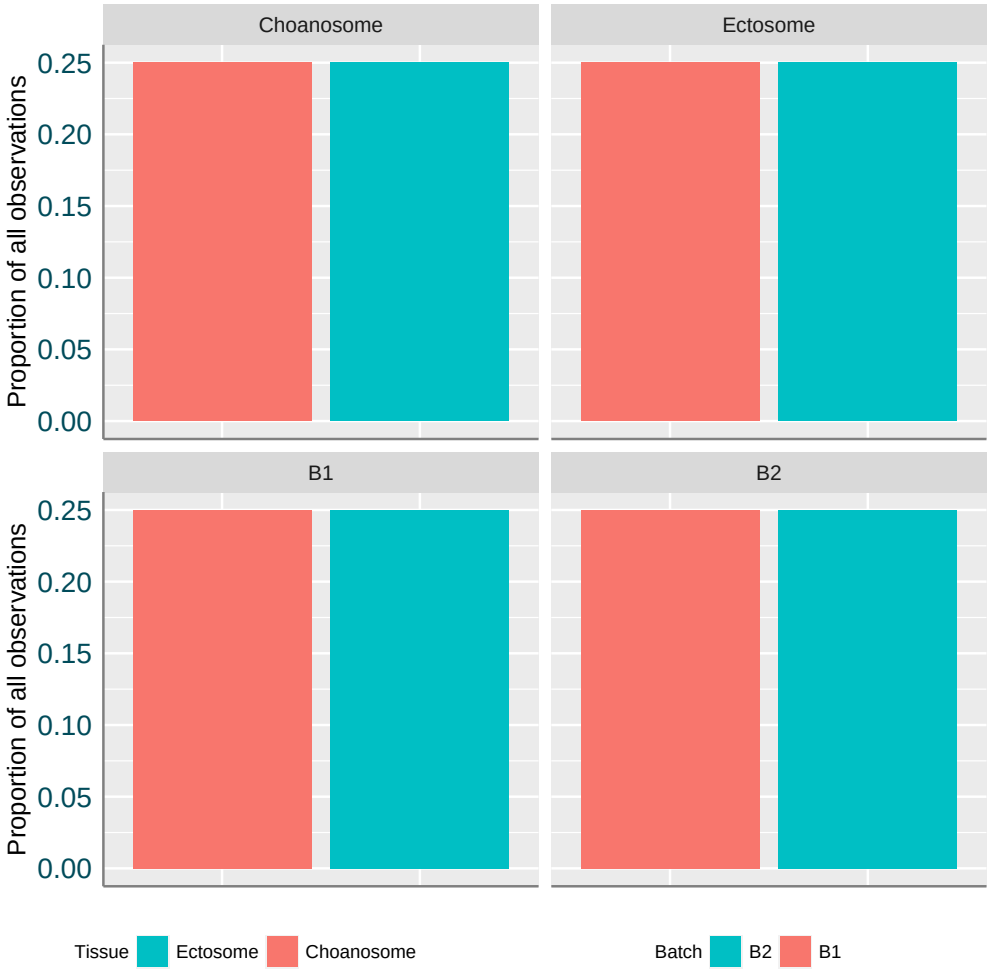


Figure 4.3: A mosaic plot depicting the distribution of samples by tissue and batch, i.e., biological factor and batch factor. For this sponge dataset the samples are distributed evenly for both factors.

Ordination plots show the clustering of data points in two or three-dimensional coordinate systems. This form of representation can be used to show how samples group according to distance metrics, e.g., ecological distances for the depiction of beta-diversity, or to represent the data in principle coordinates as shown in Figure 4.4. The newly formed axes are a hierarchical ordering of the directions of the most extensive variability in the data. An ideal tool for exploratory data analyses is displaying how the samples are clustered in a two or three-dimensional plot.

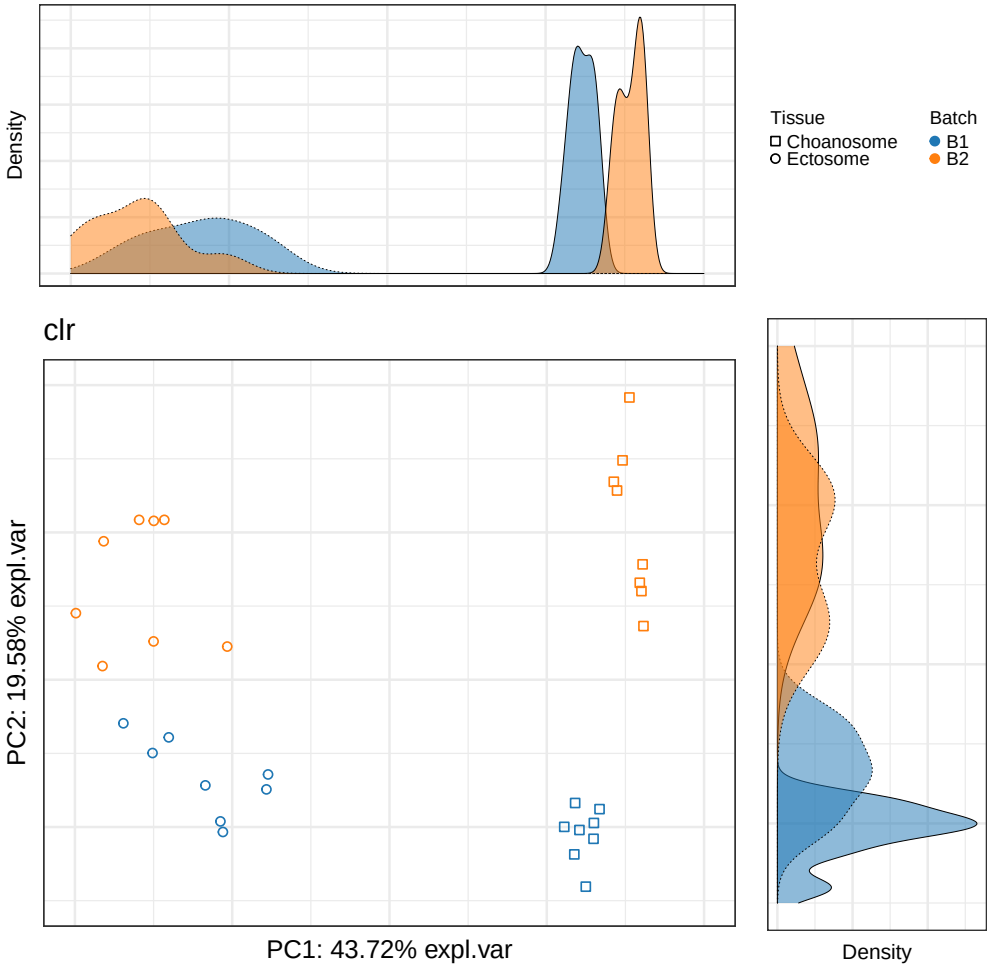


Figure 4.4: An ordination plot depicting the grouping of samples along the first two principal components as well as the distribution of values over these axes. The first principal component reflects sample separation according to batch and the second components captures the separation due to tissue effects.

Visualization

The following section provides more insight into the characteristics of the features. This section includes a box plot of relative log-expression per sample in study groups separated by samples, which helps visually assess variation in the data. The most

variable features, according to interquartile ranges, can be displayed in the form of heatmaps as shown in Figure 4.6 for all samples or as box plots that show the features' mean abundances over the respective batches shown in Figure 4.6.

Let $Y_{m\times n}$ be the matrix of OTU abundance, to produce the relative log expression plot subtract from all samples their respective median of feature abundances, i.e., $\hat{y}_{i,j} = y_{i,j} - Med(y_{*,j})$ in terms of the matrix notation. When displayed in the form of a box plot such as in Figure 4.5 this calculation will reveal sample heterogeneity. The general assumption with expression data is that only the expression values of a few genes are affected by the biological factor of interest [167]. Under this assumption, in the absence of additional sources of variation, the majority of genes would show an average value modified by zero-mean random variation, i.e., the RLE plot only shows the remaining noise. Because the results of mean and median operators diverge when this assumption is broken by additional unwanted variation, the box plots will illustrate this circumstance with a larger displacement of boxes and pronounced differences in heights.

As such, the plot helps explore data sets for the presence of unaccounted or unwanted variation. It can also be utilized to assess whether a method aimed to remove unwanted variation has worked. However, it is not a qualitative measure of the success of such an algorithm because it is not possible to determine if the removed variation also includes the biological signal [55].

A heatmap is a form of graphical data representation where values are color-coded to visualize them in a two-dimensional graphic. Molecular biology, in particular, employs cluster heatmaps to show gene expression values in a matrix layout of colored cells. This matrix's columns and rows are sorted, e.g., by study group and up-or down-regulated genes, to indicate how the investigated phenomenon clusters and varies between samples and groups. Figure 4.6 shows OTUs selected for their high variability.

Feature differential abundance visualization technique uses box plots to show the abundance of selected features concerning specified groups. Hence the plot can indicate meaningful differences between study groups or batches for a particular gene. Figure 4.7 shows a highly variable feature from an existing data-set. This representation includes the distribution of abundance values over both groups, i.e., a density plot. This additional information shows whether or not both groups are

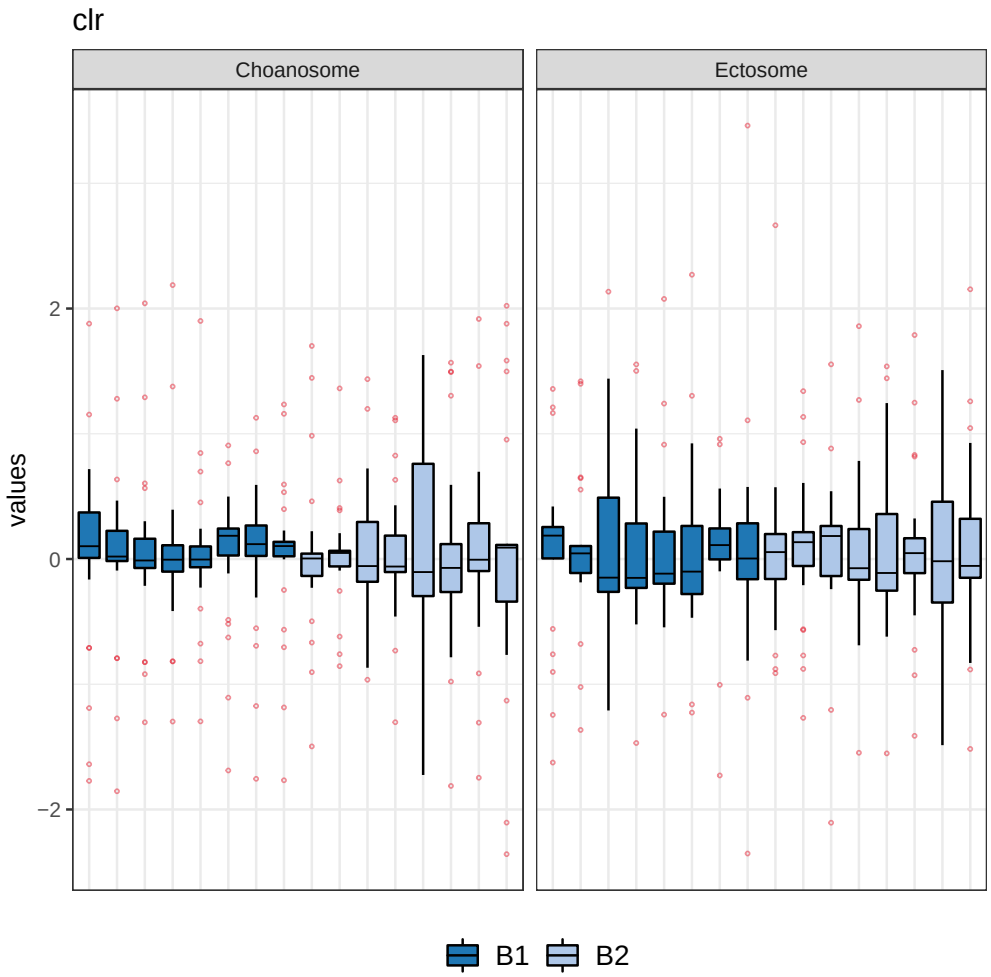


Figure 4.5: A relative log-expression plot that shows the expression values of samples in two panels that correspond to the two investigated tissue types. Samples are colored according to batch affiliation. The plot shows the heterogeneous behavior of samples within and between batches, suggesting that the variation can be attributed to both biological and batch factors.

comparable or if there is a substantial mismatch, e.g., only a small subset of samples is driving the values in either group.

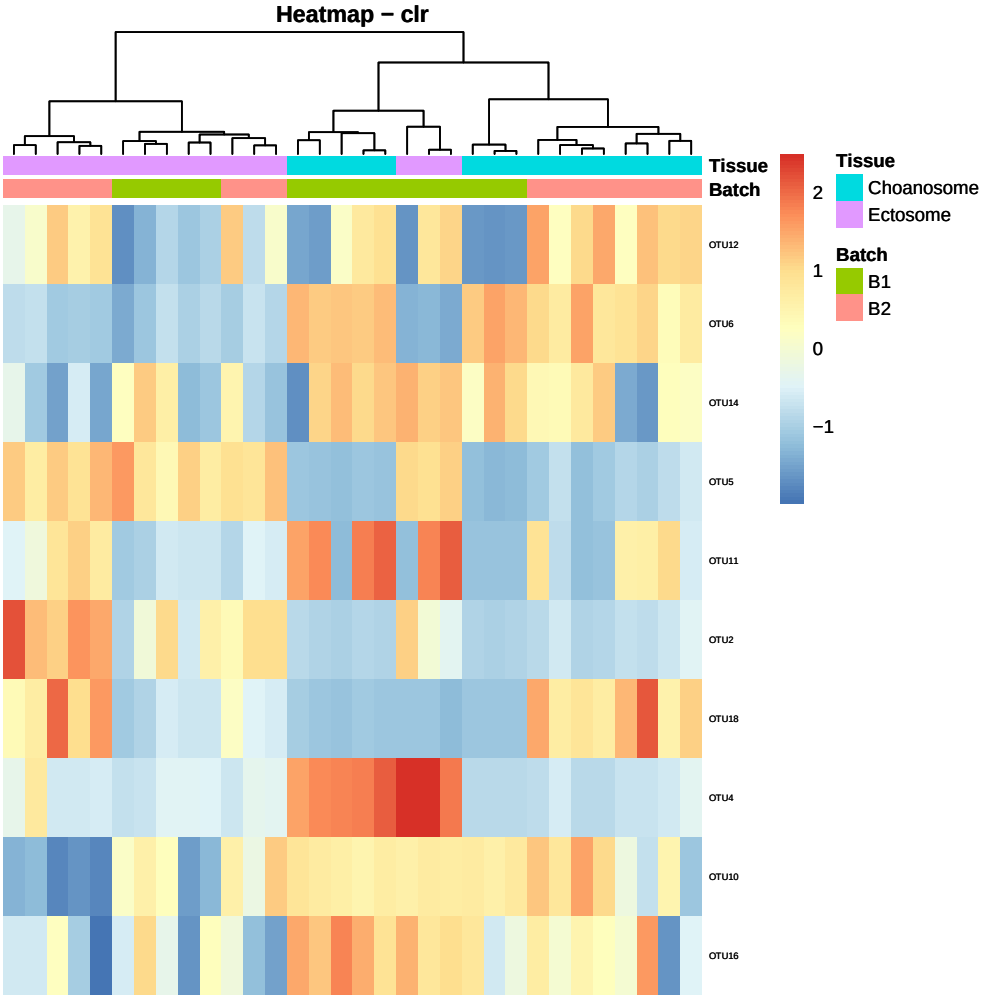


Figure 4.6: A heatmap of the expression of the 10 most variable features in the sponge dataset.

