NATIONAL PARKS AUSTRIA

Nationalpark Hohe Tauern

Langzeitmonitoring von Ökosystemprozessen - Methoden-Handbuch

Modul 04: Bodenmikrobiologie (Version: 07/2019)

Mit Unterstützung von Bund und Europäischer Union

Bundesministerium Nachhaltigkeit und Tourismus

LE 14-20

Europäischer Landwirtschaftsfonds für die Entwicklung des ländlichen Raums: Hier investiert Europa in die ländlichen Gebiete.

Impressum
Für den Inhalt verantwortlich: Dr. Fernando Fernández Mendoza & Prof. Mag Dr. Martin Grube
Institut für Biologie, Bereich Pflanzenwissenschaften, Universität Graz, Holteigasse 6, 8010 Graz

Nationalparkrat Hohe Tauern, Kirchplatz 2, 9971 Matrei i.O.

Titelbild: Ein Transekt im Untersuchungsgebiet Innergschlöss (2350 m üNN) wird im Jahr 2017 beprobt. © Newesely

Zitiervorschlag: Fernández Mendoza F, Grube M (2019) Langzeitmonitoring von Ökosystemprozessen im Nationalpark Hohe Tauern. Modul 04: Mikrobiologie. Methoden-Handbuch. Verlag der Österreichischen Akademie der Wissenschaften, Wien. ISBN-Online: 978-3-7001-8752-3, doi: 10.1553/GCP_LZM_NPHT_Modul04

Weblink: https://verlag.oeaw.ac.at und http://www.parcs.at/npht/mmd_fullentry.php?docu_id=38612

Inhaltsverzeichnis

Zielsetzung

Ziel dieses wissenschaftlichen Pilot-Projektes ist die methodische Entwicklung, Ersteinrichtung und Validierung eines interdisziplinären, integrativen Monitoring- und Forschungsprogramms, welches in den relevanten Disziplinen im Rahmen von „Modulen“ (Teil-Projekten) abgewickelt werden soll. Der in diesem Projekt verfolgte Forschungsansatz ist in dieser Form neuartig und innovativ. Die Vernetzung der Untersuchungen aus unterschiedlichen Disziplinen im selben Untersuchungsgebiet, stellt einen neuartigen, integrativen Ansatz dar, der auf dieser inhaltlichen als auch räumlichen Skala noch nirgends umgesetzt wurde.

Ziel des gegenständlichen Dokuments ist es einen ausführlichen methodischen Textbeitrag über das Monitoring im Modul 04 „Bodenmikrobiologie“, einem Teilbereich des Projektes „Langzeitmonitoring terrestrischer alpiner Ökosysteme im Nationalpark Hohe Tauern“, zu präsentieren. Damit soll es auch nach Abschluss der Pilotphase möglich sein, nach den gleichen experimentellen und analytischen Protokollen vorzugehen, damit eine homogene Datenauswertung auch in der langfristigen Fortführung des Projektes gewährleistet ist.

Ziel des Moduls 4 „Mikrobiologie“ ist eine Erhebung der mikrobiellen Vielfalt der Monitoringflächen. Die in diesem Teilprojekt berücksichtigte mikrobielle Vielfalt bezieht sich auf die Bakterien (und teilweise miterfassten Archaeen, die manchmal auch als Urbakterien bekannt sind), sowie die Pilze. Im Gegensatz zu früher verwendeten Methoden der Kultivierung haben wir uns dazu entschlossen einen kultivierungsunabhängigen Ansatz zur Feststellung der Diversität auszuwählen. Dies wurde so entschieden, weil die überwiegende Mehrheit (weit über 90%) der Bodenmikroorganismen nicht oder nur extrem langsam in Rein-Kulturen wachsen. Das liegt zum einen daran, dass sie bereits unter ihren natürlichen Bedingungen extrem langsam wachsen oder dort mit anderen Organismen in engen Beziehungen leben, die in Kultivierungsversuchen nicht simuliert werden können. Die Diversität dieser Mikroorganismen wurde daher durch Extraktion und anschließende Sequenzierung der DNA aus Bodenproben festgestellt. Somit liefert dieser Ansatz ein wesentlich vollständigeres Bild über die mikrobielle Vielfalt im Boden. Mithilfe der DNA Sequenzdaten ist es ausserdem möglich, die Bodenmikroorganismen gleichzeitig relativ gut taxonomisch einzuordnen, was sich bei kultivierungsabhängigen Verfahren viel langwieriger gestaltet. Es ist daher auch kaum überraschend, dass die DNA-Sequenzierung zur Methode der Wahl in der mikrobiellen Diversitätsforschung geworden ist. Hieraus resultiert noch ein weiterer Vorteil, denn man kann in gewissem Rahmen auch die Ergebnisse anderer Forscher mit den eigenen Ergebnissen vergleichen.

In der Pilotprojektphase des Moduls 4 „Bodenmikrobiologie“ sollte weiters festgestellt werden, ob sich entlang der Gradienten Unterschiede der mikrobiellen Vielfalt zeigen, bzw. ob derzeit Unterschiede in der Diversität zwischen den einzelnen Standorten des Monitoringprojektes zu finden sind. Die im gegenständlichen Modul gewählten Methoden der Sequenzdatenanalyse sind für diese Zwecke optimierte Verfahren der Mikrobiologie, um derartige Unterschiede nachzuweisen. In der Pilotphase wurden die Standorte nur einmal beprobt, weil es zu Projektbeginn für eine langzeitliche Untersuchung nicht zielführend erschien, engmaschiger zu beproben, solange nicht klare Veränderungen in den anderen Modulen erkennbar sind. Daher und wegen der damit verbundenen logistischen Schwierigkeiten der Probenahme wurden auch kurzzeitige Fluktuationen, wie sie zwischen den Jahreszeiten auftreten mögen nicht berücksichtigt.

Erst die Weiterführung dieses Moduls über längere Zeiträume wird zeigen, inwieweit sich langfristige Veränderungen der mikrobiellen Vielfalt ergeben. Es ist durchaus anzunehmen, dass sich mit einer Veränderung der Vegetationsdecke bzw. Veränderungen des Wassergehaltes im Boden, bzw. anderer physikalisch-chemischer Bedingungen die mikrobielle Zusammensetzung verändern wird. Es wird daher in Zukunft auch wichtig werden, die Daten in Zusammenhang mit den anderen Modulen wie etwa der Bodenzoologie, Bodenphysik, und vor allem auch der Vegetationsanalyse zu untersuchen.

Vorbereitungsarbeit und benötigtes Material

a. Materialien für die Probenahme und Probenaufbewahrung

Spatel (Bartelt GmbH, Graz), Eppendorfgefäße (Fassungsvermögen 2ml, Bartelt GmbH, Graz), Sarstedt Röhrchen (Fassungsvermögen 50 ml, Bartelt GmbH, Graz), Wasserflasche (Bartelt GmbH, Graz) mit sterilem Wasser.
Sterile Reinigungstücher (Kimtech Science Precision Wipes, Merck, Wien), Eisbox (Styropor) mit Eis, Trockeneisbox (Styropor) mit Trockeneis für den Transport.

b. Materialien und Geräte für die Laboranalyse

DNeasy PowerSoil Pro DNA Extraction Kit (Qiagen, Wien), PCR Primer (die für die notwendige Vervielfältigung der DNA herangezogenen Primer zielten auf die variable Region der 16S ribosomalen RNA für Prokaryoten ab, während für Pilze die ITS2-Spacer-Region der ribosomalen RNA verwendet wurde).

Tabelle 1. Primer für die PCR-Amplifikation von Bakterien- und Pilzgemeinschaften in den Bodenproben.

Primer Namen	Primersequenz (5'-3')	Zielregionen	Sequenzlänge (bp)	Literatur
341F_ill 802R_ill	CCTACGGGNGGCWGCAG GACTACHVGGGTATCTAATCC	V3 and V4	460	(Quast, Klindworth, et al. 2013)
ITS3 ITS4	GCATCGATGAAGAACGCAGC TCCTCCGCTTATTGATATGC	ITS2	300-400	(White et al. 1990)

Die für die Laboranalysen verwendeten Geräte (etwa Zentrifugen, PCR Geräte, Gelelektrophorese u.Ä.) entsprechend der Standardausrüstung eines molekularbiologischen Labors und werden hier nicht einzeln aufgeschlüsselt.

Arbeitsablauf

a. Lage der Untersuchungs- und Probestellen

Die beauftragten Arbeiten folgen den Richtlinien der von C. Körner konzipierten Präambel zum Manual (Körner 2019). In jedem der drei Nationalpark-Teile wurden 2016 Untersuchungsorte definiert, an der mehrere (3-6) Daueruntersuchungsflächen (Transekte, Dauerbeobachtungsflächen, DF) eingerichtet wurden. Ein derartiger Transekt umfasst idealerweise drei aneinandergrenzende Streifen von etwa 10 m Länge und 1 m Breite, die Gradienten von pessimalen Stellen (offene Vegetation) bis zu optimalen Lebensbedingungen (voll entwickelter alpiner Rasen) darstellen. Der mittlere Streifen bleibt ungestört, die links und rechts angrenzenden Streifen dienen der invasiven Beprobung (Bodenmikrobiologie).

Die Standortsauswahl und Standortsvermarkung erfolgte in einer konzertierten Aktion durch die Arbeitsgruppen „Boden u. Biomasse" (Körner, Tappeiner), „Vegetation" (Wittmann), „Bodenmesofauna" (Meyer) und „Bodenmikrobiologie" (Grube). Die Vorgangsweise und Feinstruktur der DFs sind im „Standardprotokoll für die Dauerbeobachtungsflächen" (Konzept Körner 2015) detailiert beschrieben. Die genaue Beschreibung der Untersuchungsorte, der Transekte, sowie das Probenahmeprotokoll finden sich in Newesely, Tappeiner & Körner (2019). Um Redundanzen zu vermeiden, verzichten wir deshalb auf eine genaue Beschreibung an dieser Stelle.

Beprobung

Umwelt-DNA (Environmental DNA) wurde von den gesammelten Proben extrahiert. Die daraus gewonnenen Sequenzdaten ergaben die Datengrundlage um geographische Trends des Bodenmikrobioms festzustellen, sowie Trends, die mit Unterschieden in Mikrohabitaten entlang eines topologischen Gradienten in den Beprobungsflächen korrelieren. Insgesamt wurden 45 Proben von den pessimalen Gründen der Gradienten gewonnen, 46 Proben wurden entlang des Gradienten entnommen und 50 and der oberen Kante der Gradienten, was den Vegetations-Optima entsprach.

In den in Newesely et al. 2019 abgebildeten Protokollen wird die exakte Lage der Beprobungsstellen wiedergegeben. An diesen Orten wurden die genauen Orte der Bodenkernbeprobung entsprechend Newesely et al. 2019 ausgewählt, um Proben von den

oberen (K), intermediären (M) und unteren Fraktionen der Transekten zu erhalten. Die Beprobung erfolgte nach der Entnahme der Bohrkerne für die Bodenphysik/Bodenchemie an den gleichen Stellen. Dafür wurden Proben aus einer Tiefe von etwa 3 cm unter der Erdoberfläche folendermaßen gesammelt: Zuerst wurde die durch die Kernbohrung entstandene Oberfläche vorsichtig abgeschabt um Kontaminationen durch das Bohrwerkzeug zu vermeiden, dann wurde an dieser Stelle mit dem zuvor sterilisierten Spaten das Volumen von ca. 1,5 ml Erde entnommen und in 1,5 ml Reaktionsgefäße überführt. Die gefüllten Gefäße wurden in einem 50 ml fassenden Sarstedt-Röhrchen zusammengefasst und nach der Entnahme sofort auf Eis gelagert. Für den Transport zum Labor und danach bis zur weiteren Bearbeitung wurden die gesammelten Proben bei -80 ° C (Trockeneis) eingefroren.

Umwelt-DNA Extraktion und Sequenzierung.
Zur Extraktion wurden die einzelnen Proben zuerst mit einem TissueLyser II (Retsch, erhältich über Qiagen, Wien) gemahlen, bevor die DNA-Extraktion mit dem DNeasy PowerSoil Pro DNA Extraction Kit (Qiagen, Wien) erfolgte.

Eine auf Amplifikaten basierende Sequenzierungsstrategie wurde für Standard-Barcode-Marker für Bakterien und Pilze verwendet. Das bedeutet, dass vor der Sequenzierung eine PCR (Polymerase Chain Reaction) durchgeführt wurde, um aus der DNA Probe die entsprechende Fraktion der Bakterien bzw. Pilze selektiv zu vervielfältigen. Die Primer für die DNA Amplifikation zielten auf die variable Region der 16S ribosomalen RNA für Bakterien und Archaea ab, während für Pilze die ITS2-Spacer-Region der ribosomalen RNA verwendet wurde. Für beide Amplikons wurde ein Nextera-Zwei-Schritt-PCR-Verfahren zur Bibliotheksvorbereitung verwendet. Amplicon-Bibliotheken wurden vor der Sequenzierung später gereinigt und äquimolar gepoolt. Die Sequenzierung wurde auf einer Illumina MiSeq-Plattform unter Verwendung von v2-Chemie und gepaarten 250-bp-Endläufen durchgeführt. Die Vorbereitung der Amplicon-Bibliothek, die Sequenzierung sowie das Trimmen und Adaptieren von Adaptern wurde von Microsynth GMBH, Schweiz, durchgeführt.

Datenverarbeitung

a. Verarbeitung von ITS und 16S Amplicons

Die Daten wurden unter Verwendung einer für das Projekt spezifisch entwickelten Bioinformatik-Pipeline verarbeitet. Sie ermöglichte die Untersuchung der operationellen taxonomischen Einheiten (OTUs, operational taxonomic units) im Rahmen eines soziologisch-ökologischen Ansatzes für Mikroorganismen. Wir verwenden bei diesen Untersuchungen den Begriff OTUs deshalb, weil der klassische Artbegriff, wie er für Pflanzen oder Tiere verwendet wird, nicht ohne Weiteres auf die DNA Sequenzdaten von Mikroorganismen anwendbar ist. Üblicherweise entsprechen OTUs weitgehend den Arten und sind eine adäquate Möglichkeit mikrobielle Diversität darzustellen.

Die rohen Sequenzierungsdateien müssen vor der weiteren Analyse bearbeitet werden. Erste Schritte wurden dafür vom Sequenzierungsdienstleister (Microsynth GMbh) verarbeitet, die auch das Demultiplexing und Adaptor-Trimmen mit cutadapt v1.14 (Martin 2011) durchgeführt hat, um auch die in den PCR-Primern vorhandenen Ambiguitäten zu entfernen.

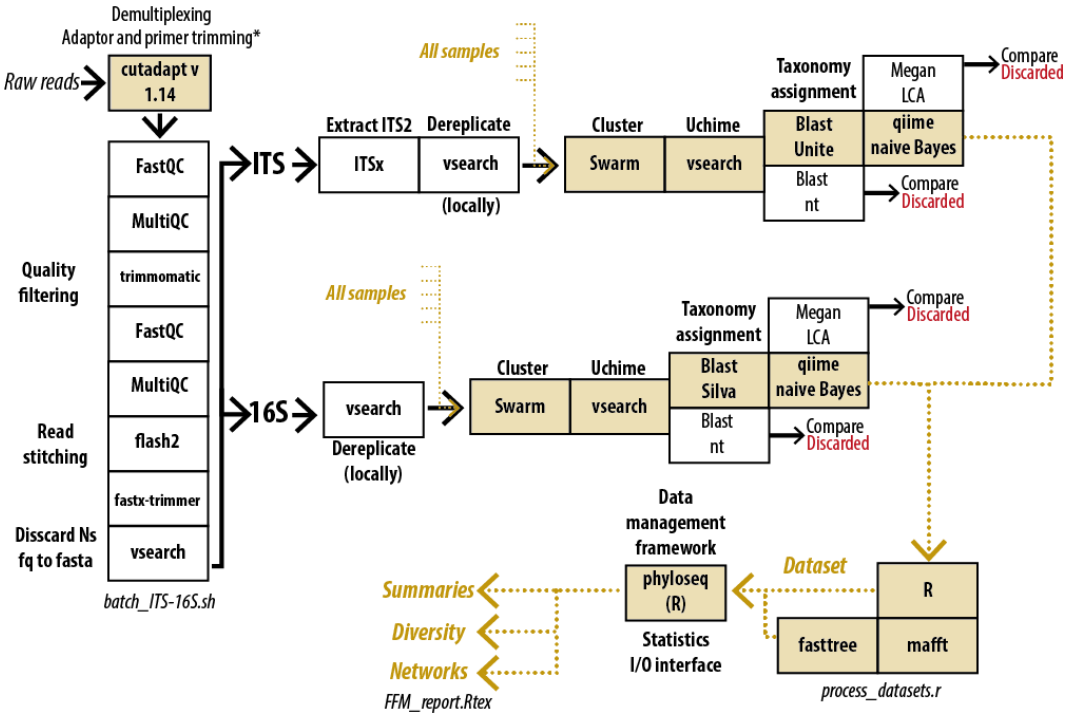

Abbildung 1. Schematische Darstellung der analytischen Pipeline. Die Skript Dateien werden vom Nationalparktservice angeboten. Der Hauptpfad, in dem alle Proben zusammen verarbeitet werden, ist farblich hervorgehoben. Namen in Kursivschrift markieren die wichtigsten resultierenden Datenstrukturen. Bibliotheksvorbereitung, Sequenzierung, Demultiplexen und Trimmen wurden von Microsynth GMbh durchgeführt.

Die sich aus diesen bioinformatischen Reinigungsschritten ergebenden Quality-fasta-Dateien wurden später mit der in Abbildung **1** zusammengefassten Pipeline verarbeitet (im Ergänzungsmaterial erläutert). Die demultiplexten Sequenzdateien wurden mit dem Programm FastQC v0.11.8 untersucht (Andrew 2010). Das Programm MultiQC v1.7 (Ewels et al. 2016) wurde verwendet, um eine breitere kontextuelle Sicht auf mögliche Sequenzierungsfehler und deren Auswirkungen auf die Qualität des Datensatzes zu erhalten. Das Programm Trimmomatic v0.36 wurde zum Reinigen von Adaptersequenzen und zum Trimmen der Qualität der gepaarten Endbibliotheken (Bolger, Lohse & Usadel 2014) verwendet. Später wurden zusammengehörige Sequenz-Paare mit flash v1.2.11 (Magoč and Salzberg 2011) zusammengefügt (*stitched*). Nach einer zweiten Qualitätsuntersuchung mit fastqc v0.11.8 und multiqc v1.7 wurde fastx-trimmer v0.0.14 (Hannon-lab 2018) zur weiteren Reinigung der zusammengesetzten Datensätze verwendet. Weiters wurde Vsearch v11.9 (Rognes et al. 2016) verwendet, um zweifelhafte Nukleotidzuordnungen (Ns) zu verwerfen, in fasta umzuwandeln und schließlich auf Lokalitätsebene zu demultiplizieren.

Bis zu diesem Zeitpunkt wurden ITS- und 16S-Amplicons auf ähnliche Weise verarbeitet, jedoch wurden ITS-Amplicons vor ihrer Dereplikation in Vsearch mit ITSx v1.1.1 (Bengtsson-Palme et al. 2013) verarbeitet, um Sequenzen auszuschließen die nicht zu Pilzen zuordenbar sind, bzw. um die Sequenzanteile der 5.8 S- und LSU-rRNA Gene auszuschließen, sowie spezifische ITS2-Spacerregionen.

Nach der Dereplikation wurden alle Beispieldateien zusammengefasst, nach Größe (Anzahl der Replikate) in Vsearch sortiert und mit einem in Swarm implementierten rekursiven Algorithmus gruppiert (Mahé et al. 2014). Nach dem Clustering der Sequnezen wurde Vsearch erneut verwendet, um eine Chimärendetektion der pro OTU geschätzten repräsentativen Sequenzen unter

Verwendung des UChime-Algorythm durchzuführen (Edgar et al. 2011; Edgar 2016). Parallel dazu wurden alle Amplicons mit Blastn v2.2.30 (Zhang et al. 2000) gegen eine lokale Kopie der Silva 16S (Pruesse et al. 2007; Quast, Pruesse, et al. 2013) und UNITE Pilz ITS (Abarenkov et al. 2010; Nilsson et al. 2019, 2013) Datenbanken geblastet. In einer ersten Instanz wurden Ausgabedateien mit Megan v.5.10.2 (Huson, Mitra, and Ruscheweyh 2011) stapelweise verarbeitet, um rohe taxonomische Profile basierend auf den am wenigsten verbreiteten Vorfahrenkriterien zu generieren. Bei der letzten Iteration wurde jedoch der in Qiime2 v2019.4.0 (Caporaso et al. 2010) enthaltene Sci-Kit-Klassifikator (Bokulich et al. 2018) verwendet.

b. Datensatz-Assemblierung und Filterung

Die Dateien wurden in R (R Development Core Team 2018) importiert, wo sie tabellarisch erfasst und gefiltert wurden. Wir sind dabei von einer Tabellendatei (.csv) ausgegangen, die die Beispielcodes sowie die Transekt- und Lokalitätskennungen enthält, zu denen die Sequenzierungsergebnisse nacheinander hinzugefügt wurden. In einem zweiten Schritt lesen wir die dereplizierten sequentiellen Sequenzdateien für jede Probe (in Abbildung 1 als All_samples benannt) mithilfe von read.dna-Funktion des Paketes ape v5.3 (Paradis, Claude & Strimmer 2004) in R ein. Die Sequenznamen pro Lokalitätsdatei wurden extrahiert, tabelliert und analysiert. Als Nächstes wurden die Swarm-Ausgabedateien analysiert, um den genetischen Ähnlichkeitscluster (den OTUs entsprechend) zu erhalten, dem jede dereplizierte Sequenz zugeordnet werden kann. Aufgrund der großen Anzahl von OTUs, die die sample_wise-Dateien mit den Swarmergebnissen füllten, war dies ein sehr langwieriger und rechenintensiver Prozess. Dann wurden die Resultate der Prüfung auf Chimären und schließlich die taxonomischen Zuordnungen aufgenommen und tabellarisch dargestellt, nachdem sie von Quiime in die TSV-Datei exportiert wurden. Nach dem Zusammenführen aller Datendateien in einem gemeinsamen Datenrahmen pro Locus wurden OTUs, für die einer chimären Ursprung errechnet wurde, oder zweifelhaft eingeschätzte OTUs aus dem Datensatz herausgefiltert (z.B. OTUs, die im 16S-Datensatz als nicht-bakteriell oder nicht-archeal oder im ITS-Datensatz als nicht-Pilz identifiziert wurden). Darüber hinaus wurden auch alle OTUs, die in weniger als 50 Lesevorgänge detektiert wurden, von weiteren Analysen ausgeschlossen.