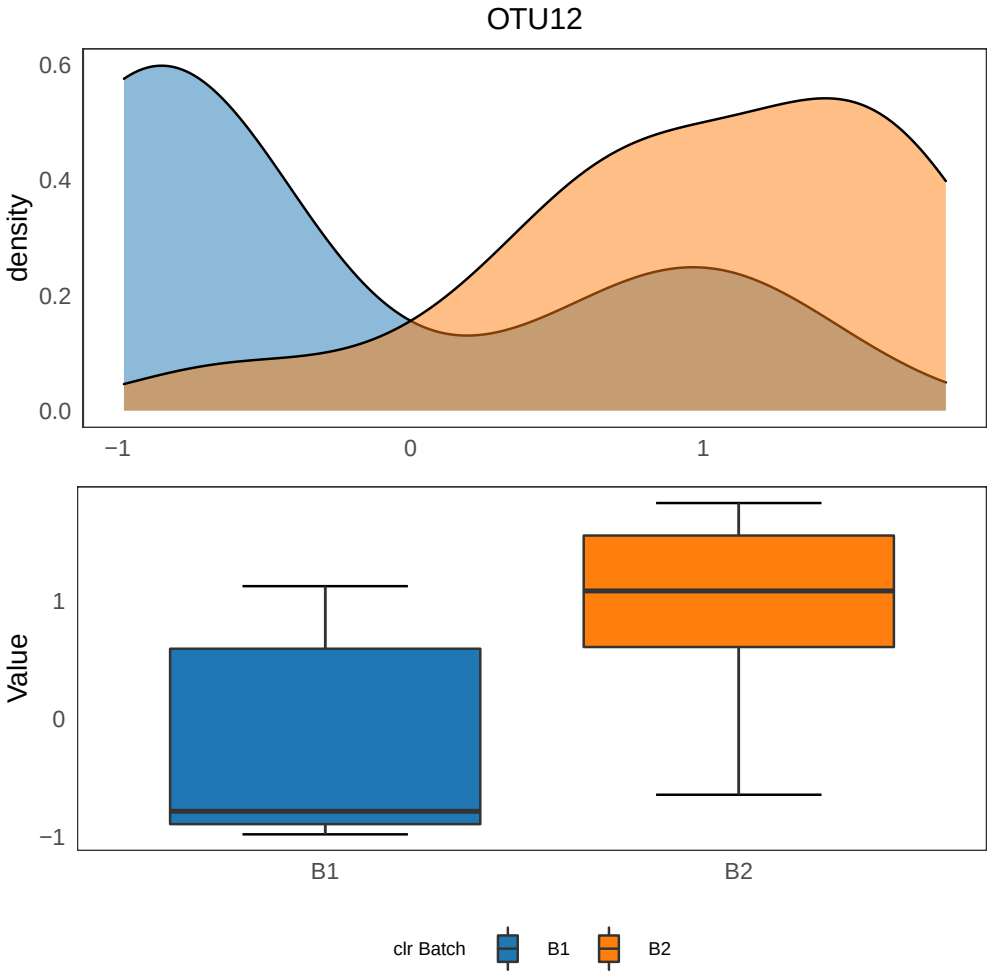
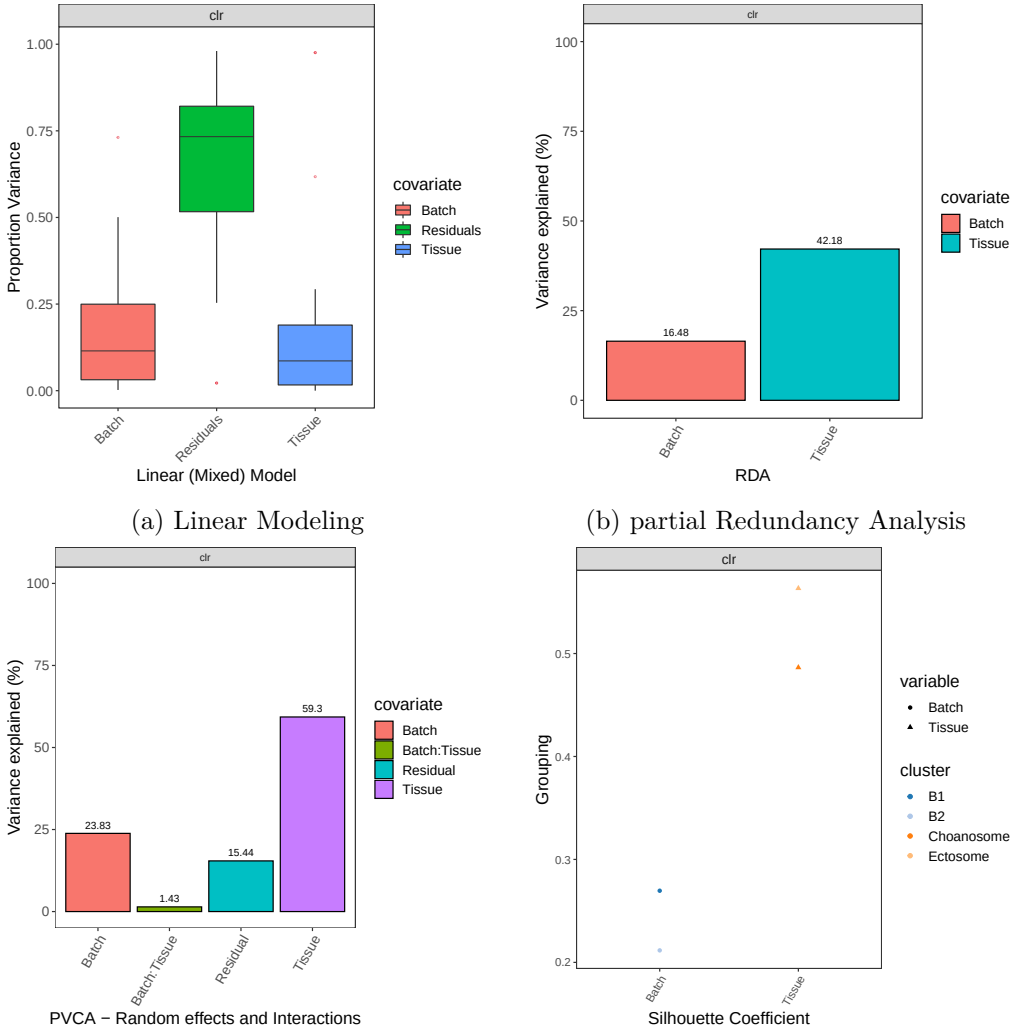


Figure 4.7: A box-plot that shows the average abundance values for OTU12 grouped according to batch factor. The feature was selected according to IQR value. The figure is an excerpt from the report in which the five most variable features are shown.

Variance Assessment

The third section comprises all the methods that estimate the amount of variability that can be attributed to biological factors and batch factors, respectively. This includes a box plot for linear mixed model fitting results and bar plots to depict variance distribution according to partial redundancy analysis and principal variance components analysis. And finally, the clustering quality in terms of the silhouette coefficient (Figure 4.8).



(a) Linear Modeling

(b) partial Redundancy Analysis

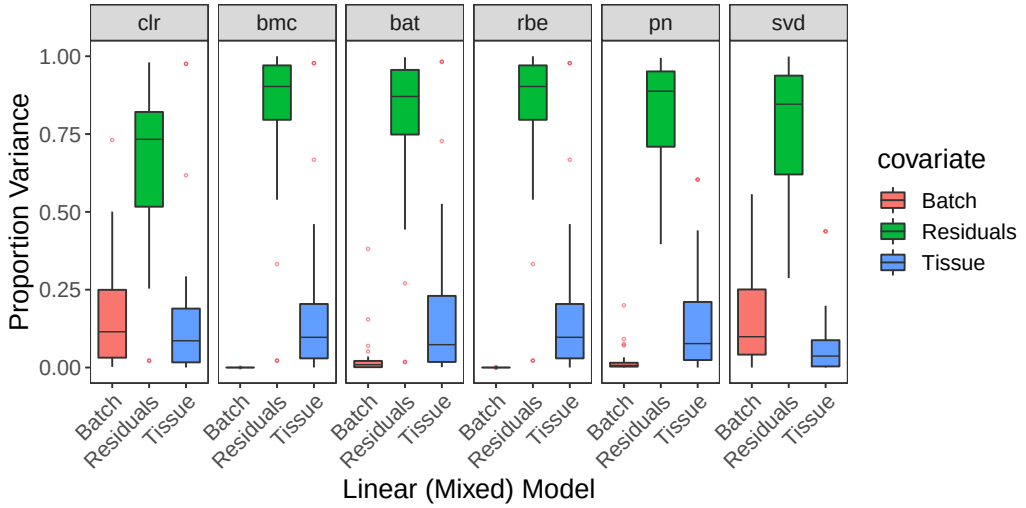
(c) Principal Variance Components Analysis

(d) Silhouette Coefficient

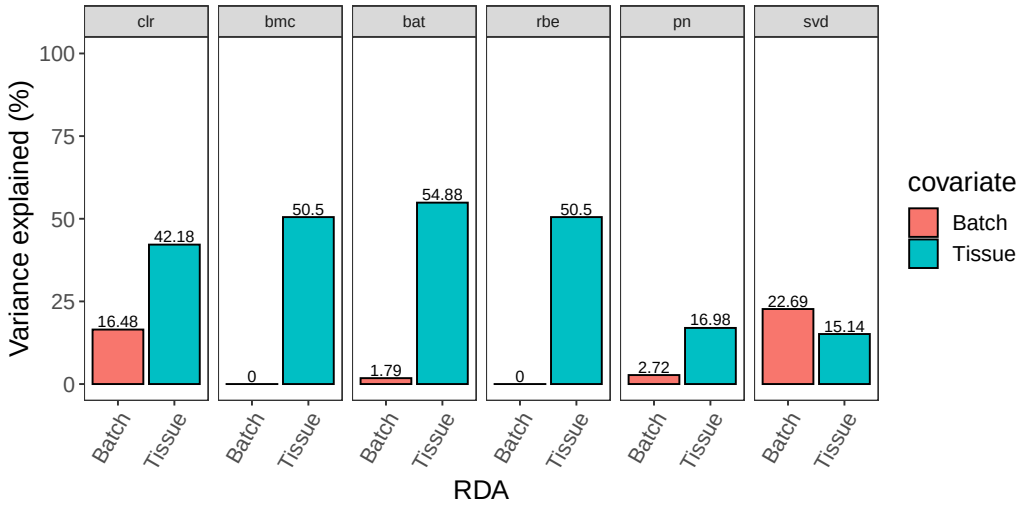
Figure 4.8: The figure shows the plots for variance assessment metrics as used in the preliminary report.

Post Correction Assessment

The post-correction report is different compared to the preliminary report in that it provides paneled plots (Figures 4.9 and 4.10) that enable the direct comparison between different correction methods.

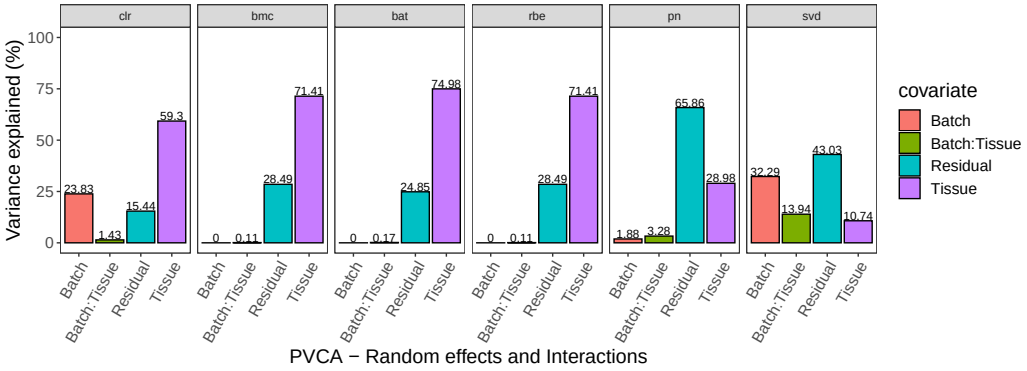


(a) Linear Model

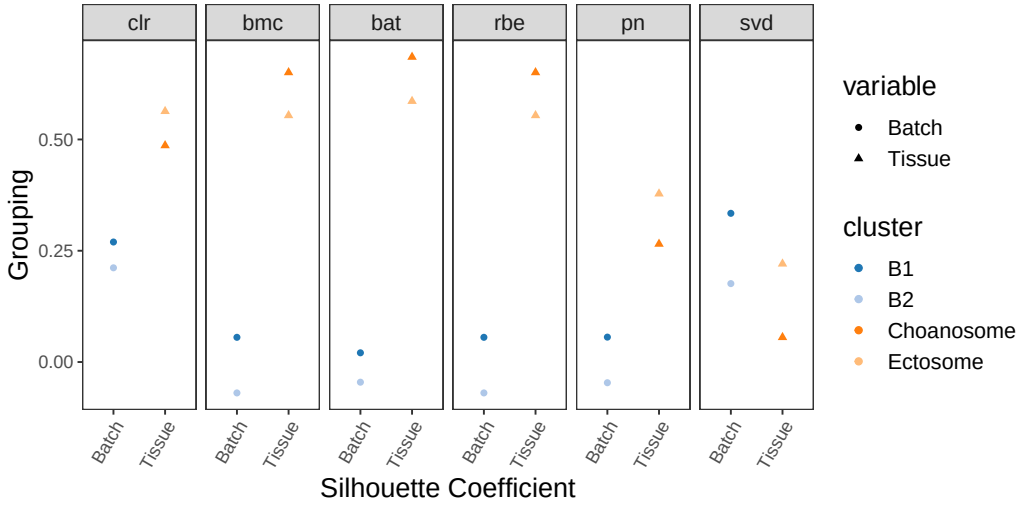


(b) Redundancy Analysis

Figure 4.9: The figure shows paneled plots that compare the success of the utilized BECAs with a linear model (Panel 4.9a and Redundancy Analysis (Panel 4.9b.



(a) Principal Variance Component Analysis



(b) Silhouette Coefficient

Figure 4.10: The figure shows paneled plots that compare the success of the utilized BECAs with Principal Variance Components Analysis (Panel 4.10a and Silhouette Coefficient (Panel 4.10b.

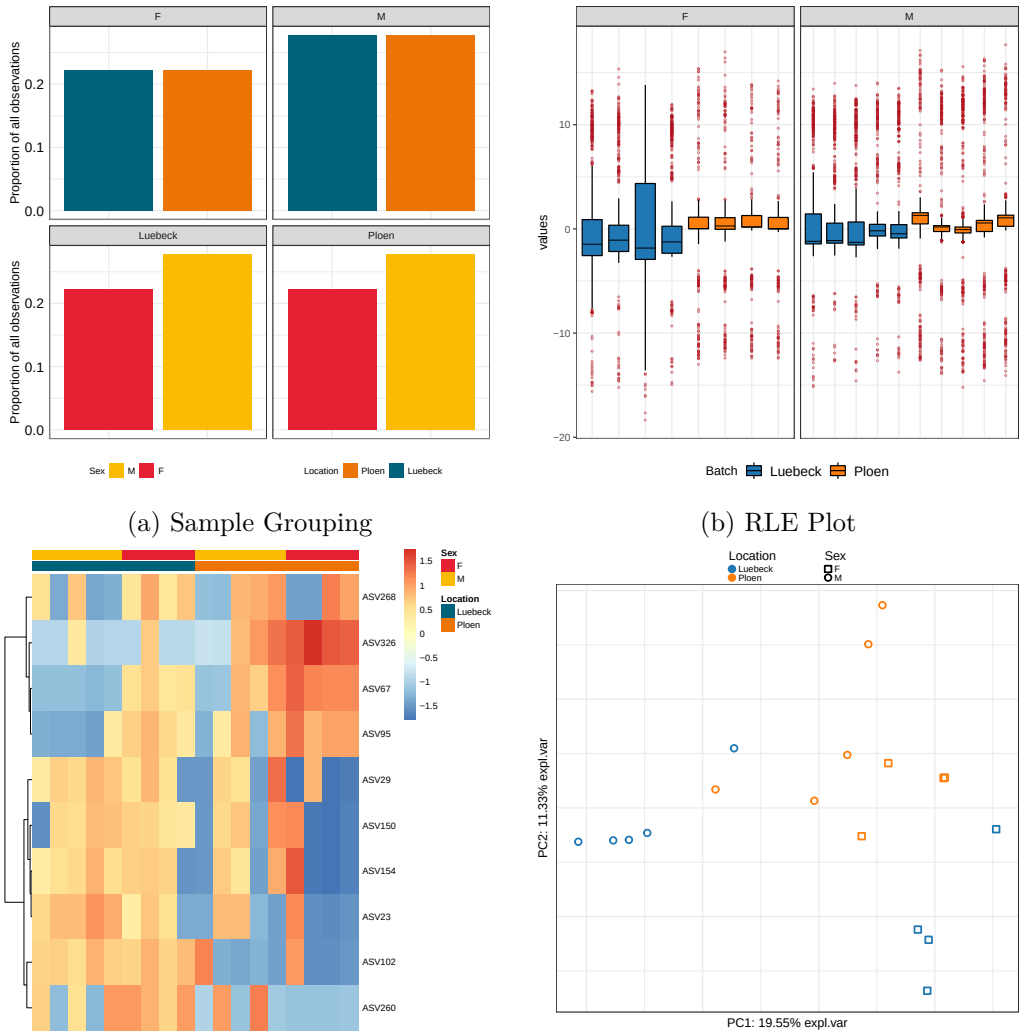
4.3 Evaluation

4.3.1 Mouse data

The distribution of mouse samples according to biological and batch factors is depicted in the mosaic plot in Figure 4.11a. Samples from both sexes have been distributed equally at the two processing locations, i.e., considered as the batch factor. The ordination plot on the first two principal components in Figure 4.11d shows a pronounced separation of male and female samples on the first principal component that follows a similar pattern for both processing sites. However, the separation is more apparent in the group that was processed in Lübeck. Either of the principal components does not represent the batch effect. The difference between the processing sites is highlighted in the heatmap of the 20 most variable features depicted in Figure 4.11c. The variables that indicate sample grouping according to batch, sex, and replicate status at the top facilitate the comparison between duplicate samples in either batch. It can be seen that the same ASVs have different abundances in both batches.

All BECAs that are available in the `MBECS` package can be applied to the mouse dataset. However, for the sake of argument, the sex factor is considered the reference in percentile normalization. As stated in Section 3.2.9 one of the limiting factors of this approach is the requirement for a two-level grouping factor in order to define a reference level for percentiles that the other group is adjusted to, i.e., this is typically regarded as the case-control grouping. Because the dataset comprises replicates, it is also possible to apply the RUV-3 algorithm that incorporates the differences between replicate samples into the estimation procedure for batch effects. The assessment of correction effectiveness is summarized in Figures 4.12 and 4.13 that shows the results of linear modeling (Figure), principal variance components analysis (Figure), partial redundancy analysis (Figure), and silhouette coefficient (Figure). The modeling with a linear mixed model was left out as it offers no additional information over the linear model due to the even distribution of samples into batches. For all of the used metrics, the comparisons include an assessment of the uncorrected CLR transformed data in addition to the outcomes of the employed correction algorithms.

Although the performance evaluation of BECAs, in general, is out of the scope of this work, the two methods show particularly poor performance, evidenced by



(a) Sample Grouping

(b) RLE Plot

(c) Heatmap

(d) PCA Ordination

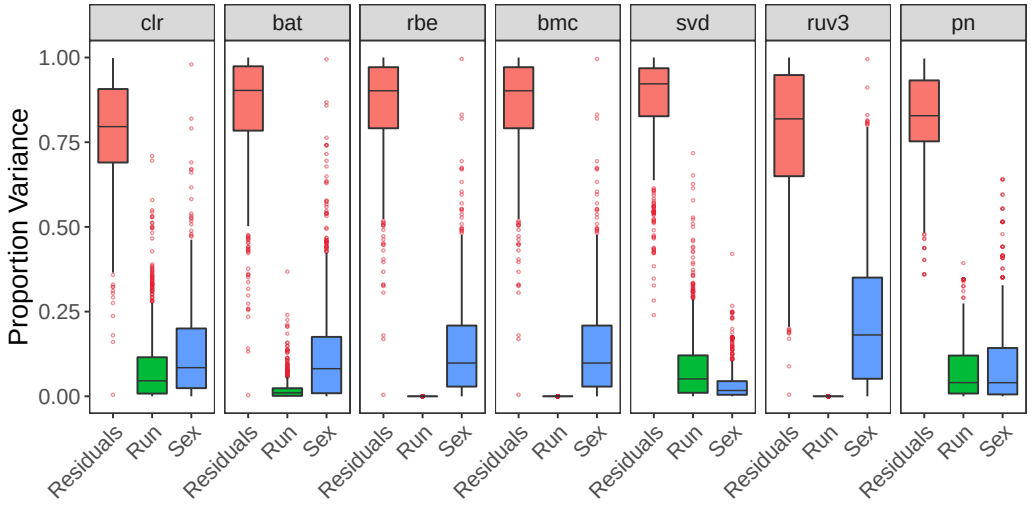
Figure 4.11: Panel 4.11a: Mosaic plot that shows the distribution of mouse data samples into two batches. The biological factor is the sex difference between the samples. The dataset contains four female and five male samples replicated at two processing locations, i.e., considered the batch factor. Panel 4.11b: The relative log-expression plot highlights the sample difference according to batch. The samples are grouped by sex and colored by batch. Because the samples are technical replicates that have been sequenced at two different laboratories, it is apparent from the difference in expression that this proceeding has introduced a batch effect. Panel 4.11c: Heatmap of the most variable features. Panel 4.11d: Principal components ordination that shows the sample clustering on the first two PCs. The biological factor seems to be represented in PC1 for the female samples and in PC2 for the male samples. The batch effect roughly separates the samples on a diagonal.

all four applied assessment methods. For singular value decomposition, the Figures 4.12a, 4.12b, and 4.13a show no reduction in the variance that can be attributed to the batch factor or even a slight exaggeration. Moreover, the effect of the biological factor is shown to be reduced after the correction procedure. This notion is also confirmed in the cluster coefficients, where the Lübeck batch shows a slightly better clustering performance, which is adverse to the desired outcome. More importantly, according to the biological factor, the clustering is worse and implies that important information was lost in the correction process. This outcome is easily explained by the algorithmic approach of SVD batch correction. Because the algorithm assumes that the first principal component, i.e., the linear combination that reflects the most extensive variability in the dataset, represents the batch effect. It will remove that component from the data. It was shown in the PCA in Figure 4.11d that the first component reflects the separation primarily according to the biological factor in any case. Hence, the method removed parts of the biological effect.

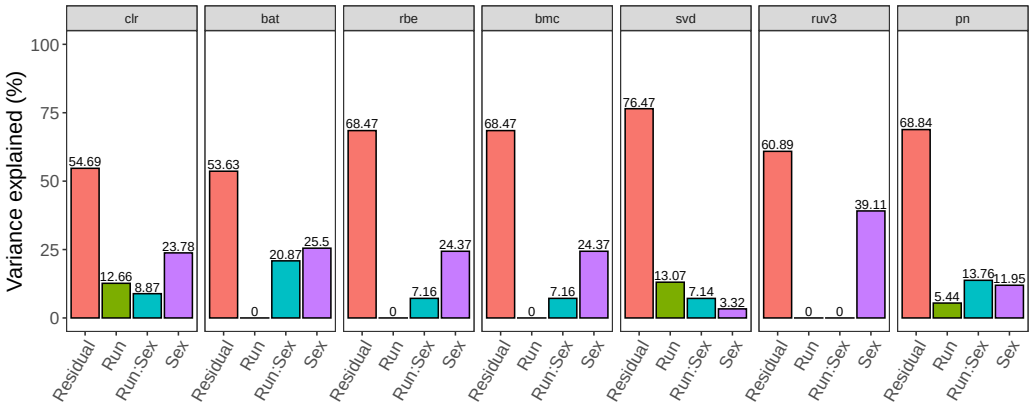
Despite the better performance of the percentile normalization approach, the algorithm still underperforms compared to the other methods. However, all of the metrics show that the batch factor has been reduced minimally while the biological factor was affected to a large degree. One of the shortcomings of this approach that has been outlined in Section 3.2.9 is the loss of information due to the transformation from abundance values into percentiles. This characteristic is likely the reason for the noticeable reduction in variability, i.e., inherent information, of the biological factor.

The remaining methods show a good performance according to the metrics, with RUV-3 exhibiting the best results. This algorithm is the only one that utilizes replicate samples in its correction approach. Since both batches essentially contain replicate samples, it is reasonable to expect an excellent performance of this algorithm.

The mouse dataset is a practical tool for the testing of the `MBECS` package due to the implicit batch effect introduced by separate processing sites and the two-level biological factor. The outlined analyses show the package's utility in assessing and correcting batch effects.

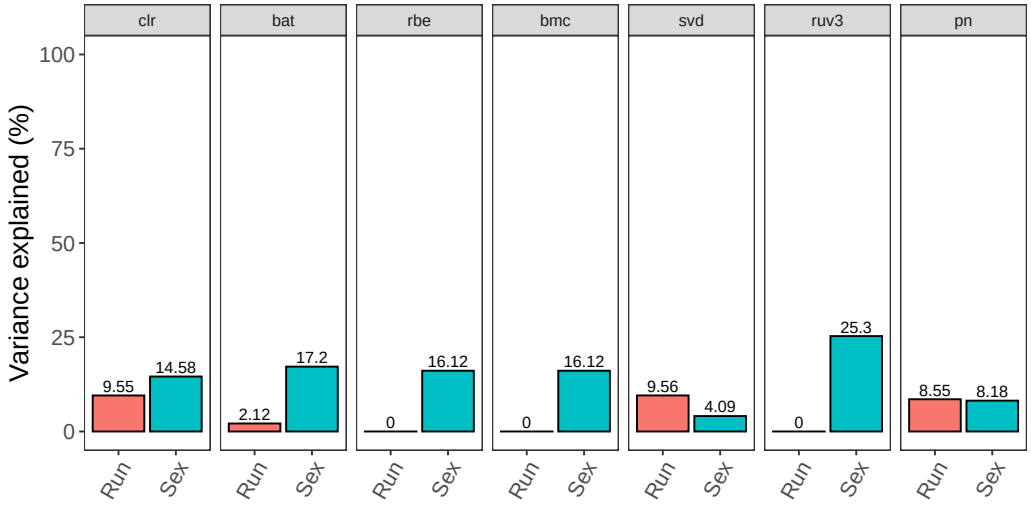


(a) Linear Model

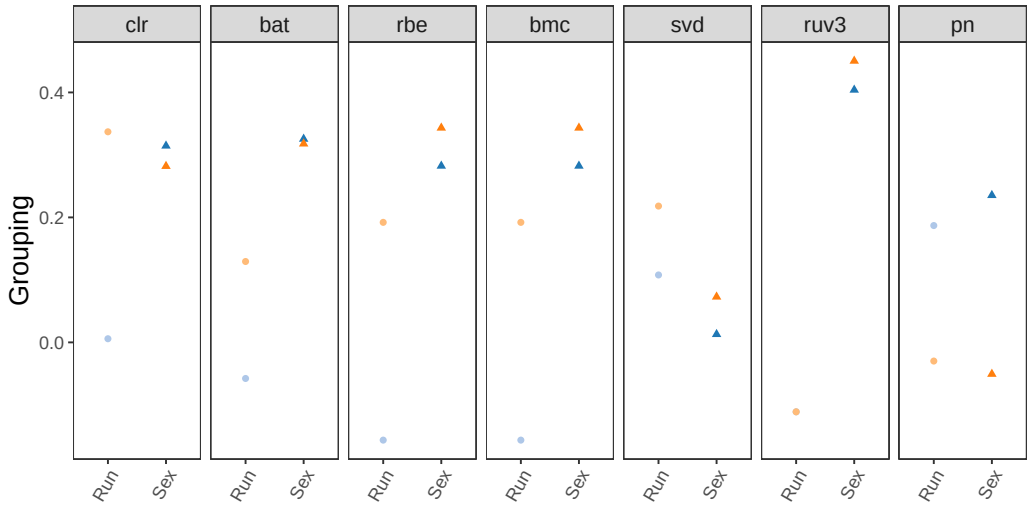


(b) Principal Variance Components Analysis

Figure 4.12: The figure shows paneled plots that compare the success of the utilized BECAs in the mouse dataset with a linear model (Panel 4.12a and Principal Variance Components Analysis (Panel 4.12b.



(a) Redundancy Analysis



(b) Silhouette Coefficient

Figure 4.13: The figure shows paneled plots that compare the success of the utilized BECAs in the mouse dataset with Redundancy Analysis (Panel 4.13a and Silhouette Coefficient (Panel 4.13b.

4.3.2 Dog Data

The dog dataset is an example of an existing study that shows an outstanding design. The dataset comprises microbial communities from 12 different sampling locations on the skin as well as a set of stool samples. The mosaic plot in Figure 4.14 show the sample grouping according to batch and location. It can be seen that the design of the study takes batch processing into account and aims to process samples according to their sampling location. Thus, the samples for each location were processed in a single run or equally distributed across two different batches at most. Performing the sample processing in batches related to the sampling location is a sensible decision because due to the different conditions that each location provides, the sites are not comparable. For example, if the samples were distributed across all batches without consideration of the compositional differences inherent to the sampling sites, this would simultaneously introduce the potential need for batch correction and limit the effectiveness of such procedures as fewer samples deliver information concerning a particular batch respectively. This is a result of the requirement to correct for potential batch effects according to their sampling site. The locations that have been processed in two batches, such as elbow, or flank, all exhibit high quality and no apparent need for batch correction procedures. The respective reports are part of the supplementary material.

Apart from presenting an excellent example of a study that limits the emergence of batch effects by appropriately grouping samples, this dataset can also serve as an example that outlines the inappropriateness of comparing different sampling sites. These body sites represent biomes with particular environmental conditions that are advantageous for some organisms and detrimental to others. This implies a difference in communal composition that renders distinct body sites incomparable.

Within the scenario of a clinical study, it is conceivable that it is considered to pool different sampling locations for analysis, e.g., to improve the statistical power of a study by incorporating more samples. The argument would be that the sampling sites might be treated as two different studies that must be integrated. Thus it is feasible to consider them as batches and perform a batch correction prior to the statistical analyses.

To illustrate the error of this assumption, consider the following scenario where the dataset has been reduced to contain only samples from the locations "abdomen" and

"stool". It is reasonable to argue that these habitats will present different communal compositions. For example, figure 4.15 shows the PCA on a subset of the dog dataset that comprises stool and abdomen samples of diseased and healthy subjects. The plot shows a distinct separation according to the sampling site, which, in this case, may also be considered the batch effect due to the processing in different batches.

Applying the batch correcting algorithms to this subset of samples is straightforward. The location is considered the batch factor, and the biological factor includes healthy and diseased samples. In contrast to the mouse, dataset RUV-3 is not applicable due to the lack of replicate samples. The assessment results for the effectiveness of the BECAs is shown in the four panels of Figures 4.16 and 4.17. Batch mean centering and RBE seemingly show good performance by reducing batch-related variability to about point two percent while the effect of the biological factor increases. Although the SVD approach performs worse than the former two algorithms, it reduces the batch variability significantly for this dataset. The PCA ordination in Figure 4.15 showed that the first principle component captured the majority of the location differences. Hence, it is reasonable that SVD performs well in reducing this effect. Notable in this comparison is the performance of ComBat, which removes not only the batch effect but also the biological factor, which is reflected in all applied metrics. Another curiosity is the inflation of variance that is attributed to the interaction of batch and biological factors despite the reduction of the batch effect shown in the results of the PVC analysis in Figure 4.16b.