Schließlich wurden gefilterte OTU- und Taxonomietabellen in phyloseq-S4-Objekte (McMurdie und Holmes 2013) umgewandelt, um sie weiter zu analysieren. Um einen phylogenetischen Baum als Hintergrunddatengrundlage zu erhalten, haben wir die repräsentativen 16S rRNA Gen- Sequenzen mit MAFFT v7 (S. Katoh 2013; K. Katoh et al. 2002) aligned und einen Maximum lieklihood Stammbaum mit Fasttree v.2.1 (Price, Dehal und Arkin 2010) berechnet. Die Phyloseq-Objekte wurden verwendet, um den Datensatz auf Proben- und OTU-Ebenen zu visualisieren und zu filtern. Schlecht repräsentierte Stichproben wurden von den Datensätzen aussortiert.

Das ITS Gen des Pilz- Datensatzes wurde mengenmäßig ziemlich heterogen sequenziert. Das weist auf die Schwierigkeit hin, ein vernünftiges äquimolares Pooling von Proben zu erhalten, wenn DNA-Fragmente extreme Größenunterschiede aufweisen, wie es für die Pilzdaten charakteristisch ist. Die meisten Proben wurden jedoch mit hoher Repräsentation sequenziert, mit Ausnahme von sechs unterrepräsentierten Proben (in Abbildung 2 orange markiert), die von der Analyse ausgeschlossen wurden: vier Proben von UN und zwei von IG; eine vom Transektorkopf (K), zwei vom mittleren Abschnitt (M) und drei von der Basis (T).

Die Einzigartigkeit der OTUs wurde auch zu Filterzwecken berücksichtigt. Der ITS-Datensatz zeigt eine starke bimodale Verteilung, in der einige seltene OTUs durchwegs von untergeordneter Bedeutung und für nur wenige Stichproben spezifisch sind, während die meisten anderen OTUs in einer mehr oder weniger kontinuierlichen Verteilung der Spezifität liegen. Die meisten OTUs sind ähnlich häufig sequenziert und ihre Darstellung in der Datenmenge korreliert mit der Anzahl der gefundenen Proben annäherungsweise einer logarithmischen Beziehung. Die große Mehrzahl der Pilz-OTUs ist nur in wenigen Proben gemeinsam vertreten (Medianwert ca. 3 Proben), während im gesamten Datensatz dennoch eine beträchtliche Anzahl von OTUs vorkommt. Um unterrepräsentierte OTUs auszuschließen, haben wir OTUs herausgefiltert, die nur in einer einzigen Stichprobe erhoben werden konnten wurden und die dort in weniger als 300 Lesevorgänge vorkamen. Dies beseitigte die Störung durch allfällige Verunreinigungen und nicht informativer OTUs der unteren modalen Elemente der in Abbildung 4 gezeigten Dichteverteilung.

Die meisten Proben des Bakteriendatensatzes (16S-rRNA Gene) hatten eine weitgehende homogene Sequenzierintensität, mit Ausnahme von IG4A2A, welche keine brauchbare Menge an Sequenzen lieferte und ausgeschlossen wurde (Abbildung 5). Die Filterung nicht-informativer OTUs wurde auf die gleiche Weise wie für die pilzlichen Daten (ITS) durchgeführt, wobei jedoch der untere Teil der OTU-Verteilung in Abbildung 6, der in einer einzelnen Probe mit weniger als 100 Lesevorgängen gefunden wurde, ausgeschlossen wurde. Die Gesamtgröße des Datensatzes vor und nach dem Filtern ist in Tabelle 2 angegeben.

Tabelle 2. Die Größe des Datensatzes vor und nach der Filterung.

Dataset	filtering	samples	OTUs	Reads
ITS	unfiltered	84	3115	16.937931 millionen
	filtered	78	2813	16.608615 millionen
16S	unfiltered	84	47727	37.753578 millionen
	filtered	83	27056	37.721765 millionen

c. Statistische Analysen

Alle statistischen Analysen und Datenmanipulationen wurden in R in der Version 3.4.2 durchgeführt. Wir haben dabei hauptsächlich das in das Bioconductor-Paket 3.8 (Huber et al., 2015) einbezogenen Pakets phyloseq v1.16.2 (McMurdie und Holmes 2013) verwendet, sowie die Programm-Pakete vegan v2.5-5 (Oksanen et al., 2019) und lme4 v1.1-21 (Bates et al., 2015).

Die sogenannten Phyloseq-Objekte liefern eine umfassende Beschreibung des Datensatzes und lassen sich leicht manipulieren, um taxonomische Muster und statistische Zusammenfassungen auf verschiedene Weise zu untersuchen. Angesichts der Dichte der Daten, die mit Amplicon-Sequenzierungsmethoden gewonnen werden, kann die Interpretation mikrobiologischer Gemeinschaften aus verschiedenen Perspektiven erfolgen. Für die vorliegende Untersuchung haben wir uns für die Interpretation der ökologischen Differenzierung zwischen Proben entschieden, bei denen die Unähnlichkeit in der Zusammensetzung von OTU gemessen wurde. Es sind auch mehrere andere Ansätze möglich und sinnvoll, etwa solche, die Zuordnungen von OTUs zu tatsächlichen taxonomischen Einheiten (Arten) verwenden, sowie solche, die auf der Interpretation der phylogenetischen Unähnlichkeit anhand von UniFrac-Distanzen basieren (Lozupone und Knight 2005; Lozupone et al., 2010; Chang et al. 2011).

Die ökologische Ähnlichkeit mikrobiotischer Gemeinschaften, die oft als Beta-Diversität in Mikrobiomstudien angesprochen wird, wurde unter Verwendung von Ordinationsmethoden für Distanzmatrizen der Zusammensetzung (Faith, Minchin und Belbin 1987) unter Verwendung des von Phyloseq bereitgestellten Rahmens untersucht. Zur Untersuchtung des Datensatzes haben wir mehrere alternative Methoden verwendet, wobei sowohl normalisierte als auch rohe Lesedatensätze verwendet wurden. Als Zielmethode bevorzugten wir die Verwendung des semimetrischen Bray-Curtis-Index (Legendre und Legendre 2012) als Abstandsmaß und nichtmetrische mehrdimensionale Skalierung (NMDS; Kruskal 1964b, 1964a) als Ordinationsmethode. Trotz der Ähnlichkeit der Sequenzierungsintensität haben wir normalisierte Zählwerte für die Ordination verwendet, da kleine numerische Unterschiede dazu führen, dass die Divergenzschätzungen allein wegen der Sequenzierungsintensität erhöht werden können.

Die Bray-Curtis-Entfernungsmetrik ist eine übliche Wahl bei Metabarcoding-Untersuchungen. Sie ist rechnerisch weniger komplex als modernere Metriken (Cao, Williams und Bark 1997), wird weniger durch das Fehlen gemeinsamer OTUs beeinflusst und kann verschiedene weitere Variationsquellen gut erfassen (Cao, Williams und Bark 1997). In Bezug auf die Ordination bietet NMDS eine robuste uneingeschränkte Ordination (Minchin 1987) analog zu MDS (multidimensionale Skalierung, PcoA). Die iterative Implementierung des NMDS in der Funktion metaMDS des Pakets vegan hat den Vorteil, dass lokale Optima der Ordination vermieden werden kann, weil eine Mittelung über mehrere Zufallsstarts des Ordinierungsverfahrens erfolgt.

Während die Pilz-ITS Sequenzen auch bei geringen taxonomischen Entfernungen der OTUs stark divergieren kann, behält der für Prokaryoten verwendete 16S-rRNA Barcode eine signifikante Homologie über die gesamte Bandbreite der bakteriellen Vielfalt. Dies begünstigt den Einsatz der gewählten Entfernungsmetrik und bezieht die phylogenetische Beziehung zwischen den OTUs stärker mit ein. Daher bevorzugten wir gewichtete, normalisierte UniFrac-Unähnlichkeitswerte (Lozupone et al., 2010; Chang, Luan und Sun 2011; Lozupone und Knight 2005) anstelle von DPCoA als eine mögliche Alternative. Während NMDS für Ordination mittels der UniFrac Distanzen verwendet werden kann, aber wegen Einschränkungen in den Phyloseq-Implementierungen und mangelnder Konvergenz bei Verwendung von nur zwei Achsen begrenzt einsetzbar ist, verwenden wir als Alternative auch ein Ordination, die auf parametrischem PcoA (MDS, multidimensionale Skalierung) basiert.

Die Alpha-Diversität pro Probe wurde mit dem phyloseq-Wrapper plot_richness erhalten, der eingesetzt wurde, um eine Rohschätzung der beobachteten OTU-Reichhaltigkeit zu berechnen, sowie zwei verschiedene Schätzwerte für die rarefizierte Reichhaltigkeit (Chao1 und ACE) und drei alternative Diversitätsbewertungen zu berechnen, die nicht nur Reichhaltigkeit, sondern auch Gleichförmigkeit der OTU-Verteilung (Shannon, Simpson und InvSimpson) berücksichtigt. Obwohl die Normalisierung der Daten einen relativ geringen Einfluss auf die Ordinationsergebnisse hat (sie wird die durch die Verwendung der gewählten Entfernungsmetrik gepuffert), sind die Alphadiversität-Schätzwerte stark von den Unterschieden der Sequenzierungsintensität beeinflusst. Dieser Effekt ist besonders stark im 16S-Datensatz zu beachten, der eine viel größere Anzahl von OTUs enthält (Tabelle 2).

Für Diversitätsabschätzungen haben wir uns gegen die Verwendung von normalisierten Prozentsätzen oder normalisierten Pseudoreads entschieden, die eine unkontrollierte numerische Abweichung einführen. Stattdessen haben wir jedoch die rohen Sequenzierungswerte verwendet. Um durch ungleichmäßige Samplingintensität hervorgerufene Fehler zu überwinden, haben wir

zwei alternative Ansätze gewählt. Für die Beschreibung haben wir bevorzugt Schätzmethoden für die OTU-Vielfalt die Berechnung von Chao1 und ACE Indices vorgezogen, die Vergleiche unabhängig von der Samplingintensität der Probenahme abhängen.