The subsequent figures investigate both locations as separate datasets in order to shed light on this issue. The principal components ordination in Figures 4.18a and 4.18b naturally show a less clear separation, seeing that the biological factor is the distinguishing parameter now.

Panels 4.19a and 4.19b show the linear model estimates for variance attributable to the grouping factor in abdomen and stool samples respectively. Panels 4.19c and 4.19d show the results of the Principal Variance Components Analysis for the biological factor in both sampling sites. For both methods, the estimates are higher than the respective analyses for the combined dataset, which indicates that information is concealed when grouping these sampling sites for analysis.

Panels 4.20a and 4.20b shows the RDA estimates for variance attributable to the grouping factor in abdomen and stool samples respectively. Notably, the estimates are almost twice as high for abdomen samples and four times higher for stool samples

than in the combined dataset. Panels 4.20c and 4.20d show the Silhouette Coefficient in both sampling sites for the biological factor. Although the coefficient does not indicate an overwhelmingly good fit for either group, the clustering is better for the separate inspection of sampling sites.

The relative abundance on the Phylum level (Figure 4.21 shows that the variability in the dataset has to be considered in the context of communal composition. Because the environmental conditions in a habitat shape the communal composition of microorganisms, i.e., the presence and absence of particular species due to the living conditions, the two sampling sites show very different compositions on the Phylum level already. Although both sampling sites share the presence of microorganisms in the Phyla Actinobacteria, Bacteroidetes, Firmicutes, Fusobacteria, and Campilobacterota, apart from Bacteroidetes these Phyla are present in very different numbers between both sampling sites. This means that large parts of the variability between the two sampling sites can be attributed to the differential prevalence of taxa. Thus it is not reasonable to use batch effect correction to make vastly different sampling sites comparable.

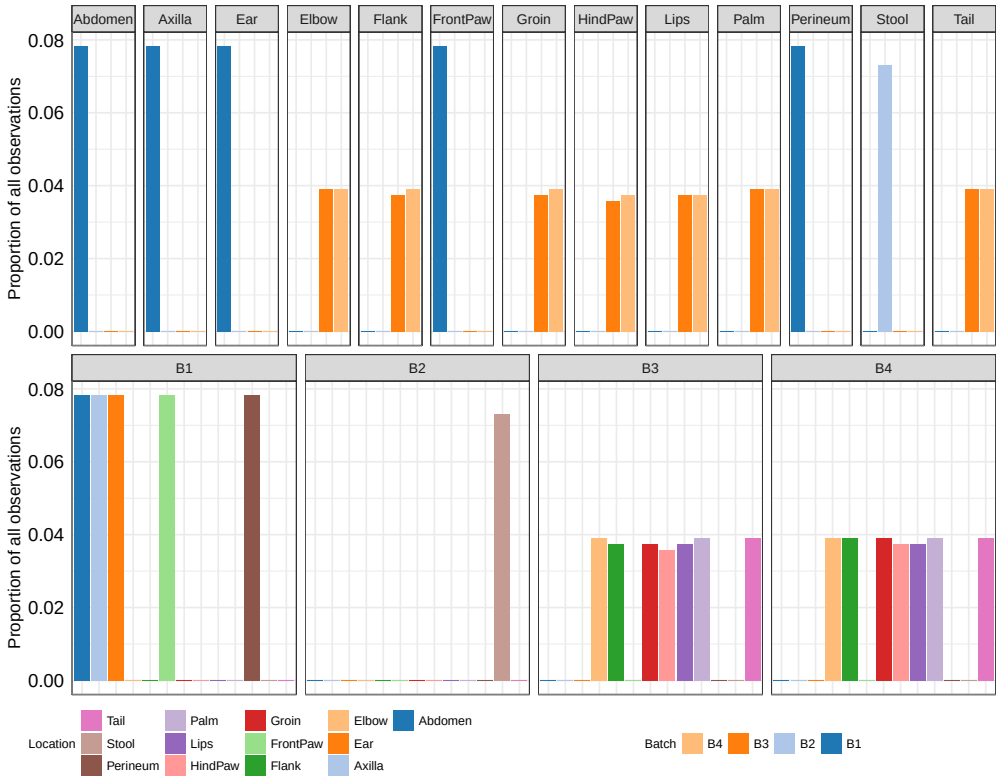


Figure 4.14: Mosaic plot that shows the grouping of samples according to batch factor and sampling location. The study's design took batch processing into account and aimed to process samples according to their sampling location. Thus, the samples for each location were processed in a single run or equally distributed across two different batches at most. The dataset comprises microbial communities from different sampling locations on the skin as well as stool samples that were processed in an exclusive batch. Performing the sample processing in batches related to the sampling location is a sensible decision because due to the different conditions that each location provides, the sites are not comparable. For example, if the samples were distributed across all batches without considering the compositional differences inherent to the sampling sites, this would simultaneously introduce the potential need for batch correction and limit the effectiveness of such procedures as fewer samples deliver information concerning a particular batch, respectively. This is a result of the requirement to correct for potential batch effects according to their sampling site.

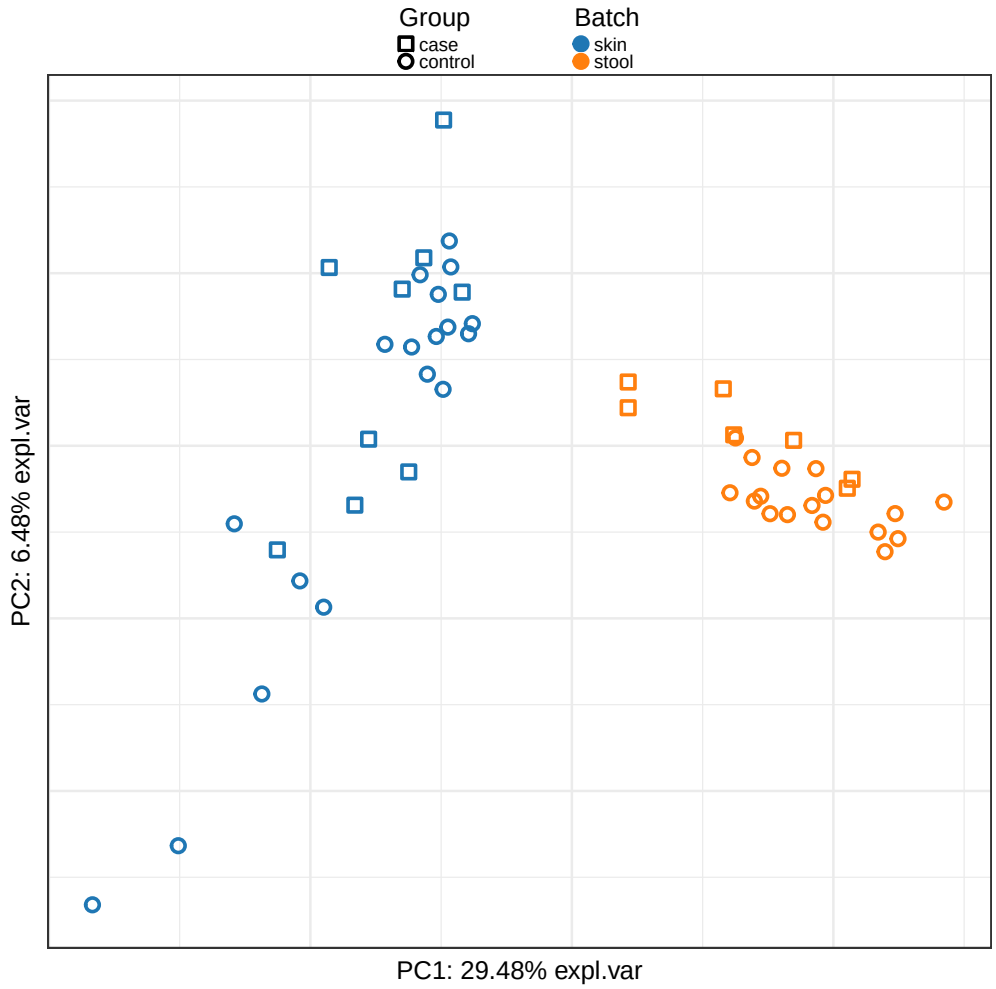
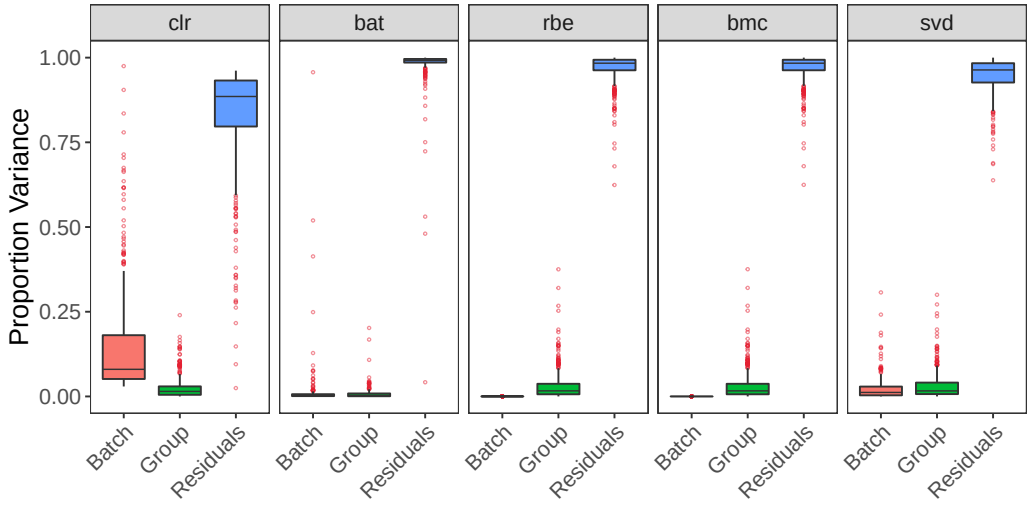
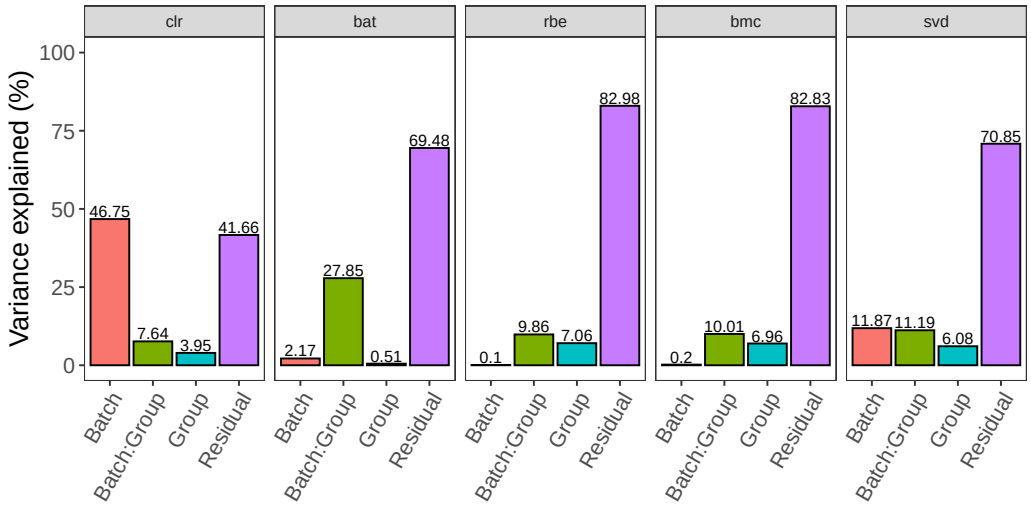


Figure 4.15: A principal component plot of the dog dataset after subsetting for the two habitats "abdomen" and "stool". The plot shows a distinct separation according to the sampling site which, in this case, may also be considered the batch effect due to the processing in different batches.

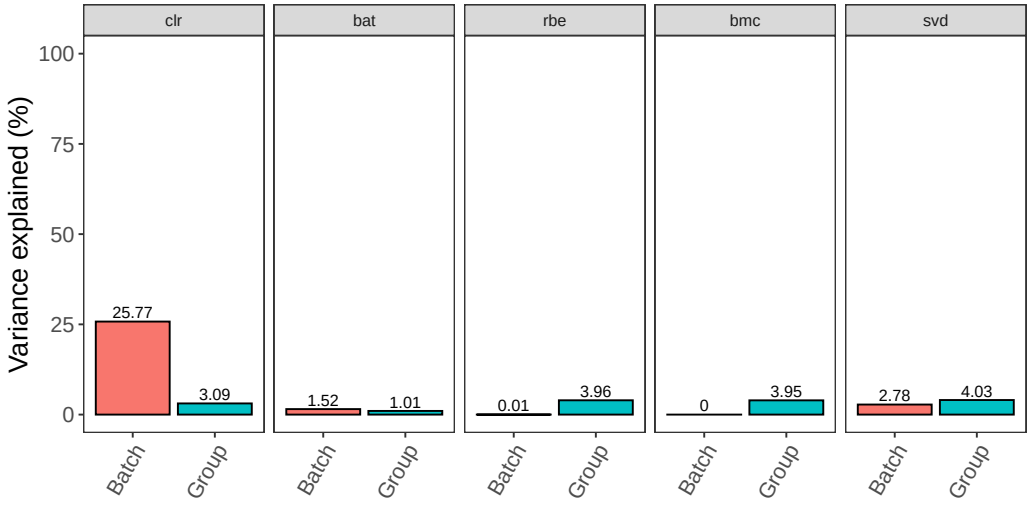


(a) Linear Model

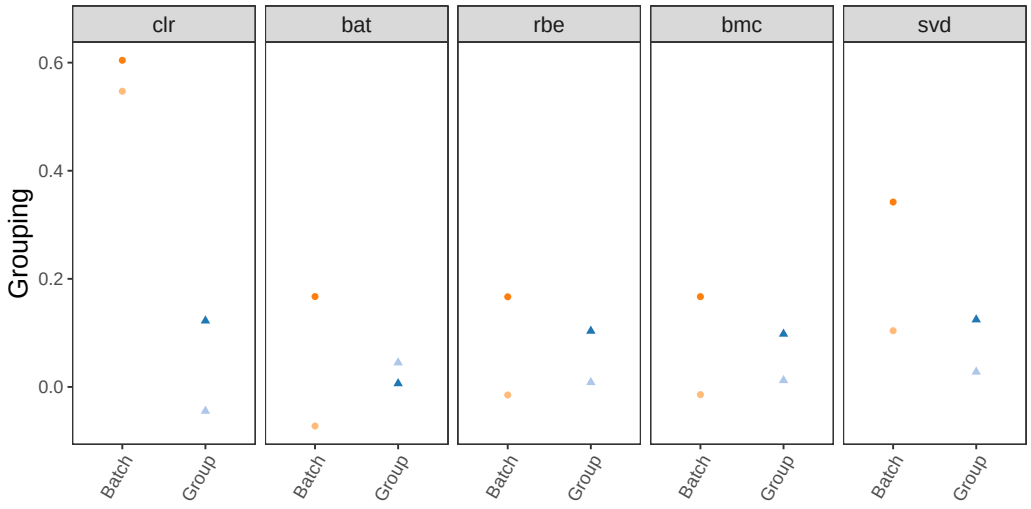


(b) Principal Variance Components Analysis

Figure 4.16: The figure shows paneled plots that compare the success of the utilized BECAs in the reduced dog dataset with a linear model (Panel 4.16a and Principal Variance Components Analysis (Panel 4.16b.

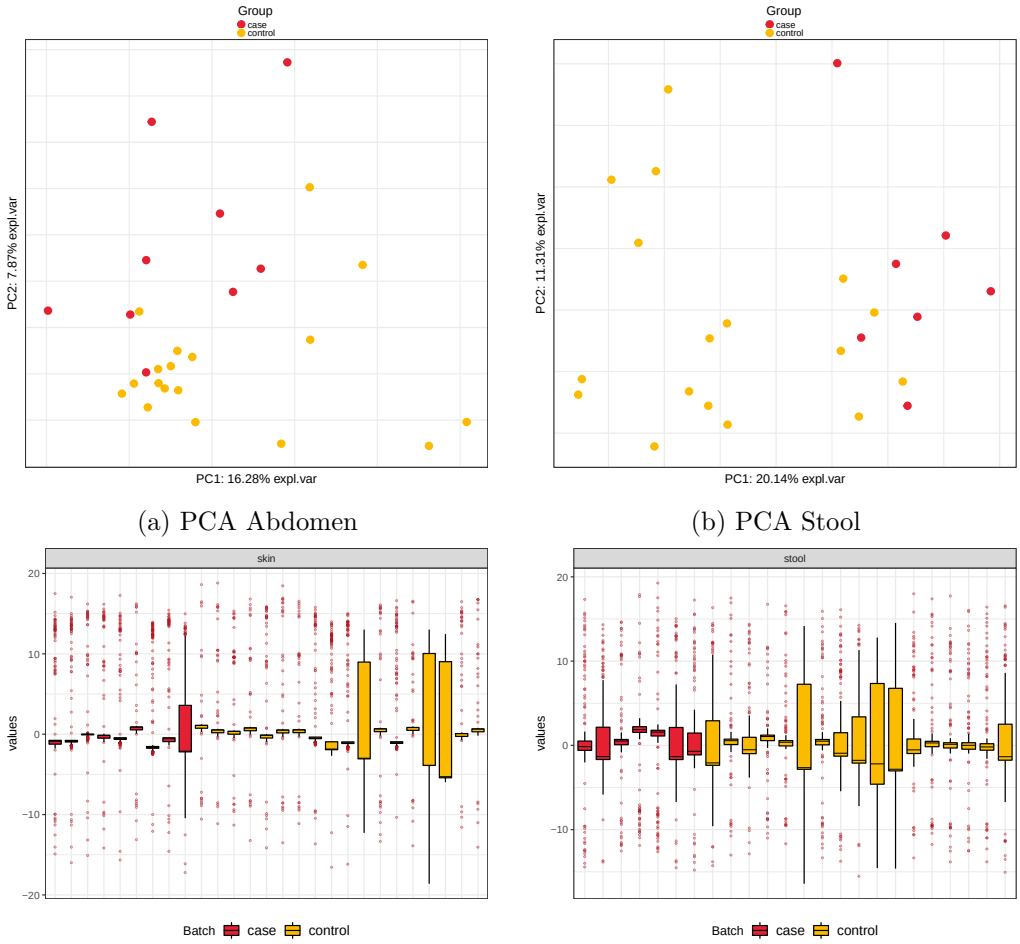


(a) partial Redundancy Analysis



(b) Silhouette Coefficient

Figure 4.17: The figure shows paneled plots that compare the success of the utilized BECAs in the reduced dog dataset with Redundancy Analysis (Panel 4.17a and Silhouette Coefficient (Panel 4.17b.



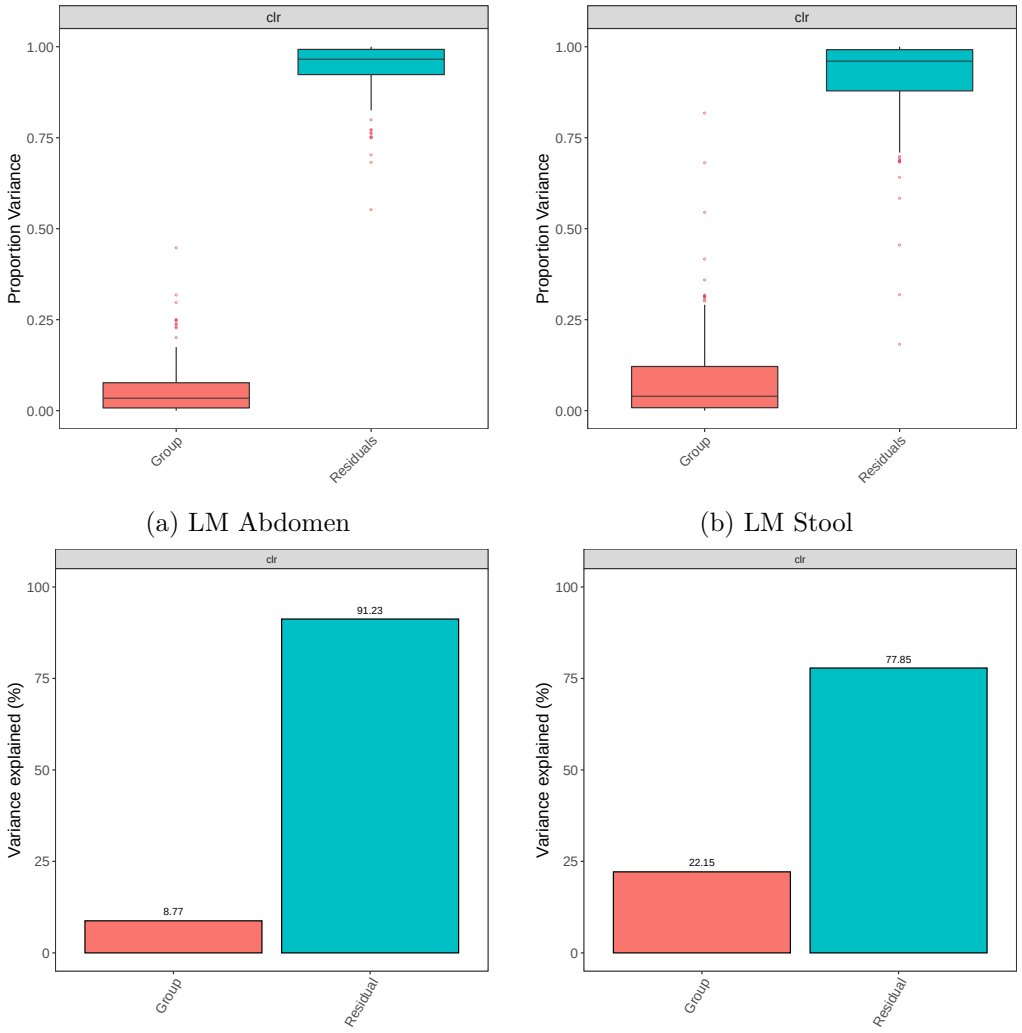
(a) PCA Abdomen

(b) PCA Stool

(c) RLE Abdomen

(d) RLE Stool

Figure 4.18: The panels 4.18a and 4.18b show the ordination of samples on the first two principal components after subsetting the dog dataset to process the sampling locations for "abdomen" and "stool" separately.



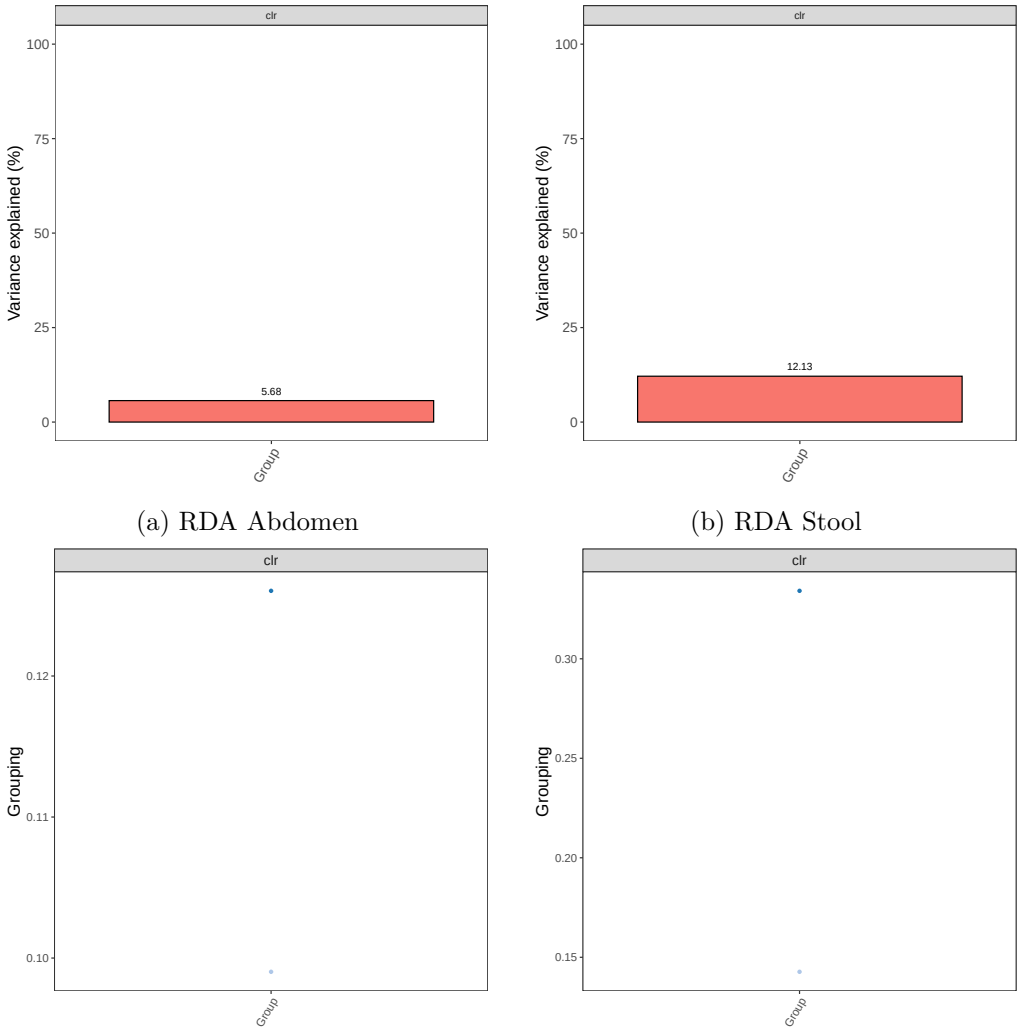
(a) LM Abdomen

(b) LM Stool

(c) PVCA Abdomen

(d) PVCA Stool

Figure 4.19: The panels 4.19a and 4.19b show the linear model estimates for variance attributable to the grouping factor in abdomen and stool samples respectively. Subfigures 4.19c and 4.19d show the results of the Principal Variance Components Analysis for the biological factor in both sampling sites. For both methods, the estimates are higher than the respective analyses for the combined dataset, which indicates that information is concealed when grouping these sampling sites for analysis.



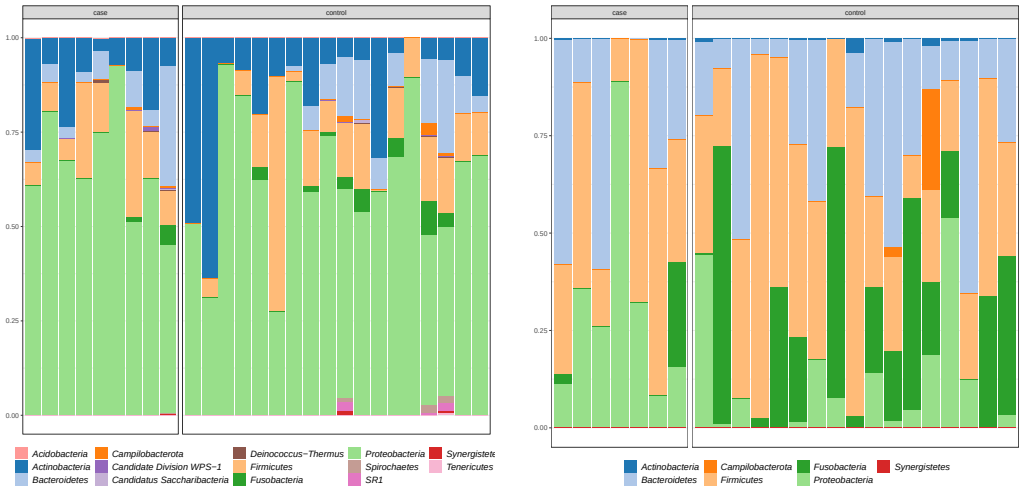
(a) RDA Abdomen

(b) RDA Stool

(c) Silhouette Coefficient Abdomen

(d) Silhouette Coefficient Stool

Figure 4.20: The panels 4.20a and 4.20b show the variance attributable to the grouping factor in abdomen and stool samples respectively. Notably, the estimates are almost twice as high for abdomen samples and four times higher for stool samples than in the combined dataset. Subfigures 4.20c and 4.20d show the Silhouette Coefficient in both sampling sites for the biological factor. Although the coefficient does not indicate an overwhelmingly good fit for either group, the clustering is better for the separate inspection of sampling sites.



(a) Abdomen Phylum relative abundance

(b) Stool Phylum relative abundance

Figure 4.21: The relative abundance on the Phylum level shows that the variability in the dataset has to be considered in the context of communal composition. Because the environmental conditions in a habitat shape the communal composition of microorganisms, i.e., the presence and absence of particular species due to the living conditions, the two sampling sites show very different compositions on the Phylum level already. Although both sampling sites share the presence of microorganisms in the Phyla Actinobacteria, Bacteroidetes, Firmicutes, Fusobacteria, and Campilobacterota apart from Bacteroidetes, these Phyla are present in very different numbers between both sampling sites. This means that large parts of the variability between the two sampling sites can be attributed to the differential prevalence of taxa. Thus it is not reasonable to use batch effect correction to make vastly different sampling sites comparable.

5 Discussion and Conclusion

The emergence of unwanted variation is an inherent quality of signal capturing processes and consequently also applies to next-generation sequencing methods utilized in microbial ecology. A particular form of noise, the batch effect, is commonly encountered in all types of NGS data. The comprehensive scientific research into causes and strategies for the prevention and correction of batch effects indicates the importance of this topic. This chapter reviews the executed strategy toward implementing a software application that facilitates stringent handling of batch effects in microbiome datasets.

5.1 Summary

This work presented a comprehensive overview of the challenges posed by batch effects in microbiome datasets. Next-generation sequencing technology advancements provide the means to satisfy these requirements by generating high throughput datasets. The fundamental steps in generating these datasets were outlined and put into the context of batch effects. The delineation of batch effects as a particular form of noise related to performing any required sample processing steps in a distinct group, i.e., processing in batches, illustrated how batch effects could emerge.

When investigating batch effects, it is necessary to assess their impact and magnitude on a particular dataset and apply BECAs to correct them for downstream analyses. To that end, the approaches to data modeling and metrics for estimating central tendency have been outlined. Correction algorithms were explained to provide an insight into the underlying assumptions and potential applicability due to strengths and weaknesses. Finally, the R-framework and all utilized software resources were introduced as the workspace for application development.

This work results in the Microbiome Batch Effect Correction Suite, which provides a solution in the form of a toolbox that incorporates both evaluation and correction

methods for BEs. The software structure, utilization, and utility were shown in Chapter 4.

5.2 Assessment

Dealing with batch effects in microbiome datasets is a curious issue. Although plenty publications can be found that discuss strategies [63, 147], review the effectiveness of algorithms [142], or propose new computational approaches [37], the matter is rarely discussed in publications of experimental findings. In a review of upcoming priorities for MB research Cullen et al. never even mention batch effects as an issue [32].

The causes for the emergence of batch effects that were outlined in Section 2.3 introduce several challenges from the inception and design of a study, over the sample acquisition, to the lab work and the final analysis. However, understanding the causes provides a variety of measures and strategies that help prevent the emergence or at least mitigate the adverse impact of batch effects [63].