Bei der Modellierung von Unterschieden in der Diversität zwischen den untersuchten Umgebungsgradienten haben wir uns entschieden, die Abtastintensität (Gesamtzahl der Ablesevorgänge pro Stichprobe) als Kovariate im Rahmen von *mixed effect* Modellen (Modellen mit gemischten Effekten) einzuführen. Die Verwendung von Modellen mit gemischten Effekten ermöglichte uns eine Kontrolle der Lokalität als zufälligen Effekt. Lineare Mischeffektmodelle wurden dabei so verwendet, wie sie in der Funktion lmer des R package lme4 v1.1-21 implementiert sind (Bates et al., 2015). Um die Signifikanz des Einflusses des Umgebungsgradienten auf die Ergebnisse zu testen, haben wir einen Anova-Test verwendet. Hierbei werden Null-Modelle, bei denen die Abtastintensität und die Stichprobenlokalitäten als zufällige Effekte berücksichtigt wurden, mit den vollständigen Modellen verglichen. Vor dem Ausführen der linearen Modelle verwendeten wir den Shapiro-Wilk-Test (Shapiro & Wilk 1965), um die Normalität der Verteilung der Reichhaltigkeits- und Diversitätswerte in den Proben zu testen, den Bartlett-Test zur Kontrolle der Homogenität der Varianz (Bartlett 1937) und Wilcoxon's Rangsummentest (Wilcoxon 1945), um einen Testwert zu erhalten für erste nicht gerichtete Schätzung der Bruttodifferenz der Mittelwerte zwischen den verschiedenen Positionen im Gradienten.

Um die unterschiedliche Häufigkeit von OTUs und verschiedener taxonomischen Gruppen zwischen den verschiedenen Abschnitten des Gradienten zu untersuchen, verwendeten wir das in Bioconductor Package Edge v3.9 (Robinson, McCarthy und Smyth 2010) eingeführte methodische Rahmenkonzept, das auf negativen Binomialmodellen basiert um auf Muster von differentiellen Unterschieden zwischen den Proben zu stoßen. Diese Methoden wurden als Teil des Bioconductor-Pakets PathoStat v1.10.0 weiterentwickelt, jedoch haben wir eine modifizierte Version der zusätzlichen Skripts von phyloseq verwendet, die unter https://joey711.github.io/phyloseq-extensions/edgeR.html verfügbar ist.

Ausgehend von Phyloseq-Objekten erhielten wir normalisierte OTU-Matrizen mit der Funktion transform_sample_counts. Diese wurden mithilfe der Funktion phyloseq_to_edgeR konvertiert, in der calcNormFactors zur Normalisierung von Faktoren mithilfe der Relative Log Expression (RLE) -Methode verwendet wurde (Anders & Huber 2010). Die enthaltenen OTUs wurden verwendet um eine Ad-hoc-Varianz des Schwellenwerts zu ermitteln, der auf der Basis der Schätzung der Kerndichte für die Repräsentationsabweichung zwischen den Stichproben ausgewählt wurde. Die übliche tendenzielle Streuung (McCarthy, Chen und Smyth 2012) der OTUs im gesamten Datensatz wurde unter Verwendung eines negativen binomialen Wahrscheinlichkeitsrahmens geschätzt, wie er in der Funktionsschätzung Disp implementiert ist. Die Abundanzenunterschiede zwischen den oberen und unteren Anteilen des Gradienten wurden mit der Funktion exactTest untersucht. Die genauen Testergebnisse wurden mit der Funktion topTags visualisiert. Die vorgestellten differenziellen Abundanzmethoden gestalten sich methodisch relativ einfach. Sie könnten in der langfristigen Weiterführung des Projektes erweitert und kompliziert werden, um etwa mit linearen Modelle eine Zeitreihenanalyse durchzuführen, die für die Langzeitüberwachung relevant wäre.

d. Datenspeicherung und Verfügbarkeit

Die Rohdaten, sowie die verarbeiteten Datenmatrizen werden im Nationalparkzentrum auf einer Festplatte gespeichert. Alle für die Datenverarbeitung und statistische Auswertungen verwendeten Skripts werden als elektronische Ergänzung auf der Festplatte mit den Datenmatrizen sowie als Anhang zum Methodenhandbuch zur Verfügung gestellt.

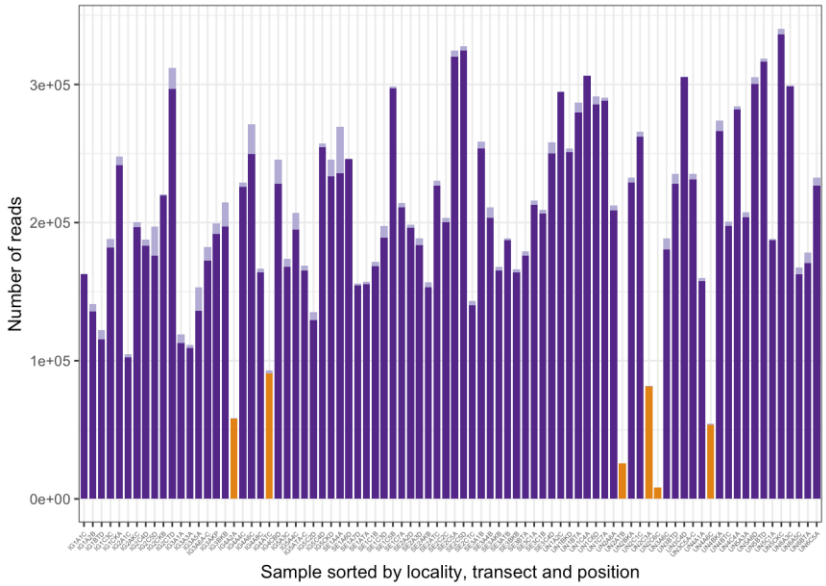

Abbildung 2. Sequenzierungsdichte aller Samples für die Amplicon Sequenzierung der variablen Region der pilzlichen ITS Gene. Die Höhen der Säulen geben die Anzahl der Sequenzen wieder, heller gefärbte Anteile sind chimäre Sequenzen, die von der weiteren Analyse ausgeschlossen wurden. Orange repräsentiert schlecht sequenzierte Samples die ebenfalls von der Analyse ausgeschlossen wurden: Vier Samples von UN und zwei von IG; Eines von der Kopfzeile (K), zwei vom mittleren Bereich (M) und drei von der Tiefzeile (T).

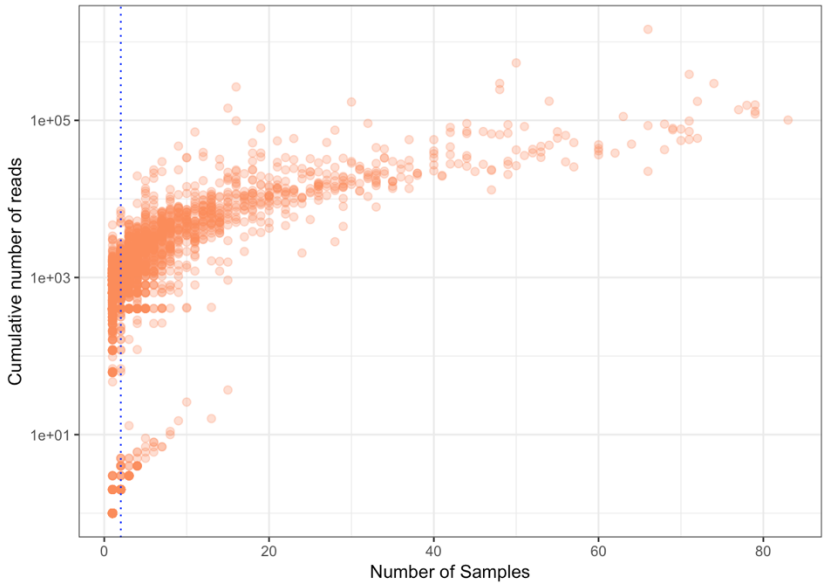

Abbildung 3. Einzigartigkeit der OTUs in der ITS Datenmatrix (chimäre Sequenzen wurden ausgeschlossen). Jeder Punkt entspricht einem OTU, das entsprechend der Anzahl an reads in allen Samples aufgetragen wurde in Relation zur Anzahl der Samples in denen diese OTU vorhanden war. Die blau punktierte Linie repräsentiert den Medianwert der Samples in denen individuelle OTUs gefunden wurden.

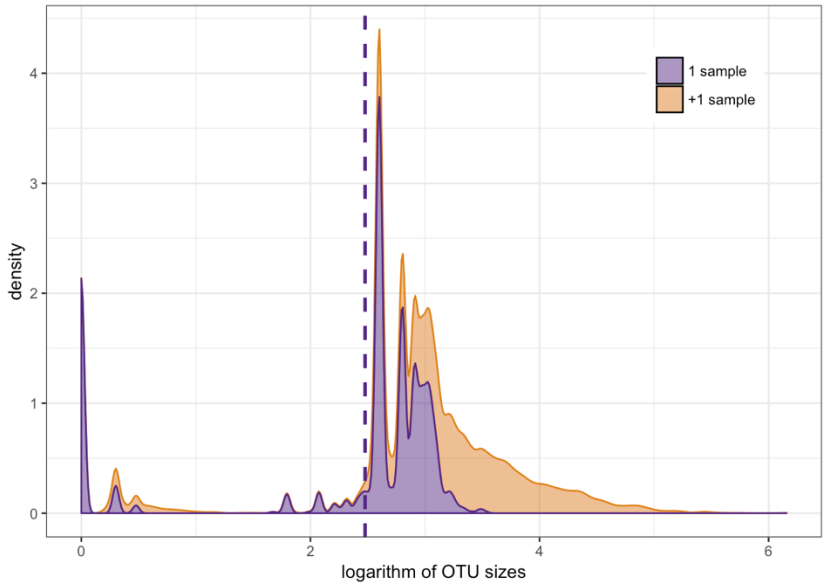

Abbildung 4. Dichteprofile der Sequenzierung des ITS Gen Datensatzes. Die Farbcodes trennen OTUs, die in einzelnen Samples gefunden wurden, von jenen die in mehr als einem Sample vorhanden waren.

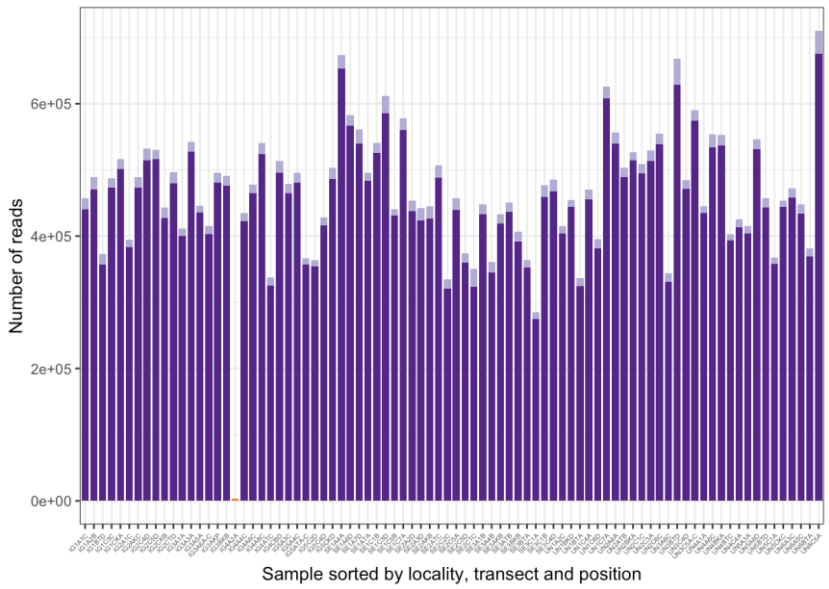

Abbildung 5. Sequenzierungsdichte aller Samples für die Amplicon Sequenzierung der variablen Region bakterieller 16S rRNA Gene. Die Höhen der Säulen geben die Anzahl der Sequenzen wieder, heller gefärbte Anteile sind chimäre Sequenzen, die von der weiteren Analyse ausgeschlossen wurden. Orange repräsentiert schlecht sequenzierte Samples die ebenfalls von der Analyse ausgeschlossen wurden (Ein Sample von IG).

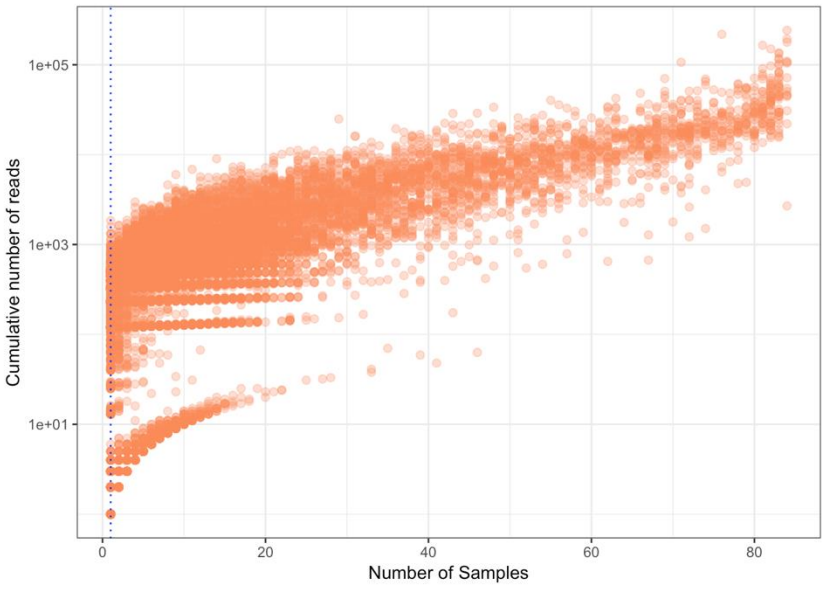

Abbildung 6. Einzigartigkeit der OTUs in der 16S Datenmatrix (chimäre Sequenzen wurden ausgeschlossen). Jeder Punkt entspricht einem OTU, das entsprechend der Anzahl an reads in allen Samples aufgetragen wurde in Relation zur Anzahl der Samples in denen diese OTU vorhanden war. Die blau punktierte Linie repräsentiert den Medianwert der Samples in denen individuelle OTUs gefunden wurden.

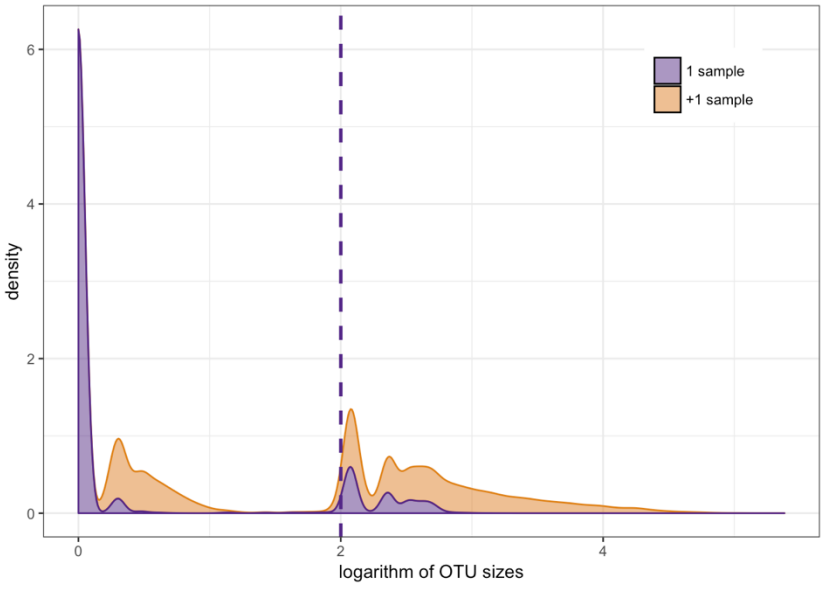

Abbildung 7. Dichteprofile der Sequenzierung des 16S rRNA Gen Datensatzes. Die Farbcodes trennen OTUs, die in einzelnen Samples gefunden wurden, von jenen die in mehr als einem Samples vorhanden waren.

Qualitätssicherung

a. Feldarbeit

Bei den Feldarbeiten wurde entsprechend der best practice darauf geachtet, dass bei der Probennahme keine Kontaminationen aus umgebenden Bodenanteilen eingeschleppt werden, Entnahmewerkzeuge wurden zwischen den Probenahmen sorgfältig gereinigt und die Proben wurden aureichend gekühlt bzw. eingefroren, um mikrobielle Veränderungen nach der Probenahme auszuschließen.

b. Laborarbeit

Die Laborarbeiten wurden routinemäßig nach best practice, wie in der Molekularbiologie üblich, und mit den nötigen Kontrollen durchgeführt.

c. Datenverarbeitung

Die Qualitätssicherung in der Datenanalyse ist bereits implizit in der Darstellung der Analyseschritte für dieses Modul geschildert worden und wird daher hier nicht mehr im Detail wiederholt. Wichtige Schritte sind hier die Ausfilterung von Sequenzen unzureichender Qualität, chimären Sequenzen, sowie unzureichend sequenzierter Proben.

Interpretation der wichtigsten Erhebungsparameter

Zum gegenwärtigen Zeitpunkt lassen sich noch keine Langzeit-Aussagen mit den in der Pilotphase einmalig erhobenen Daten anstellen. Erst die wiederholte Beprobung wird zeigen, in wie weit sich im Laufe des Langzeitmonitorings Verschiebungen in der Bodengemeinschaft einstellen werden. Die durch die Pilotphase gewonnenen Daten bilden die Datengrundlage für künftige Vergleiche. Wir schlagen vor die Beprobung alle 4 Jahre und dann wieder konzertiert mit der Probenahme durch die anderen Module des Langzeitmonitorings durchzuführen. Was sich aber anhand der vorliegenden Erhebung der mikrobiellen Diversitätmuster anhand der erstellten OTU-Tabellen sagen lässt, ist, das es deutliche Unterschiede zwischen den Standorten als innerhalb der Gradienten gibt. Dies ist für die Pilze deutlicher als für die Bakteriengemeinschaften festgestellt worden, letztere scheinen hingegen in ihrem OTU-Reichtum gegen die pessimalen Bereiche der Gradienten abzunehmen. Weitere Interpretationen sind dem Endbericht zu entnehmen.

Abbildungsverzeichnis

Literatur- und Quellenverzeichnis

Abarenkov K, Nilsson RH, Larsson KH, Alexander IJ, Eberhardt U, Erland S, Høiland K, et al. (2010) The UNITE database for molecular identification of fungi -- Recent updates and future perspectives. *New Phytologist* 186: 281–85.

Anders S, Huber W (2010) Differential expression analysis for sequence count data. *Genome Biology* 11:R106

Andrew S (2010) FastQC: A quality control tool for high throughput sequence data. https://github.com/s-andrews/FastQC

Bartlett M (1937) Properties of sufficiency and statistical tests. *Proceedings of the Royal Statistical Society Series* A 160: 268–282.

Bates D, Mächler M, Bolker B, Walker S (2015) Fitting linear mixed-effects models using Lme4. *Journal of Statistical Software* 67: 1–48.

Bengtsson-Palme J, Veldre V, Ryberg M, Hartmann M, Branco S, Wang Z, Godhe A, et al. (2013) ITSx: Improved software detection and extraction of ITS1 and ITS2 from ribosomal ITS sequences of fungi and other Eukaryotes for use in environmental sequencing. *Methods in Ecology and Evolution* 4: 914–19.

Bokulich NA, Kaehler BD, Rideout JR, Dillon M, Bolyen E, Knight R, Huttley GA, Caporaso JG (2018) Optimizing taxonomic classification of marker-gene amplicon sequences with QIIME 2's Q2-Feature-Classifier plugin. *Microbiome* 6: 1–17.

Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics* 30: 2114–2120.

Cao Y, Williams WP, Bark AW (1997) Similarity measure bias in river benthic Aufwuchs community analysis. *Water Environment Research* 69: 95–106.

Caporaso JG, Kuczynski J, Stombaugh J, Bittinger K, Bushman FD, Costello EK, Fierer N, et al. (2010) QIIME allows analysis of high-throughput community sequencing data. *Nature Methods* 7: 335–336.

Chang Q, Luan Y, Sun F (2011) Variance Adjusted Weighted UniFrac : A powerful beta diversity measure for comparing communities based on phylogeny. *BMC Bioinformatics* 11: 118.

Edgar R (2016) UCHIME2: Improved chimera prediction for amplicon sequencing. *BioRxiv*, 074252. https://doi.org/10.1101/074252.

Edgar RC, Haas BJ, Clemente JC, Quince C, Knight R (2011) UCHIME improves sensitivity and speed of chimera detection. *Bioinformatics* 27: 2194–2200.

Ewels P, Magnusson M, Lundin S, Käller M (2016) MultiQC: Summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* 32: 3047–3048.

Faith DP, Minchin PR, Belbin L (1987) Compositional dissimilarity as a robust measure of ecological distance. *Vegetatio* 69: 57–68.

Hannon-lab (2018) FASTX Toolkit. *Http://Hannonlab.Cshl.Edu/Fastx_toolkit/Index.Html*. http://hannonlab.cshl.edu/fastx_toolkit/.

Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC et al. (2015) Orchestrating high-throughput genomic analysis with Bioconductor. *Nature Methods* 12: 115–121.

Huson Dh, Mitra S, Ruscheweyh HJ. (2011) Integrative analysis of environmental sequences using MEGAN4. *Genome Research* 21: 1552–1560.

Katoh K, Misawa K, Kuma K, Miyata T (2002) MAFFT: A novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Research* 30: 3059–3066.

Katoh S (2013) MAFFT Multiple Sequence Alignment Software Version 7: Improvements in performance and usability. *Molecular Biology and Evolution* 30: 772–780.

Körner C (2019) Langzeitmonitoring terrestrischer alpiner Ökosysteme im Nationalpark Hohe Tauern - Konzept und Rahmenbedingungen. Wissenswert, Methoden – Handbuch.

Kruskal JB (1964a) Multidimensional scaling by optimizing Goodness-of-Fit to a nonmetric hypothesis. *Psychometrika* 29: 1–28.