Batch effect correcting algorithms are to be considered an auxiliary strategy that shows robust performance in assisting in the mitigation of unavoidable technical issues such as the limited capacity of a sequencer or finite reagent batches [169]. The performance of BECAs also depends on the characteristics of the particular dataset, i.e., number of samples, size of the feature space, and the magnitude of variability, and it is therefore not obvious which method will produce the best result [31, 142]. In the case of strongly confounded batch and class factors, the effectiveness of BECAs is limited [169]. In the best-case scenario, a batch factor can not be assessed and consequently not removed from the data. In the worst case, the correction procedure may systematically introduce group differences that are interpreted as significant results during downstream analyses [110].

Thus the optimal strategy for proper handling of batch effects is to avoid them through clever study designs and the implementation of, and adherence to strict processing protocols [31, 63, 169]. However, exploring a dataset in terms of unwanted variation should be considered a regular part of quality control efforts. The `MBECS` package presents a resource that facilitates the convenient handling of batch effects in microbiome data. Due to the integrated reporting function, the option to run a batch correction, and the evaluation of effectiveness with minimal user input, the

workflow can be incorporated into existing analysis procedures. The package was already utilized throughout the development in this workgroup.

5.3 Outlook

Batch effects pose a challenge in microbiome analyses that is unlikely to disappear. Even coming technological improvements that are to be expected will not resolve this problem in its entirety. Thus the study and development of batch effect correcting algorithms will remain relevant.

Concerning the further development of the `MBECS` package, this means that new algorithms and evaluation metrics will become available. One of the shortcomings of current computational approaches is their original development for micro-array data. The growing importance of microbial ecology as the scientific field will continually spawn new algorithms that should be integrated into the software to provide the best possible results and make it an invaluable resource in microbiome analysis.

As for the functional improvements of the package, the upcoming steps should include refinement of the visual representation and the integration of additional functionality. Because the software builds on the phyloseq class, a prudent next step in development would be overloading the functionalities that phyloseq provides. This means that functionality like subsetting of samples, agglomeration of taxonomic levels, or access to data fields is possible for objects of class MbecData by using the functions provided by phylosec.

5.4 Conclusion

In conclusion, the created software and the documentation represent a comprehensive resource concerning the prevention and correction of batch effects in microbiome datasets. In particular, the automation and the generation of comprehensive reports may facilitate the consideration and integration into existing workflows.

Acknowledgements

I want to thank Prof. Dr. rer. nat. Hauke Busch, for taking a chance in providing me with the opportunity to study and grow in an unfamiliar scientific field. I am grateful for the support and sage advice given to me by Dr. rer. nat. Anke Fähnrich. Finally, I thank Dr. rer. nat. Axel Künstner for his continued support and counseling during my work.

List of Figures

List of Tables

Bibliography

[1] Scott Abbott et al. “Experiments on Plant Hybrids by Gregor Mendel”. In: *Genetics* 204.2 (Oct. 2016), pp. 407–422. ISSN: 1943-2631. DOI: 10.1534/genetics.116.195198 (cit. on p. 5).

[2] Isabel Abellan-Schneyder et al. “Primer, Pipelines, Parameters: Issues in 16S rRNA Gene Sequencing”. In: *mSphere* 6.1 (Feb. 2021). Ed. by Susannah Green Tringe, e01202–20. ISSN: 2379-5042. DOI: 10.1128/mSphere.01202-20 (cit. on pp. 19, 27).

[3] Irina Abnizova et al. “STATISTICAL COMPARISON OF METHODS TO ESTIMATE THE ERROR PROBABILITY IN SHORT-READ ILLUMINA SEQUENCING”. In: *Journal of Bioinformatics and Computational Biology* 08.03 (June 2010), pp. 579–591. ISSN: 0219-7200, 1757-6334. DOI: 10.1142/S021972001000463X (cit. on p. 23).

[4] Ian M. Adcock et al. “Transcription Factors”. In: *Asthma and COPD*. Elsevier, 2009, pp. 373–380. ISBN: 978-0-12-374001-4. DOI: 10.1016/B978-0-12-374001-4.00031-6 (cit. on p. 8).

[5] Luisa Alba-Lois et al. “Yeast Fermentation and the Making of Beer and Wine”. In: *nature education* 3 (Jan. 2010), p. 17 (cit. on p. 12).

[6] JJ Allaire et al. *rmarkdown: Dynamic Documents for R*. R package version 2.13. 2022. URL: https://github.com/rstudio/rmarkdown (cit. on p. 78).

[7] Amnon Amir. “Microbiome Analysis Using 16S Amplicon Sequencing: From Samples to ASVs”. In: *Deep Sequencing Data Analysis*. Ed. by Noam Shomron. Vol. 2243. New York, NY: Springer US, 2021, pp. 123–141. ISBN: 978-1-07-161102-9. DOI: 10.1007/978-1-0716-1103-6_7 (cit. on p. 27).

[8] Naomi Attar. "Raymond Gosling: The Man Who Crystallized Genes". In: *Genome Biology* 14.4 (2013), p. 402. ISSN: 1465-6906. DOI: 10.1186/gb-2013-14-4-402 (cit. on p. 8).

[9] Baptiste Auguie. *gridExtra: Miscellaneous Functions for "Grid" Graphics*. 2017 (cit. on pp. 76, 78).

[10] Stephan Bache et al. *Magrittr: A Forward-Pipe Operator for R*. 2020 (cit. on p. 73).

[11] Douglas Bates et al. "Fitting Linear Mixed-Effects Models Using **Lme4**". In: *Journal of Statistical Software* 67.1 (2015). ISSN: 1548-7660. DOI: 10.18637/jss.v067.i01 (cit. on p. 74).

[12] Alexandre Bazin et al. "A De Novo Robust Clustering Approach for Amplicon-Based Sequence Data". In: *Journal of Computational Biology* 26.6 (June 2019), pp. 618–624. ISSN: 1557-8666. DOI: 10.1089/cmb.2018.0170 (cit. on p. 27).

[13] Keith R. Benson. "T. H. Morgan's Resistance to the Chromosome Theory". In: *Nature Reviews Genetics* 2.6 (June 2001), pp. 469–474. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/35076532 (cit. on p. 5).

[14] Ronald Bentley. "The Development of Penicillin: Genesis of a Famous Antibiotic". In: *Perspectives in Biology and Medicine* 48.3 (2005), pp. 444–452. ISSN: 1529-8795. DOI: 10.1353/pbm.2005.0068 (cit. on p. 14).

[15] Gabriele Berg et al. "Microbiome Definition Re-Visited: Old Concepts and New Challenges". In: *Microbiome* 8.1 (Dec. 2020), p. 103. ISSN: 2049-2618. DOI: 10.1186/s40168-020-00875-0 (cit. on p. 14).

[16] Pooja Bhardwaj. "Types of Sampling in Research". In: *Journal of the Practice of Cardiovascular Sciences* 5.3 (2019), p. 157. ISSN: 2395-5414. DOI: 10.4103/jpcs.jpcs_62_19 (cit. on pp. 18, 19).

[17] Steve M. Blevins et al. "Robert Koch and the 'Golden Age' of Bacteriology". In: *International Journal of Infectious Diseases* 14.9 (Sept. 2010), e744–e751. ISSN: 12019712. DOI: 10.1016/j.ijid.2009.12.003 (cit. on p. 13).

[18] Herb Brody. "The Gut Microbiome". In: *Nature* 577.7792 (Jan. 2020), S5–S5. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/d41586-020-00194-2 (cit. on p. 16).

[19] Maren Büttner et al. *Assessment of Batch-Correction Methods for scRNA-seq Data with a New Test Metric*. Preprint. Bioinformatics, Oct. 2017. DOI: 10.1101/200345 (cit. on pp. 33, 34).

[20] Allyson L. Byrd et al. "The Human Skin Microbiome". In: *Nature Reviews Microbiology* 16.3 (Mar. 2018), pp. 143–155. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/nrmicro.2017.157 (cit. on p. 16).

[21] Charles H. Calisher. "Taxonomy: What's in a Name? Doesn't a Rose by Any Other Name Smell as Sweet?" In: *Croatian Medical Journal* 48.2 (Apr. 2007), pp. 268–270. ISSN: 1332-8166 (cit. on p. 10).

[22] Benjamin J Callahan et al. "DADA2: High-resolution Sample Inference from Illumina Amplicon Data". In: *Nature Methods* 13.7 (July 2016), pp. 581–583. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3869 (cit. on p. 24).

[23] Arturo Casadevall et al. "Host-Pathogen Interactions: Redefining the Basic Concepts of Virulence and Pathogenicity". In: *Infection and Immunity* 67.8 (Aug. 1999). Ed. by V. A. Fischetti, pp. 3703–3713. ISSN: 0019-9567, 1098-5522. DOI: 10.1128/IAI.67.8.3703-3713.1999 (cit. on p. 13).

[24] Erwin Chargaff et al. *The Nucleic Acids. Volume III Volume III*. New York: Academic Press, 1960. ISBN: 978-0-323-14477-3 (cit. on p. 7).

[25] Chao Chen et al. "Removing Batch Effects in Analysis of Expression Microarray Data: An Evaluation of Six Batch Adjustment Methods". In: *PLoS ONE* 6.2 (Feb. 2011). Ed. by Daniel Kliebenstein, e17238. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0017238 (cit. on p. 38).

[26] Nicolas Cichocki et al. "Bacterial Mock Communities as Standards for Reproducible Cytometric Microbiome Analysis". In: *Nature Protocols* 15.9 (Sept. 2020), pp. 2788–2812. ISSN: 1754-2189, 1750-2799. DOI: 10.1038/s41596-020-0362-0 (cit. on p. 59).

[27] Lauren C. Cline et al. "Moving beyond *de Novo* Clustering in Fungal Community Ecology". In: *New Phytologist* 216.3 (Nov. 2017), pp. 629–634. ISSN: 0028-646X, 1469-8137. DOI: 10.1111/nph.14752 (cit. on p. 27).

[28] *Concise Medical Dictionary*. Oxford; New York: Oxford University Press, 2010. ISBN: 978-0-19-172701-6 (cit. on p. 13).

[29] Angela N. H. Creager. *The Life of a Virus: Tobacco Mosaic Virus as an Experimental Model, 1930-1965*. Chicago: University of Chicago Press, 2002. ISBN: 978-0-226-12025-6 (cit. on p. 13).

[30] Jesús de la Cruz et al. "Functions of Ribosomal Proteins in Assembly of Eukaryotic Ribosomes In Vivo". In: *Annual Review of Biochemistry* 84.1 (June 2015), pp. 93–129. ISSN: 0066-4154, 1545-4509. DOI: 10.1146/annurev-biochem-060614-033917 (cit. on p. 11).

[31] Jelena Čuklina et al. "Review of Batch Effects Prevention, Diagnostics, and Correction Approaches". In: *Mass Spectrometry Data Analysis in Proteomics*. Ed. by Rune Matthiesen. Vol. 2051. New York, NY: Springer New York, 2020, pp. 373–387. ISBN: 978-1-4939-9743-5. DOI: 10.1007/978-1-4939-9744-2_16 (cit. on p. 120).

[32] Chad M. Cullen et al. "Emerging Priorities for Microbiome Research". In: *Frontiers in Microbiology* 11 (Feb. 2020), p. 136. ISSN: 1664-302X. DOI: 10.3389/fmicb.2020.00136 (cit. on p. 120).

[33] Marleide Da Mota Gomes. "Louis Pasteur and Dom Pedro II Engaged in Rabies Vaccine Development". In: *Journal of Preventive Medicine and Hygiene* (Apr. 2021), E231 Pages. DOI: 10.15167/2421-4248/JPMH2021.62.1.1631 (cit. on p. 14).

[34] Wendy J. Dahl et al. "Diet, Nutrients and the Microbiome". In: *Progress in Molecular Biology and Translational Science*. Vol. 171. Elsevier, 2020, pp. 237–263. ISBN: 978-0-12-820000-1. DOI: 10.1016/bs.pmbts.2020.04.006 (cit. on p. 16).

[35] Ralf Dahm. "Friedrich Miescher and the Discovery of DNA". In: *Developmental Biology* 278.2 (Feb. 2005), pp. 274–288. ISSN: 00121606. DOI: 10.1016/j.ydbio.2004.11.028 (cit. on p. 5).

[36] Zhenwei Dai et al. "Batch Effects Correction for Microbiome Data with Dirichlet-multinomial Regression". In: (2018), p. 8 (cit. on p. 39).

[37] Zhenwei Dai et al. "Batch Effects Correction for Microbiome Data with Dirichlet-multinomial Regression". In: *Bioinformatics* 35.5 (Mar. 2019). Ed. by Inanc Birol, pp. 807–814. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/bty729 (cit. on p. 120).

[38] Charles Darwin. *On the Origin of Species*. London: HarperPress, 2011. ISBN: 978-0-00-790223-1 (cit. on pp. 5, 10).

[39] Emma de Jong et al. "Unlocking Immune-Mediated Disease Mechanisms with Transcriptomics". In: *Biochemical Society Transactions* 49.2 (Apr. 2021), pp. 705–714. ISSN: 0300-5127, 1470-8752. DOI: 10.1042/BST20200652 (cit. on p. 15).

[40] Ayelet Di Segni et al. "Guided Protocol for Fecal Microbial Characterization by 16S rRNA-Amplicon Sequencing". In: *Journal of Visualized Experiments* 133 (Mar. 2018), p. 56845. ISSN: 1940-087X. DOI: 10.3791/56845 (cit. on p. 19).

[41] Ruth F. Dubin et al. "Proteomics and Metabolomics in Kidney Disease, Including Insights into Etiology, Treatment, and Prevention". In: *Clinical Journal of the American Society of Nephrology* 15.3 (Mar. 2020), pp. 404–411. ISSN: 1555-9041, 1555-905X. DOI: 10.2215/CJN.07420619 (cit. on p. 15).

[42] "Due Credit". In: *Nature* 496.7445 (Apr. 2013), pp. 270–270. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/496270a (cit. on p. 6).

[43] Martin Dworkin et al. "Sergei Winogradsky: A Founder of Modern Microbiology and the First Microbial Ecologist". In: *FEMS Microbiology Reviews* 36.2 (Mar. 2012), pp. 364–379. ISSN: 1574-6976. DOI: 10.1111/j.1574-6976.2011.00299.x (cit. on p. 14).

[44] Carl Eckart et al. "The Approximation of One Matrix by Another of Lower Rank". In: *Psychometrika* 1.3 (Sept. 1936), pp. 211–218. ISSN: 0033-3123, 1860-0980. DOI: 10.1007/BF02288367 (cit. on p. 49).

[45] Frank N. Egerton. "A History of the Ecological Sciences, Part 19: Leeuwenhoek's Microscopic Natural History". In: *Bulletin of the Ecological Society of America* 87.1 (Jan. 2006), pp. 47–58. ISSN: 0012-9623. DOI: 10.1890/0012-9623(2006)87[47:AHOTES]2.0.CO;2 (cit. on p. 10).

[46] Mohamed Elfil et al. “Sampling Methods in Clinical Research; an Educational Review”. In: *Emergency (Tehran, Iran)* 5.1 (2017), e52. ISSN: 2345-4563 (cit. on p. 19).

[47] Harold Ellis. “James Phipps, First to Be Vaccinated against Smallpox by Edward Jenner”. In: *Journal of Perioperative Practice* 31.1-2 (Jan. 2021), pp. 51–52. ISSN: 1750-4589, 2515-7949. DOI: 10.1177/1750458920950165 (cit. on p. 14).

[48] Teng Fei et al. “Mitigating the Adverse Impact of Batch Effects in Sample Pattern Detection”. In: *Bioinformatics* 34.15 (Aug. 2018). Ed. by Inanc Birol, pp. 2634–2641. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/bty117 (cit. on p. 32).