———. 1964b. Nonmetric multidimensional scaling: A numerical method. *Psychometrika* 29: 115–29.

Legendre P, Legendre L (2012) *Numerical Ecology*. 3rd Ed. Elsevier.

Lozupone C, Knight R (2005) UniFrac: A new phylogenetic method for comparing microbial communities 71: 8228–8235.

Lozupone C, Lladser ME, Knights D, Stombaugh J, Knight R (2010) UniFrac: An effective distance metric for microbial community comparison. *The ISME Journal* 5: 169–172.

Magoč T, Salzberg SL (2011) FLASH: Fast length adjustment of short reads to improve genome assemblies. *Bioinformatics* 27: 2957–2963.

Mahé F, Rognes T, Quince C, de Vargas C, Dunthorn M (2014) Swarm: Robust and fast clustering method for amplicon-based studies. *PeerJ* 2: e593.

Martin M (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.Journal* 17: 10–12.

McCarthy DJ, Chen Y, Smyth GK (2012) Differential expression analysis of multifactor RNA-seq experiments with respect to biological variation. *Nucleic Acids Research* 40: 4288–4297.

McMurdie PJ, Holmes S (2013) Phyloseq: An R package for reproducible interactive analysis and graphics of microbiome census data. *PLoS ONE*. 8: e61217.

——— (2014) Waste Not, Want Not: Why rarefying microbiome data is inadmissible. *PLoS Computational Biology* 10: e1003531

Minchin PR (1987) An evaluation of the relative robustness of techniques for ecological ordination. *Vegetatio* 69: 89–107.

Newesely C, Tappeiner U, Körner C (2019) Langzeitmonitoring von Ökosystemprozessen im Nationalpark Hohe Tauern. Modul 01: Standortklima, Bodenphysik, Bodenchemie und pflanzliche Produktivität. Methoden-Handbuch. Verlag der Österreichischen Akademie der Wissenschaften, Wien. ISBN-Online: 978-3-7001-8749-3, doi: 10.1553/GCP_LZM_NPHT_Modul01

Nilsson RH, Larsson K-H, Taylor AFS, Bengtsson-Palme J, Jeppesen TS, Schigel D, Kennedy P, et al. (2019) The UNITE database for molecular identification of fungi: Handling dark taxa and parallel taxonomic classifications. *Nucleic Acids Research* 47 (D1): D259–64.

Nilsson RH, Taylor AFS, Bates ST, Thomas D, Bengtsson-Palme J, Callaghan TM, Douglas B, et al. (2013) Towards a unified paradigm for sequence-based identification of fungi. *Molecular Ecology* 22: 5271–5277.

Oksanen J, Blanchet FG, Friendly M, Kindt R, Legendre P, McGlinn D, Minchin PR, et al. (2019) Package, Vegan: Community Ecology. https://cran.r-project.org/package=vegan.

Paradis E, Claude J, Strimmer K (2004) APE: Analyses of phylogenetics and evolution in R language. *Bioinformatics* 20: 289–90.

Price MN, Dehal PS, Arkin AP (2010) FastTree 2 - Approximately maximum-likelihood trees for large alignments. *PLoS ONE* 5(3): e9490.

Pruesse E, Quast C, Knittel K, Fuchs BM, Glo FO, Ludwig W (2007) SILVA: A comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB. *Nucleic Acids Research* 35 (21): 7188–96.

Quast C , Klindworth A, Pruesse E, Schweer T, Horn M, Oliver Glo FO (2013) Evaluation of general 16S ribosomal RNA gene PCR primers for classical and next-generation sequencing-based diversity studies. *Nucleic Acids Research* 41: 1–11.

Quast C, Pruesse E, Yilmaz P, Gerken J, Schweer T, Glo FO, Yarza P (2013) The SILVA ribosomal RNA gene database project: improved data processing and web-based tools. *Nucleic Acids Research* 41: 590–96.

R Development Core Team, R Foundation for Statistical Computing (2018) R: A language and environment for statistical computing. Vienna, Austria.

Robinson MD, McCarthy DJ, Smyth GK (2010) EdgeR: A Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* 26: 139–40.

Rognes T, Flouri T, Nichols B, Quince C, Mahé F (2016) VSEARCH: A versatile open source tool for metagenomics. *PeerJ*. 18,4:e2584.

Shapiro SS, Wilk MB (1965) An analysis of variance test for normality (complete samples). *Biometrika* 52: 591-611.

White TJ, ThoBruns TD, Lee S, Taylor JW (1990) Amplification and direct sequencing of fungal ribosomal RNA genes for hylogenetics. In *PCR Protocols: A Guide to Methods and Applications*, edited by M.A. Innis, D.H. Gelfand, J.J. Sninsky, and T.J. White, 315–322. San Diego: Academic Press Inc.

Wilcoxon F (1945) Individual comparisons by ranking methods. *Biometrics Bulletin* 1: 80–83

Zhang Z, Schwartz S, Wagner L, Miller W (2000) A greedy algorithm for aligning DNA sequences. *Journal of Computational Biology* 7: 203–14.