[49] Kristýna Fiedorová et al. “The Impact of DNA Extraction Methods on Stool Bacterial and Fungal Microbiota Community Recovery”. In: *Frontiers in Microbiology* 10 (Apr. 2019), p. 821. ISSN: 1664-302X. DOI: 10.3389/fmicb.2019.00821 (cit. on p. 19).

[50] B. M. Forde et al. “Next-Generation Sequencing Technologies and Their Impact on Microbial Genomics”. In: *Briefings in Functional Genomics* 12.5 (Sept. 2013), pp. 440–453. ISSN: 2041-2649, 2041-2657. DOI: 10.1093/bfgp/els062 (cit. on p. 15).

[51] Fiona Fouhy et al. “The Effects of Freezing on Faecal Microbiota as Determined Using MiSeq Sequencing and Culture-Based Investigations”. In: *PLOS ONE* 10.3 (Mar. 2015). Ed. by Josef Neu, e0119355. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0119355 (cit. on p. 19).

[52] George E. Fox et al. “Classification of Methanogenic Bacteria by 16S Ribosomal RNA Characterization”. In: *Proceedings of the National Academy of Sciences* 74.10 (Oct. 1977), pp. 4537–4541. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.74.10.4537 (cit. on p. 11).

[53] Christine Fülling et al. “Gut Microbe to Brain Signaling: What Happens in Vagus. . . ” In: *Neuron* 101.6 (Mar. 2019), pp. 998–1002. ISSN: 08966273. DOI: 10.1016/j.neuron.2019.02.008 (cit. on p. 16).

[54] J. A. Gagnon-Bartsch et al. “Using Control Genes to Correct for Unwanted Variation in Microarray Data”. In: *Biostatistics* 13.3 (July 2012), pp. 539–552. ISSN: 1465-4644, 1468-4357. DOI: 10.1093/biostatistics/kxr034 (cit. on pp. 56, 58).

[55] Luke C. Gandolfo et al. “RLE Plots: Visualizing Unwanted Variation in High Dimensional Data”. In: *PLOS ONE* 13.2 (Feb. 2018). Ed. by Enrique Hernandez-Lemus, e0191629. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0191629 (cit. on p. 93).

[56] John M. Gaspar. “NGmerge: Merging Paired-End Reads via Novel Empirically-Derived Models of Sequencing Errors”. In: *BMC Bioinformatics* 19.1 (Dec. 2018), p. 536. ISSN: 1471-2105. DOI: 10.1186/s12859-018-2579-2 (cit. on p. 26).

[57] Adriane Gelpi et al. “Magic Bullet: Paul Ehrlich, Salvarsan and the Birth of Venereology”. In: *Sexually Transmitted Infections* 91.1 (Feb. 2015), pp. 68.2–69. ISSN: 1368-4973, 1472-3263. DOI: 10.1136/sextrans-2014-051779 (cit. on p. 13).

[58] Robert C Gentleman et al. “Bioconductor: Open Software Development for Computational Biology and Bioinformatics”. In: *Genome Biology* 5.10 (2004), R80. ISSN: 14656906. DOI: 10.1186/gb-2004-5-10-r80 (cit. on p. 81).

[59] Sean M. Gibbons et al. “Correcting for Batch Effects in Case-Control Microbiome Studies”. In: *PLOS Computational Biology* 14.4 (Apr. 2018). Ed. by Morgan Langille, e1006102. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1006102 (cit. on p. 3).

[60] Richard A. Gibbs. “The Human Genome Project Changed Everything”. In: *Nature Reviews Genetics* 21.10 (Oct. 2020), pp. 575–576. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/s41576-020-0275-3 (cit. on p. 17).

[61] Yoav Gilad et al. “A Reanalysis of Mouse ENCODE Comparative Gene Expression Data”. In: *F1000Research* 4 (May 2015), p. 121. ISSN: 2046-1402. DOI: 10.12688/f1000research.6536.1 (cit. on p. 1).

[62] Gregory B. Gloor et al. “Microbiome Datasets Are Compositional: And This Is Not Optional”. In: *Frontiers in Microbiology* 8 (Nov. 2017), p. 2224. ISSN: 1664-302X. DOI: 10.3389/fmicb.2017.02224 (cit. on pp. 29, 37).

[63] Wilson Wen Bin Goh et al. "Why Batch Effects Matter in Omics Data, and How to Avoid Them". In: *Trends in Biotechnology* 35.6 (June 2017), pp. 498–507. ISSN: 01677799. DOI: 10.1016/j.tibtech.2017.02.012 (cit. on p. 120).

[64] G.H. Golub et al. "A Generalization of the Eckart-Young-Mirsky Matrix Approximation Theorem". In: *Linear Algebra and its Applications* 88–89 (Apr. 1987), pp. 317–327. ISSN: 00243795. DOI: 10.1016/0024-3795(87)90114-5 (cit. on p. 49).

[65] Fred Griffith. "The Significance of Pneumococcal Types". In: *Journal of Hygiene* 27.2 (Jan. 1928), pp. 113–159. ISSN: 0022-1724. DOI: 10.1017/S0022172400031879 (cit. on p. 6).

[66] Xavier A. Harrison et al. "A Brief Introduction to Mixed Effects Modelling and Multi-Model Inference in Ecology". In: *PeerJ* 6 (May 2018), e4794. ISSN: 2167-8359. DOI: 10.7717/peerj.4794 (cit. on pp. 50, 51).

[67] J. A. Hartigan et al. "Algorithm AS 136: A K-Means Clustering Algorithm". In: *Applied Statistics* 28.1 (1979), p. 100. ISSN: 00359254. DOI: 10.2307/2346830 (cit. on p. 54).

[68] James M. Heather et al. "The Sequence of Sequencers: The History of Sequencing DNA". In: *Genomics* 107.1 (Jan. 2016), pp. 1–8. ISSN: 08887543. DOI: 10.1016/j.ygeno.2015.11.003 (cit. on p. 17).

[69] Robert Hine. *A Dictionary of Biology*. 2015. ISBN: 978-0-19-105944-5 (cit. on p. 16).

[70] Robert Hooke. *Micrographia : Or, Some Physiological Descriptions of Minute Bodies Made by Magnifying Glasses. With Observations and Inquiries Thereupon*. London, England: J. Allestry, 1665 (cit. on p. 5).

[71] Ross Ihaka. *R : Past and Future History*. 1998 (cit. on p. 62).

[72] Jethro S. Johnson et al. "Evaluation of 16S rRNA Gene Sequencing for Species and Strain-Level Microbiome Analysis". In: *Nature Communications* 10.1 (Dec. 2019), p. 5029. ISSN: 2041-1723. DOI: 10.1038/s41467-019-13036-1 (cit. on p. 11).

[73] W. Evan Johnson et al. "Adjusting Batch Effects in Microarray Expression Data Using Empirical Bayes Methods". In: *Biostatistics* 8.1 (Jan. 2007), pp. 118–127. ISSN: 1468-4357, 1465-4644. DOI: 10.1093/biostatistics/kxj037 (cit. on p. 60).

[74] Sven Erik Jørgensen et al. *Encyclopedia of Ecology*. Amsterdam: Elsevier, 2008. ISBN: 978-0-08-045405-4 (cit. on p. 14).

[75] Leonard Kaufman et al. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series in Probability and Mathematical Statistics. New York: Wiley, 1990. ISBN: 978-0-471-87876-6 (cit. on p. 54).

[76] Raivo Kolde. *Pheatmap: Pretty Heatmaps*. 2019 (cit. on pp. 76, 78).

[77] Allan Konopka. "What Is Microbial Community Ecology?" In: *The ISME Journal* 3.11 (Nov. 2009), pp. 1223–1230. ISSN: 1751-7362, 1751-7370. DOI: 10.1038/ismej.2009.88 (cit. on p. 15).

[78] Alexandra Kuznetsova et al. "lmerTest Package: Tests in Linear Mixed Effects Models". In: *Journal of Statistical Software* 82.13 (2017). ISSN: 1548-7660. DOI: 10.18637/jss.v082.i13 (cit. on p. 74).

[79] Jean-Christophe Lagier et al. "Culturing the Human Microbiota and Culturomics". In: *Nature Reviews Microbiology* 16.9 (Sept. 2018), pp. 540–550. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/s41579-018-0041-0 (cit. on p. 15).

[80] Jean-Christophe Lagier et al. "Current and Past Strategies for Bacterial Culture in Clinical Microbiology". In: *Clinical Microbiology Reviews* 28.1 (Jan. 2015), pp. 208–236. ISSN: 0893-8512, 1098-6618. DOI: 10.1128/CMR.00110-14 (cit. on p. 10).

[81] Jean-Christophe Lagier et al. "Koch Postulate: Why Should We Grow Bacteria?" In: *Archives of Medical Research* 48.8 (Nov. 2017), pp. 774–779. ISSN: 01884409. DOI: 10.1016/j.arcmed.2018.02.003 (cit. on p. 13).

[82] Jean-Christophe Lagier et al. "The Rebirth of Culture in Microbiology through the Example of Culturomics To Study Human Gut Microbiota". In: *Clinical Microbiology Reviews* 28.1 (Jan. 2015), pp. 237–264. ISSN: 0893-8512, 1098-6618. DOI: 10.1128/CMR.00014-14 (cit. on p. 10).

[83] Kim-Anh Lê Cao et al. *Managing Batch Effects in Microbiome Data*. Oct. 2019 (cit. on p. 3).

[84] Joshua Lederberg et al. "Ome Sweet Omics–A Genealogical Treasury of Words". In: *The Scientist* 15.7 (Apr. 2001), p. 8. ISSN: 08903670 (cit. on pp. 1, 9, 15).

[85] Jeffrey T Leek et al. "Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis". In: *PLoS Genetics* 3.9 (Sept. 2007). Ed. by Greg Gibson, e161. ISSN: 1553-7404. DOI: 10.1371/journal.pgen.0030161 (cit. on pp. 56, 58).

[86] Jeffrey T. Leek et al. *sva: Surrogate Variable Analysis*. R package version 3.42.0. 2021 (cit. on pp. 2, 74).

[87] Jeffrey T. Leek et al. "Tackling the Widespread and Critical Impact of Batch Effects in High-Throughput Data". In: *Nature Reviews Genetics* 11.10 (Oct. 2010), pp. 733–739. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/nrg2825 (cit. on pp. 35, 36, 41).

[88] Jianying Li et al. "Principal Variance Components Analysis: Estimating Batch Effects in Microarray Gene Expression Data". In: *Batch Effects and Noise in Microarray Experiments*. Ed. by Andreas Scherer. Chichester, UK: John Wiley & Sons, Ltd, Oct. 2009, pp. 141–154. ISBN: 978-0-470-68598-3. DOI: 10.1002/9780470685983.ch12 (cit. on pp. 52, 53).

[89] Guanxiang Liang et al. "The Human Virome: Assembly, Composition and Host Interactions". In: *Nature Reviews Microbiology* 19.8 (Aug. 2021), pp. 514–527. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/s41579-021-00536-5 (cit. on p. 15).

[90] Aristidis Likas et al. "The Global K-Means Clustering Algorithm". In: *Pattern Recognition* 36.2 (Feb. 2003), pp. 451–461. ISSN: 00313203. DOI: 10.1016/S0031-3203(02)00060-2 (cit. on p. 54).

[91] Charles E. Mackenzie. *Coded Character Sets: History and Development*. Reading, Mass: Addison-Wesley Pub. Co, 1980. ISBN: 978-0-201-14460-4 (cit. on pp. 22, 23).

[92] Jean M. Macklaim et al. "From RNA-seq to Biological Inference: Using Compositional Data Analysis in Meta-Transcriptomics". In: *Microbiome Analysis*. Ed. by Robert G. Beiko et al. Vol. 1849. New York, NY: Springer New York, 2018, pp. 193–213. ISBN: 978-1-4939-8726-9. DOI: 10.1007/978-1-4939-8728-3_13 (cit. on pp. 2, 3, 28).

[93] Martin Maechler et al. *Cluster: Cluster Analysis Basics and Extensions*. 2021 (cit. on pp. 74, 75).

[94] Solaiappan Manimaran et al. "BatchQC: Interactive Software for Evaluating Sample and Batch Effects in Genomic Data". In: *Bioinformatics* 32.24 (Dec. 2016), pp. 3836–3838. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btw538 (cit. on p. 3).

[95] Mansi El-Mansi, ed. *Fermentation Microbiology and Biotechnology*. 3rd ed. Boca Raton, FL: Taylor & Francis/CRC Press, 2012. ISBN: 978-1-4398-5579-9 (cit. on p. 12).

[96] Midhuna Immaculate Joseph Maran et al. "Benefits of Merging Paired-End Reads before Pre-Processing Environmental Metagenomics Data". In: *Marine Genomics* 61 (Feb. 2022), p. 100914. ISSN: 18747787. DOI: 10.1016/j.margen.2021.100914 (cit. on p. 26).

[97] Ernst Mayr. *The Growth of Biological Thought: Diversity, Evolution, and Inheritance*. Cambridge, Mass.: Harvard Univ. Pr, 1982. ISBN: 978-0-674-36446-2 (cit. on p. 10).

[98] Paul J. McMurdie et al. "Phyloseq: An R Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data". In: *PLoS ONE* 8.4 (Apr. 2013). Ed. by Michael Watson, e61217. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0061217 (cit. on pp. 41, 73, 74, 82).

[99] F. Menges et al. "TOTALRECALLER: Improved Accuracy and Performance via Integrated Alignment and Base-Calling". In: *Bioinformatics* 27.17 (Sept. 2011), pp. 2330–2337. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btr393 (cit. on p. 31).

[100] *Microbiome Informatics: OTU vs. ASV*. URL: https://www.zymoresearch.com/blogs/blog/microbiome-informatics-otu-vs-asv (cit. on p. 26).

[101] John L. Mohr. "Protozoa as Indicators of Pollution". In: *The Scientific Monthly* 74.1 (1952), pp. 7–9. ISSN: 00963771. URL: http://www.jstor.org/stable/20835 (cit. on p. 1).

[102] Ramyar Molania et al. "A New Normalization for Nanostring nCounter Gene Expression Data". In: *Nucleic Acids Research* 47.12 (July 2019), pp. 6073–6083. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkz433 (cit. on pp. 59, 74, 75).

[103] Martin Morgan. *BiocManager: Access the Bioconductor Project Package Repository*. R package version 1.30.16. 2021. URL: https://CRAN.R-project.org/package=BiocManager (cit. on pp. 70, 71).

[104] David W. Mount. *Bioinformatics: Sequence and Genome Analysis*. Cold Spring Harbor, N.Y: Cold Spring Harbor Laboratory Press, 2001. ISBN: 978-0-87969-655-9 (cit. on p. 26).

[105] Kirill Müller et al. *Tibble: Simple Data Frames*. 2021 (cit. on p. 73).

[106] Robert Müller et al. "On the Use of Sequence-Quality Information in OTU Clustering". In: *PeerJ* 9 (Aug. 2021), e11717. ISSN: 2167-8359. DOI: 10.7717/peerj.11717 (cit. on p. 27).

[107] Andrea K. Nash et al. "The Gut Mycobiome of the Human Microbiome Project Healthy Cohort". In: *Microbiome* 5.1 (Nov. 2017), p. 153. ISSN: 2049-2618. DOI: 10.1186/s40168-017-0373-4 (cit. on p. 16).

[108] *NGS Basics - DE*. URL: %5Curl%7Bwww.thermofisher.com/de/de/home/life-science/sequencing/sequencing-learning-center/next-generation-sequencing-information/ngs-basics.html%7D (cit. on pp. 20, 21).

[109] William S Noble. "How Does Multiple Testing Correction Work?" In: *Nature Biotechnology* 27.12 (Dec. 2009), pp. 1135–1137. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt1209-1135 (cit. on p. 56).

[110] Vegard Nygaard et al. "Methods That Remove Batch Effects While Retaining Group Differences May Lead to Exaggerated Confidence in Downstream Analyses". In: *Biostatistics* 17.1 (Jan. 2016), pp. 29–39. ISSN: 1468-4357, 1465-4644. DOI: 10.1093/biostatistics/kxv027 (cit. on pp. 38, 60, 120).

[111] Robert C. Olby. *Francis Crick: Hunter of Life's Secrets*. Cold Spring Harbor, N.Y: Cold Spring Harbor Laboratory Press, 2008. ISBN: 978-0-87969-798-3 (cit. on p. 6).

[112] Andrzej Oleś. *BiocStyle: Standard styles for vignettes and other Bioconductor documents*. R package version 2.22.0. 2021. URL: `https://github.com/Bioconductor/BiocStyle` (cit. on pp. 70, 71).

[113] Bioconductor Package Maintainer et al. *BiocCheck: Bioconductor-specific package checks*. R package version 1.30.0. 2021. URL: `https://github.com/Bioconductor/BiocCheck/issues` (cit. on p. 71).

[114] Marina Panek et al. "Methodology Challenges in Studying Human Gut Microbiota – Effects of Collection, Storage, DNA Extraction and next Generation Sequencing Technologies". In: *Scientific Reports* 8.1 (Dec. 2018), p. 5143. ISSN: 2045-2322. DOI: `10.1038/s41598-018-23296-4` (cit. on pp. 31, 32).

[115] Chandra Shekhar Pareek et al. "Sequencing Technologies and Genome Sequencing". In: *Journal of Applied Genetics* 52.4 (Nov. 2011), pp. 413–435. ISSN: 1234-1983, 2190-3883. DOI: `10.1007/s13353-011-0057-x` (cit. on p. 21).

[116] Marta Paterlini. "There Shall Be Order: The Legacy of Linnaeus in the Age of Molecular Biology". In: *EMBO reports* 8.9 (Sept. 2007), pp. 814–816. ISSN: 1469-221X, 1469-3178. DOI: `10.1038/sj.embor.7401061` (cit. on pp. 12, 15).

[117] V. Pawlowsky-Glahn et al. "Compositional Data and Their Analysis: An Introduction". In: *Geological Society, London, Special Publications* 264.1 (2006), pp. 1–10. ISSN: 0305-8719, 2041-4927. DOI: `10.1144/GSL.SP.2006.264.01.01` (cit. on p. 29).

[118] Diego G. Peroni et al. "Microbiome Composition and Its Impact on the Development of Allergic Diseases". In: *Frontiers in Immunology* 11 (Apr. 2020), p. 700. ISSN: 1664-3224. DOI: `10.3389/fimmu.2020.00700` (cit. on p. 15).

[119] Anton S. Petrov et al. "History of the Ribosome and the Origin of Translation". In: *Proceedings of the National Academy of Sciences* 112.50 (Dec. 2015), pp. 15396–15401. ISSN: 0027-8424, 1091-6490. DOI: `10.1073/pnas.1509761112` (cit. on p. 8).

[120] Monica Pichler et al. "A 16S rRNA Gene Sequencing and Analysis Protocol for the Illumina MiniSeq Platform". In: *MicrobiologyOpen* 7.6 (Dec. 2018), e00611. ISSN: 2045-8827, 2045-8827. DOI: 10.1002/mbo3.611 (cit. on p. 20).

[121] Dorota L. Porazinska et al. "The Nature and Frequency of Chimeras in Eukaryotic Metagenetic Samples". In: *Journal of Nematology* 44.1 (Mar. 2012), pp. 18–25. ISSN: 0022-300X (cit. on p. 26).

[122] Plinio Prioreschi. *A History of Medicine*. 1st ed. Mellen History of Medicine v. 1-3, 5. Lewiston [N.Y.]: Edwin Mellen Press, 1991. ISBN: 978-0-7734-9661-3 (cit. on p. 9).

[123] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2021. URL: https://www.R-project.org/ (cit. on pp. 61, 68, 69, 71, 74, 76).

[124] Tonse NK Raju. "The Nobel Chronicles". In: *The Lancet* 353.9156 (Mar. 1999), p. 936. ISSN: 01406736. DOI: 10.1016/S0140-6736(05)75055-8 (cit. on p. 14).

[125] Marta Reyman et al. "Impact of Delivery Mode-Associated Gut Microbiota Dynamics on Health in the First Year of Life". In: *Nature Communications* 10.1 (Dec. 2019), p. 4997. ISSN: 2041-1723. DOI: 10.1038/s41467-019-13014-7 (cit. on p. 16).

[126] Maria Vittoria Ristori et al. "Proteomics and Metabolomics Approaches towards a Functional Insight onto AUTISM Spectrum Disorders: Phenotype Stratification and Biomarker Discovery". In: *International Journal of Molecular Sciences* 21.17 (Aug. 2020), p. 6274. ISSN: 1422-0067. DOI: 10.3390/ijms21176274 (cit. on p. 15).

[127] Matthew E. Ritchie et al. "Limma Powers Differential Expression Analyses for RNA-sequencing and Microarray Studies". In: *Nucleic Acids Research* 43.7 (Apr. 2015), e47–e47. ISSN: 1362-4962, 0305-1048. DOI: 10.1093/nar/gkv007 (cit. on pp. 2, 74, 75).

[128] Somak Roy et al. "Standards and Guidelines for Validating Next-Generation Sequencing Bioinformatics Pipelines". In: *The Journal of Molecular Diagnostics* 20.1 (Jan. 2018), pp. 4–27. ISSN: 15251578. DOI: 10.1016/j.jmoldx.2017.11.003 (cit. on p. 24).

[129] Manikandan S. "Measures of Central Tendency: Median and Mode". In: *Journal of Pharmacology and Pharmacotherapeutics* 2.3 (Sept. 2011), pp. 214–215. ISSN: 0976-500X, 0976-5018. DOI: 10.4103/0976-500X.83300 (cit. on p. 44).

[130] Oriol Sacristán-Soriano et al. "Exploring the Links between Natural Products and Bacterial Assemblages in the Sponge *Aplysina Aerophoba*". In: *Applied and Environmental Microbiology* 77.3 (Feb. 2011), pp. 862–870. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.00100-10 (cit. on p. 65).

[131] Milton H. Saier. "Understanding the Genetic Code". In: *Journal of Bacteriology* 201.15 (Aug. 2019). Ed. by William Margolin. ISSN: 0021-9193, 1098-5530. DOI: 10.1128/JB.00091-19 (cit. on p. 7).

[132] F. Sanger et al. "DNA Sequencing with Chain-Terminating Inhibitors". In: *Proceedings of the National Academy of Sciences* 74.12 (Dec. 1977), pp. 5463–5467. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.74.12.5463 (cit. on pp. 17, 21).

[133] Mark P. Sawicki et al. "Human Genome Project". In: *The American Journal of Surgery* 165.2 (Feb. 1993), pp. 258–264. ISSN: 00029610. DOI: 10.1016/S0002-9610(05)80522-7 (cit. on p. 17).

[134] Andreas Scherer, ed. *Batch Effects and Noise in Microarray Experiments: Sources and Solutions*. Wiley Series in Probability and Statistics. Chichester, U.K: J. Wiley, 2009. ISBN: 978-0-470-74138-2 (cit. on pp. 19, 28, 34, 35).

[135] Jeffery A. Schloss et al. "Cultivating DNA Sequencing Technology After the Human Genome Project". In: *Annual Review of Genomics and Human Genetics* 21.1 (Aug. 2020), pp. 117–138. ISSN: 1527-8204, 1545-293X. DOI: 10.1146/annurev-genom-111919-082433 (cit. on p. 17).

[136] Conrad L. Schoch et al. "Nuclear Ribosomal Internal Transcribed Spacer (ITS) Region as a Universal DNA Barcode Marker for *Fungi*". In: *Proceedings of the National Academy of Sciences* 109.16 (Apr. 2012), pp. 6241–6246. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1117018109 (cit. on p. 11).

[137] Scientific American et al. "Celebrating The Genetic Jubilee: A Conversation with James D.Watson". In: *Scientific American* 288.4 (2003), pp. 66–69. ISSN: 00368733, 19467087 (cit. on p. 6).

[138] Shailesh K. Shahi et al. "Microbiota Analysis Using Two-step PCR and Next-generation 16S rRNA Gene Sequencing". In: *Journal of Visualized Experiments* 152 (Oct. 2019), p. 59980. ISSN: 1940-087X. DOI: 10.3791/59980 (cit. on pp. 19, 20).

[139] Gavin L. Simpson. *Permute: Functions for Generating Restricted Permutations of Data*. 2019 (cit. on p. 74).

[140] Andrew H Sims et al. "The Removal of Multiplicative, Systematic Bias Allows Integration of Breast Cancer Gene Expression Datasets – Improving Meta-Analysis and Prediction of Prognosis". In: *BMC Medical Genomics* 1.1 (Dec. 2008), p. 42. ISSN: 1755-8794. DOI: 10.1186/1755-8794-1-42 (cit. on p. 60).

[141] Gregg W C Thomas et al. "Referee: Reference Assembly Quality Scores". In: *Genome Biology and Evolution* 11.5 (May 2019). Ed. by Belinda Chang, pp. 1483–1486. ISSN: 1759-6653. DOI: 10.1093/gbe/evz088 (cit. on p. 23).

[142] Hoa Thi Nhu Tran et al. "A Benchmark of Batch-Effect Correction Methods for Single-Cell RNA Sequencing Data". In: *Genome Biology* 21.1 (Dec. 2020), p. 12. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1850-9 (cit. on p. 120).

[143] *Translational Systems Medicine and Oral Disease*. Elsevier, 2020. ISBN: 978-0-12-813762-8. DOI: 10.1016/C2017-0-00348-8 (cit. on p. 16).

[144] Andrew Travers et al. "DNA Structure and Function". In: *FEBS Journal* 282.12 (June 2015), pp. 2279–2295. ISSN: 1742464X. DOI: 10.1111/febs.13307 (cit. on p. 7).

[145] Peter C.B. Turnbull. "Anthrax Vaccines: Past, Present and Future". In: *Vaccine* 9.8 (Aug. 1991), pp. 533–539. ISSN: 0264410X. DOI: 10.1016/0264-410X(91)90237-Z (cit. on p. 13).

[146] Lary Walker et al. "Koch's Postulates and Infectious Proteins". In: *Acta Neuropathologica* 112.1 (July 2006), pp. 1–4. ISSN: 0001-6322, 1432-0533. DOI: 10.1007/s00401-006-0072-x (cit. on p. 13).

[147] Yiwen Wang et al. "Managing Batch Effects in Microbiome Data". In: *Briefings in Bioinformatics* 21.6 (Dec. 2020), pp. 1954–1970. ISSN: 1477-4054. DOI: 10.1093/bib/bbz105 (cit. on pp. 31, 120).

[148] J. D. Watson et al. "Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid". In: *Nature* 171.4356 (Apr. 1953), pp. 737–738. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/171737a0 (cit. on p. 6).

[149] John M Whipps et al. "Commercial use of fungi as plant disease biological control agents: status and prospects". In: *Fungal biocontrol agents: progress, problems and potential* (2001), pp. 9–22 (cit. on p. 1).

[150] Hadley Wickham. *Ggplot2: Elegant Graphics for Data Analysis*. Second edition. Use R! Switzerland: Springer, 2016. ISBN: 978-3-319-24277-4. DOI: 10.1007/978-3-319-24277-4 (cit. on p. 78).

[151] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN: 978-3-319-24277-4. URL: https://ggplot2.tidyverse.org (cit. on p. 76).

[152] Hadley Wickham. "testthat: Get Started with Testing". In: *The R Journal* 3 (2011), pp. 5–10. URL: https://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf (cit. on pp. 70, 71).

[153] Hadley Wickham. *Tidyr: Tidy Messy Data*. 2021 (cit. on p. 73).

[154] Hadley Wickham et al. *devtools: Tools to Make Developing R Packages Easier*. R package version 2.4.3. 2021. URL: https://CRAN.R-project.org/package=devtools (cit. on pp. 68, 71).

[155] Hadley Wickham et al. *Dplyr: A Grammar of Data Manipulation*. 2021 (cit. on p. 73).

[156] Hadley Wickham et al. *roxygen2: In-Line Documentation for R*. R package version 7.1.2. 2021. URL: https://CRAN.R-project.org/package=roxygen2 (cit. on p. 71).

[157] Hadley Wickham et al. *usethis: Automate Package and Project Setup*. R package version 2.1.5. 2021. URL: https://CRAN.R-project.org/package=usethis (cit. on p. 71).

[158] Hadley Wickham et al. "Welcome to the Tidyverse". In: *Journal of Open Source Software* 4.43 (Nov. 2019), p. 1686. ISSN: 2475-9066. DOI: 10.21105/joss.01686 (cit. on p. 74).

[159] Hadley Wickham et al. "Welcome to the tidyverse". In: *Journal of Open Source Software* 4.43 (2019), p. 1686. DOI: 10.21105/joss.01686 (cit. on p. 74).

[160] Jan Witkowski. "The Forgotten Scientists Who Paved the Way to the Double Helix". In: *Nature* 568.7752 (Apr. 2019), pp. 308–309. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/d41586-019-01176-9 (cit. on p. 6).

[161] Yinglin Xia. "Correlation and Association Analyses in Microbiome Study Integrating Multiomics in Health and Disease". In: *Progress in Molecular Biology and Translational Science*. Vol. 171. Elsevier, 2020, pp. 309–491. ISBN: 978-0-12-820000-1. DOI: 10.1016/bs.pmbts.2020.04.003 (cit. on p. 51).

[162] Yihui Xie. *Dynamic Documents with R and knitr*. 2nd. ISBN 978-1498716963. Boca Raton, Florida: Chapman and Hall/CRC, 2015. URL: https://yihui.org/knitr/ (cit. on p. 78).

[163] Yihui Xie. "knitr: A Comprehensive Tool for Reproducible Research in R". In: *Implementing Reproducible Computational Research*. Ed. by Victoria Stodden et al. ISBN 978-1466561595. Chapman and Hall/CRC, 2014. URL: http://www.crcpress.com/product/isbn/9781466561595 (cit. on p. 78).

[164] Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.38. 2022. URL: https://yihui.org/knitr/ (cit. on p. 78).

[165] Yihui Xie et al. *R Markdown Cookbook*. ISBN 9780367563837. Boca Raton, Florida: Chapman and Hall/CRC, 2020. URL: https://bookdown.org/yihui/rmarkdown-cookbook (cit. on p. 78).

[166] Yihui Xie et al. *R Markdown: The Definitive Guide*. ISBN 9781138359338. Boca Raton, Florida: Chapman and Hall/CRC, 2018. URL: https://bookdown.org/yihui/rmarkdown (cit. on p. 78).

[167] In Seok Yang et al. "Analysis of Whole Transcriptome Sequencing Data: Workflow and Software". In: *Genomics & Informatics* 13.4 (Dec. 2015), pp. 119–125. ISSN: 1598-866X. DOI: 10.5808/GI.2015.13.4.119 (cit. on p. 93).

[168] Feng Yue et al. "A Comparative Encyclopedia of DNA Elements in the Mouse Genome". In: *Nature* 515.7527 (Nov. 2014), pp. 355–364. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature13992 (cit. on p. 35).

[169] Longjian Zhou et al. "Examining the Practical Limits of Batch Effect-Correction Algorithms: When Should You Care about Batch Effects?" In: *Journal of Genetics and Genomics* 46.9 (Sept. 2019), pp. 433–443. ISSN: 16738527. DOI: 10.1016/j.jgg.2019.08.002 (cit. on p. 120).