Anhang

a. Vorlagen digitale Datenverarbeitung. Pipeline 16S

Die folgende Pipeline ist für die Analye der bakteriellen 16S rRNA Gene vorgesehen.

```
#!/bin/bash/
# ------------------------------
# Pipeline Metabarcoding 16S
# HTNP pilot project
#
# FFM 2018
#------------------------------
#
#
working_dir="/home/fernando/Documents/8_Martin"
raw_data_16S="/home/fernando/Documents/8_Martin/M577/Reads_16S/Trimmed_reads/"
raw_data_ITS="/home/fernando/Documents/8_Martin/M577/Reads_16S/Trimmed_reads/"
#
#source activate qiime2-2018.2
#
# 1. FastQC to allow manual inspection of the quality of sequences
#-------------------------------------------------------------------
#
cd $working_dir
#[ -d 01_fastqc ] || mkdir 01_fastqc/16S && mkdir 01_fastqc/ITS
#
#!#fastqc -t 4 $raw data 16S/* -o ./01 fastqc/16S
#!#fastqc -t 4 $raw_data_ITS/* -o ./01_fastqc/ITS
#
# 2. Process with multiqc
#------------------------------
#
#!#multiqc ./16S/ -o ./16S/
#!#multiqc ./ITS/ -o ./ITS/
#
#
export PATH=$PATH:/usr/local/etc/microbiome_helper/
#
# 3. Create data structure
#-----------------------------------
#1_CLEAN=00_Cleaned
#2_MERGE=01_Merged
#3_DISC=02_Discarded
#4_QUAL=03_quality
#5_TRIM=
#6_CHIM=
[ -d 01_fastqc ] || mkdir 01_fastqc/16S && mkdir 01_fastqc/ITS
mkdir -p ./02_16S/00_Cleaned
mkdir -p ./03 ITS/00 Cleaned
mkdir -p ./02_16S/01_Merged
mkdir -p ./03_ITS/01_Merged
mkdir -p ./02_16S/02_Discarded
mkdir -p ./03_ITS/02_Discarded
mkdir -p ./02_16S/03_quality
mkdir -p ./03_ITS/03_quality
mkdir -p ./02 16S/04 trimmed
mkdir -p ./03_ITS/04_trimmed
mkdir -p ./02_16S/05_fasta/
mkdir -p ./03_ITS/05_fasta/
mkdir -p ./02_16S/06_derep/
mkdir -p ./03_ITS/06_derep/

#-------------------------
# X. Adapter trimming (nope)
#-------------------------


#for BASENAME in $(ls M577/Reads_16S/Trimmed_reads/ | rev | cut -c 24- | rev | uniq);
#do
```

```
#trimmomatic PE -threads 16 -phred33 \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}R1_001_trimmed.fastq.gz \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}R2_001_trimmed.fastq.gz \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}_forward_paired.fq.gz \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}_forward_unpaired.fq.gz \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}_reverse_paired.fq.gz \
#./M577/Reads_16S/Trimmed_reads/${BASENAME}_reverse_unpaired.fq.gz \
#ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3
#TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
#HEADCROP:10 MINLEN:200 AVGQUAL:30
#((i=i%N)); ((i++==0)) && wait
#echo  ./02_16S/01_Merged/${BASENAME}L001_flash2.extendedFrags.fastq
#fastx_trimmer -i ./02_16S/01_Merged/${BASENAME}L001_flash2.extendedFrags.fastq -f 11 -m
340 -Q33| fastx_trimmer -t 10 -Q33| fastq_quality_filter -q 25 -p 95 -Q33 >
./02_16S/04_trimmed/${BASENAME}trimmed.fastq
#&



#
# 4. Stitching reads
#-------------------------------
#
#Stitch paired-end reads together PEAR not available BBMERGE
#cd $working_dir
#mkdir 02_16S
#mkdir 03_ITS
#mkdir 02_16S/01_Merged
#mkdir 03_ITS/01_Merged
#mkdir 02_16S/02_Discarded
#mkdir 03_ITS/02_Discarded

#
#
#N=4
#for BASENAME in {$(ls M577/Reads_16S/Trimmed_reads/ | rev | cut -c 24- | rev | uniq)};
#do
#  ((i=i%N)); ((i++==0)) && wait
#/usr/local/etc/BBMap/bbmerge.sh \
#in1=M577/Reads_16S/Trimmed_reads/${BASENAME}R1_001_trimmed.fastq.gz \
#in2=M577/Reads_16S/Trimmed_reads/${BASENAME}R2_001_trimmed.fastq.gz \
#out=./02_16S/${BASENAME}merged \
#outu1=./02_16S/${BASENAME}R1_unmerged \
#outu2=./02_16S/${BASENAME}R2_unmerged
#done

#Stitch paired-end reads together (summary of stitching results are written to
"pear_summary_log.txt")
#/usr/local/etc/microbiome_helper/run_pear.pl -p 4 -o ./02_16S/01_stitched_reads
$raw_data_16S/*
#/usr/local/etc/microbiome_helper/run_pear.pl -p 4 -o ./03_ITS/01_stitched_reads
$raw_data_ITS/*
#
#
# 4.1. STITCHING WAS DONE DONE USING FLASH2
#
#
# 16 S
#
#N=4
for BASENAME in $(ls M577/Reads_16S/Trimmed_reads/ | rev | cut -c 24- | rev | uniq);
do
#  ((i=i%N)); ((i++==0)) && wait
flash2 ./M577/Reads_16S/Trimmed_reads/${BASENAME}R1_001_trimmed.fastq.gz \
./M577/Reads_16S/Trimmed_reads/${BASENAME}R2_001_trimmed.fastq.gz \
-d ./02_16S \
-o ${BASENAME}flash2
done
#
# ITS
#
for BASENAME in $(ls M577/Reads_ITS/Trimmed_reads/ | rev | cut -c 29- | rev | uniq);
do
flash2 ./M577/Reads_ITS/Trimmed_reads/${BASENAME}L001_R1_001_trimmed.fastq.gz \
```

```
./M577/Reads_ITS/Trimmed_reads/${BASENAME}L001_R2_001_trimmed.fastq.gz \
-d ./03_ITS \
-o ${BASENAME}flash2
done
#
# 4.2. Reorder files

for dir in ./02_16S ./03_ITS
do
mv  $dir/*.extendedFrags.fastq $dir/01_Merged/
mv  $dir/*.* $dir/02_Discarded
done
#
# 5. READ FILTERING
#
#
# 5.1 Controlo las cosas
x
fastqc -t 4 ./02_16S/01_Merged/* -o ./02_16S/03_quality/
fastqc -t 4 ./03_ITS/01_Merged/* -o ./03_ITS/03_quality/
#
# 5.2. Process with multiqc
#------------------------------
#
multiqc ./02_16S/03_quality/* -o ./02_16S/03_quality/
multiqc ./03_ITS/03_quality/* -o ./03_ITS/03_quality/

#
#5.3. Read filtering
#-------------------
#Option A: Read filtering without requiring primer sequences to be at the beginning and
end of each re
#read_filter.pl -q 30 -p 90 -l 400 --primer_check none --thread 4 02_16S/01_Merged
#
#
#
for BASENAME in $(ls ./02_16S/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
fastx_trimmer -i ./02_16S/01_Merged/${BASENAME}L001_flash2.extendedFrags.fastq -f 11 -m
340 -Q33| fastx_trimmer -t 10 -Q33| fastq_quality_filter -q 25 -p 95 -Q33 >
./02_16S/04_trimmed/${BASENAME}trimmed.fastq &
done
#
# 6. Dereplicate
#
# 6.1. Convert FASTQ stitched files to FASTA AND remove any sequences that have an 'N'
in them.
#
for BASENAME in $(ls ./02_16S/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
# 6.2. Discard sequences containing Ns, add expected error rates
vsearch \
--quiet \
--fastq_filter ./02_16S/04_trimmed/${BASENAME}trimmed.fastq \
--fastq_maxns 0 \
--relabel_keep \
--eeout \
--fastqout ./02_16S/04_trimmed/${BASENAME}trimmed_2.fastq 2>>
./02_16S/04_trimmed/0vsearch.log

# 6.3. Discard sequences containing Ns, convert to fasta
vsearch \
--quiet \
--fastq_filter ./02_16S/04_trimmed/${BASENAME}trimmed_2.fastq \
--fastq_maxns 0 \
--fastaout ./02_16S/05_fasta/${BASENAME}trimmed_2.fasta 2>>
./02_16S/04_trimmed/0vsearch.log

# 6.4. Dereplicate at the study level
vsearch \
--quiet \
--derep_fulllength ./02_16S/05_fasta/${BASENAME}trimmed_2.fasta \
--sizeout \
--fasta_width 0 \
```

```
--relabel_keep \
--output ./02_16S/06_derep/${BASENAME}_derep.fasta 2>> ./02_16S/04_trimmed/0vsearch.log
done
#
#
# 7. Global dereplication, clustering and chimera detection
#
#--------------------------------
#VSEARCH=$(which vsearch)
#SWARM=$(which swarm)
TMP_FASTA=$(mktemp --tmpdir=".")
FINAL_FASTA="./02_16S/complete_16S.fas"
#
# 7.1. Pool sequences
#
cat ./02_16S/06_derep/*.fasta > "${TMP_FASTA}"
#
# 7.2. Dereplicate (vsearch) Created a problem downstream
#
#vsearch --derep_fulllength "${TMP_FASTA}" \
#             --sizein \
#             --sizeout \
#             --fasta_width 0 \
#             --output "${FINAL_FASTA}" > /dev/null

vsearch --sortbysize "${TMP_FASTA}" \
             --output "${FINAL_FASTA}" > /dev/null

rm -f "${TMP_FASTA}"


# 7.3. Clustering
THREADS=16
TMP REPRESENTATIVES=$(mktemp --tmpdir=".")
swarm \
    -d 1 -f -t $THREADS -z \
    -i ${FINAL_FASTA/.fas/_1f.struct} \
    -s ${FINAL_FASTA/.fas/_1f.stats} \
    -w ${TMP_REPRESENTATIVES} \
    -o ${FINAL_FASTA/.fas/_1f.swarms} < ${FINAL_FASTA}

# 7.4. Sort representatives
vsearch --fasta_width 0 \
             --sortbysize ${TMP_REPRESENTATIVES} \
             --output ${FINAL_FASTA/.fas/_1f_representatives.fas}
rm ${TMP_REPRESENTATIVES}

# 7.5. Chimera checking
REPRESENTATIVES=${FINAL_FASTA/.fas/_1f_representatives.fas}
UCHIME=${REPRESENTATIVES/.fas/.uchime}
BLAST=${REPRESENTATIVES/.fas/.uchime}
vsearch --uchime_denovo "${REPRESENTATIVES}" \
             --uchimeout "${UCHIME}"

#
# 8. blast
#
#
#blastn -query "${REPRESENTATIVES}" \
blastn -query
/home/fernando/Documents/8_Martin/02_16S/complete_16S_1f_representatives.fas \
-db nt \
-outfmt 0 \
-num_threads 30 \
-out /home/fernando/Documents/8_Martin/02_16S/complete_16S_1f_blast.txt

blastn -query
/home/fernando/Documents/8_Martin/02_16S/complete_16S_1f_representatives.fas \
-db /home/fernando/Documents/8_Martin/99_silva/ARB_sequin.fasta \
-outfmt 0 \
-num_threads 30 \
-out /home/fernando/Documents/8_Martin/02_16S/complete_16S_Silva.txt

#
```

```
# 9. Qiime to calculate LCA instead of Megan
#
#
source activate qiime2-2018.2
qiime tools import \
  --input-path complete_16S_1f_representatives.fas \
  --output-path complete_16S_1f_representatives.qza \
  --type 'FeatureData[Sequence]'


  qiime feature-classifier classify-sklearn \
    --i-classifier gg-13-8-99-515-806-nb-classifier.qza \
    --i-reads complete_16S_1f_representatives.qza \
    --o-classification taxonomy.qza

  qiime metadata tabulate \
    --m-input-file taxonomy.qza \
    --o-visualization taxonomy.qzv

qiime tools export taxonomy.qza --output-dir 07_taxonomy

#
#
#
#
#
#
#
#In R
#
load("/Users/ferninfm/Desktop/microbiome/dataset_v2.Rdata")
tax_16S<-read.table("/Users/ferninfm/Desktop/microbiome/taxonomy_16S.tsv",sep="\t")

library(ape)
ITS_files<-list()
i=1
for (FILE in list.files(pattern="_ITSx.fasta.ITS2.fasta"))
{
ITS_files[[i]]<-read.dna(FILE,format="fasta")
i<-i+1
}
names(ITS_files)<-strsplit(list.files(pattern="_ITSx.fasta.ITS2.fasta"),"_")
save.image("../ITS_dataset_v1.Rdata")
```

b. Vorlagen digitale Datenverarbeitung. Pipeline ITS

Die folgende Pipeline ist für die Analye der pilzlichen ITS Gene vorgesehen.

```
#!/bin/bash/
# Set working directory
working_dir="/home/fernando/Documents/8_Martin"
# Set location of demultiplexed data files
raw_data_ITS="/home/fernando/Documents/8_Martin/M577/Reads_ITS/Trimmed_reads/"
# Activate Conda environment for QUIIME
source activate qiime2-2018.2
#------------------------------------------------------------------
# 1. FastQC to allow manual inspection of the quality of sequences
#------------------------------------------------------------------
#
cd $working_dir
#[ -d 01_fastqc ] || mkdir 01_fastqc/16S && mkdir 01_fastqc/ITS
#
#!#fastqc -t 4 $raw_data_16S/* -o ./01_fastqc/16S
fastqc -t 4 $raw_data_ITS/* -o ./01_fastqc/ITS
#
# 2. Process with multiqc
#------------------------------
#
#!#multiqc ./16S/ -o ./16S/
multiqc ./ITS/ -o ./ITS/
```

```
#
#
export PATH=$PATH:/usr/local/etc/microbiome_helper/
#
# 3. Create data structure
#-----------------------------------
#1_CLEAN=00_Cleaned
#2_MERGE=01_Merged
#3_DISC=02_Discarded
#4_QUAL=03_quality
#5_TRIM=
#6_CHIM=
[ -d 01_fastqc ] || mkdir 01_fastqc/ITS

mkdir -p ./03_ITS/00_Cleaned
mkdir -p ./03_ITS/01_Merged
mkdir -p ./03_ITS/02_Discarded
mkdir -p ./03_ITS/03_quality
mkdir -p ./03_ITS/04_trimmed
mkdir -p ./03_ITS/05_fasta/
mkdir -p ./03 ITS/06 ITSx
mkdir -p ./03_ITS/07_derep/

#-------------------------
# X. Adapter trimming (nope)
#-------------------------
#
# 4. Stitching reads
#--------------------------------
#
#
# 4.1. STITCHING WAS DONE DONE USING FLASH2
#
#
# ITS
#
for BASENAME in $(ls M577/Reads_ITS/Trimmed_reads/ | rev | cut -c 29- | rev | uniq);
do
flash2 ./M577/Reads_ITS/Trimmed_reads/${BASENAME}L001_R1_001_trimmed.fastq.gz \
./M577/Reads_ITS/Trimmed_reads/${BASENAME}L001_R2_001_trimmed.fastq.gz \
-d ./03_ITS \
-o ${BASENAME}flash2
done
#
# 4.2. Reorder files
#
for dir in ./03_ITS
do
mv  $dir/*.extendedFrags.fastq $dir/01_Merged/
mv  $dir/*.* $dir/02_Discarded
done
#
# 5. READ FILTERING
#
#
# 5.1 Controlo las cosas
fastqc -t 4 ./03_ITS/01_Merged/* -o ./03_ITS/03_quality/
#
# 5.2. Process with multiqc
#-------------------------------
#
multiqc ./03_ITS/03_quality/* -o ./03_ITS/03_quality/

#
#5.3. Read filtering
#-------------------
#
for BASENAME in $(ls ./03_ITS/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
fastx_trimmer -i ./03_ITS/01_Merged/${BASENAME}L001_flash2.extendedFrags.fastq -f 11 -m
340 -Q33| fastx_trimmer -t 10 -Q33| fastq_quality_filter -q 25 -p 95 -Q33 >
./03_ITS/04_trimmed/${BASENAME}trimmed.fastq &
done
#
```

```
# 6. Dereplicate
#
# 6.1. Convert FASTQ stitched files to FASTA AND remove any sequences that have an 'N'
in them.
#
for BASENAME in $(ls ./03_ITS/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
# 6.2. Discard sequences containing Ns, add expected error rates
vsearch \
--quiet \
--fastq_filter ./03_ITS/04_trimmed/${BASENAME}trimmed.fastq \
--fastq_maxns 0 \
--relabel_keep \
--eeout \
--fastqout ./03_ITS/04_trimmed/${BASENAME}trimmed_2.fastq 2>>
./03_ITS/04_trimmed/0vsearch.log

# 6.3. Discard sequences containing Ns, convert to fasta
vsearch \
--quiet \
--fastq filter ./03 ITS/04 trimmed/${BASENAME}trimmed 2.fastq \
--fastq_maxns 0 \
--fastaout ./03_ITS/05_fasta/${BASENAME}trimmed_2.fasta 2>>
./03_ITS/04_trimmed/0vsearch.log
done


#

#
# 7. ITS2 extraction
#
for BASENAME in $(ls ./03_ITS/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
ITSx \
-i ./03_ITS/05_fasta/${BASENAME}trimmed_2.fasta \
-o ./03_ITS/06_ITSx/${BASENAME}ITSx.fasta \
-p /usr/local/etc/ITSx_1.1/ITSx_db/HMMs \
--save_regions ITS2 \
-t F \
--cpu 32
done
#
for BASENAME in $(ls ./03_ITS/01_Merged/ | rev | cut -c 32- | rev | uniq)
do
# 6.4. Dereplicate at the study level
vsearch \
--quiet \
--derep_fulllength ./03_ITS/06_ITSx/${BASENAME}ITSx.fasta.ITS2.fasta \
--sizeout \
--fasta_width 0 \
--relabel_keep \
--output ./03_ITS/07_derep/${BASENAME}_ITS2_derep.fasta 2>>
./03_ITS/07_derep/vsearchITS2.log
done
#
#
# 8. Global dereplication, clustering and chimera detection
#
#---------------------------------
#VSEARCH=$(which vsearch)
#SWARM=$(which swarm)
TMP FASTA="./03 ITS/complete ITS2 erase.fas"
FINAL_FASTA="./03_ITS/08_swarm/complete_ITS2.fas"
#
# 8.1. Pool sequences
#
cat ./03_ITS/07_derep/*__ITS2_derep.fasta > "${TMP_FASTA}"
#
# 8.2. Dereplicate (vsearch) # This creates a problem Information on sequences belonging
to each cluster is lost
#
#vsearch --derep_fulllength "${TMP_FASTA}" \
#             --sizein \
```

```
#             --sizeout \
#             --fasta_width 0 \
#             --output "${FINAL_FASTA}" > /dev/null
vsearch --sortbysize "${TMP_FASTA}" \
             --output "${FINAL FASTA}" > /dev/null

rm -f "${TMP_FASTA}"

# 8.3. Clustering
THREADS=30
TMP_REPRESENTATIVES=$(mktemp --tmpdir=".")
swarm \
    -d 1 -f -t $THREADS -z \
    -i ${FINAL_FASTA/.fas/_1f.struct} \
    -s ${FINAL_FASTA/.fas/_1f.stats} \
    -w ${TMP_REPRESENTATIVES} \
    -o ${FINAL_FASTA/.fas/_1f.swarms} < ${FINAL_FASTA}

# 8.4. Sort representatives
vsearch --fasta_width 0 \
             --sortbysize ${TMP REPRESENTATIVES} \
             --output ${FINAL_FASTA/.fas/_1f_representatives.fas}
rm ${TMP_REPRESENTATIVES}

# 8.5. Chimera checking
REPRESENTATIVES=${FINAL_FASTA/.fas/_1f_representatives.fas}
UCHIME=${REPRESENTATIVES/.fas/.uchime}
vsearch --uchime_denovo "${REPRESENTATIVES}" \
             --uchimeout "${UCHIME}"

#
# 9. blast
#
#
#blastn -query "${REPRESENTATIVES}" \
blastn -query ./03_ITS/08_swarm/complete_ITS2_1f_representatives.fas \
-db nt \
-outfmt 0 \
-num_threads 30 \
-out ./03_ITS/complete_ITS2_1f_blast_Unite.txt

#
# 9. Qiime to calculate LCA instead of Megan
#
#

source activate qiime2-2018.6
##
#
#. Import unite
#

#qiime tools import \
# --type FeatureData[Sequence] \
# --input-path sh_refs_qiime_ver7_dynamic_s_01.12.2017.fasta \
# --output-path unite_ver7_dynamic_seqs_01.12.2017.qza
#
#
#
#qiime tools import \
# --type FeatureData[Taxonomy] \
# --input-path sh_taxonomy_qiime_ver7_dynamic_s_01.12.2017.txt \
# --output-path unite ver7 dynamic tax 01.12.2017.qza \
# --source-format HeaderlessTSVTaxonomyFormat
#
#
#
#qiime feature-classifier fit-classifier-naive-bayes \
# --i-reference-reads unite_ver7_dynamic_seqs_01.12.2017.qza \
# --i-reference-taxonomy unite_ver7_dynamic_tax_01.12.2017.qza \
# --o-classifier unite_ver7_dynamic_classifier_01.12.2017.qza

#
# Taxonomy
```

```
#
source activate qiime2-2018.2
qiime tools import \
  --input-path ./03_ITS/08_swarm/complete_ITS2_1f_representatives.fas \
  --output-path ./03 ITS/11 LCA qiime/complete ITS2 1f representatives.qza \
  --type 'FeatureData[Sequence]'


  qiime feature-classifier classify-sklearn \
    --i-classifier
03_ITS/11_LCA_qiime/0_reference/unite_ver7_dynamic_classifier_01.12.2017.qza \
    --i-reads ./03_ITS/11_LCA_qiime/complete_ITS2_1f_representatives.qza \
    --o-classification ./03_ITS/11_LCA_qiime/ITS_taxonomy.qza

  qiime metadata tabulate \
    --m-input-file ./03_ITS/11_LCA_qiime/ITS_taxonomy.qza \
    --o-visualization ./03_ITS/11_LCA_qiime/ITS_taxonomy.qzv

qiime tools export ./03_ITS/11_LCA_qiime/ITS_taxonomy.qza --output-dir
./03_ITS/11_LCA_qiime
```

c. Verarbeitung im Programm R

```
#
# Processing results and importing data structures into phyloseq objects
#
# FFM 8.2018
#
#
#---------------

#
# 0. Reorganize_dataset
#
dataset<-read.table("/Users/ferninfm/Desktop/microbiome/dataset_file",sep="\t")

foo<-as.character(dataset[,4])
foo[dataset[,1]=="FU"&dataset[,4]%in%c("T","1","2")]<-"T"
foo[dataset[,1]=="FU"&dataset[,4]%in%c("3","4","5")]<-"M"
foo[dataset[,1]=="FU"&dataset[,4]%in%c("6","7","K","A")]<-"K"
#
foo[dataset[,1]=="IG"&dataset[,4]%in%c("T","1","2")]<-"T"
foo[dataset[,1]=="IG"&dataset[,4]%in%c("3","4","5")]<-"M"
foo[dataset[,1]=="IG"&dataset[,4]%in%c("6","8","B","K")]<-"K"
#
foo[dataset[,1]=="OB"&dataset[,4]%in%c("1")]<-"T"
foo[dataset[,1]=="OB"&dataset[,4]%in%c("4")]<-"M"
foo[dataset[,1]=="OB"&dataset[,4]%in%c("6","7")]<-"K"
#
foo[dataset[,1]=="UN"&dataset[,4]%in%c("T","1","2")]<-"T"
foo[dataset[,1]=="UN"&dataset[,4]%in%c("3","4","5")]<-"M"
foo[dataset[,1]=="UN"&dataset[,4]%in%c("6","7","K")]<-"K"
foo[dataset[,1]=="UN"&dataset[,2]=="6"&dataset[,4]=="5"]<-"K"
foo[dataset[,1]=="UN"&dataset[,2]=="2"&dataset[,4]=="6"]<-"M"
foo[dataset[,1]=="UN"&dataset[,2]=="1"&dataset[,4]=="6"]<-"M"
#
foo[dataset[,1]=="SE"&dataset[,4]%in%c("T","1","2")]<-"T"
foo[dataset[,1]=="SE"&dataset[,4]%in%c("3","4","5")]<-"M"
foo[dataset[,1]=="SE"&dataset[,4]%in%c("6","7","B","K")]<-"K"
foo[dataset[,1]=="SE"&dataset[,2]=="3"&dataset[,4]=="5"]<-"K"
dataset<-cbind(dataset,CODE=foo)
rownames(dataset)<-sapply(strsplit(rownames(dataset)," "),`[`,2)

#
# 1. Mainframe dataset tables
#
#
dataset.16S<-read.table("/Users/ferninfm/Desktop/microbiome/dataset_file",sep="\t")
```

```
dataset.ITS<-dataset.16S
#
# 2. Samples
#
#
# Read files
#
library(ape)
library(reshape2)
sample.list.16S<-list()
j=1
for (i in (list.files(path = "/Users/ferninfm/Desktop/microbiome/16S/06_derep")))
{
sample.list.16S[[i]]<-
names(read.dna(paste("/Users/ferninfm/Desktop/microbiome/16S/06_derep/",i,sep=""),format
="fasta"))
j<-j+1
}
#
# Generate matrix
#
#####sample.list.16S<-sample.16S
sample.list.16S<-melt(sample.list.16S)
colnames(sample.list.16S)<-c("seq","sample")
sample.list.16S[,2]<-sapply(strsplit(sample.list.16S$sample,"_"),`[`,1)
sample.list.16S<-as.matrix(sample.list.16S)
sample.list.16S<-
cbind(sample.list.16S,size=sapply(strsplit(sample.list.16S[,1],";"),`[`,2))

sample.list.16S[,3]<-substr(sample.list.16S[,3],6,nchar(sample.list.16S[,3]))
#
# Read files
#
sample.list.ITS<-list()
j=1
for (i in (list.files(path = "/Users/ferninfm/Desktop/microbiome/ITS/07_derep")))
{
sample.list.ITS[[i]]<-
names(read.dna(paste("/Users/ferninfm/Desktop/microbiome/ITS/07_derep/",i,sep=""),format
="fasta"))
j<-j+1
}
#
# Generate matrix
#
sample.list.ITS<-as.matrix(melt(sample.list.ITS))
colnames(sample.list.ITS)<-c("seq","sample")
sample.list.ITS[,2]<-substr(sample.list.ITS[,2],1,nchar(sample.list.ITS[,2])-18)
sample.list.ITS<-
cbind(sample.list.ITS,size=sapply(strsplit(as.character(sample.list.ITS[,1]),";"),`[`,2)
)
sample.list.ITS[,3]<-substr(sample.list.ITS[,3],6,nchar(sample.list.ITS[,3]))
#
# 3. Clusters
#
swarms.16S<-readLines("/Users/ferninfm/Desktop/microbiome/16S/complete_16S_1f.swarms")
swarms.ITS<-
readLines("/Users/ferninfm/Desktop/microbiome/ITS/08_swarm/complete_ITS2_1f.swarms")
#
# Add to matrix
#
sample.list.ITS<-cbind(sample.list.ITS,otu=0)
foo2<-sapply(strsplit(sample.list.ITS[,1],"\\|"),`[`,1)
for (i in 1:length(swarms.ITS))
{
foo<-sapply(strsplit(unlist(strsplit(swarms.ITS[i]," ")),"\\|"),`[`,1)
sample.list.ITS[foo2%in%foo,4]<-i
}

#
#
#
sample.list.16S<-cbind(sample.list.16S,otu=0)#
foo3<-sapply(strsplit(sample.list.16S[,1],";"),`[`,1)
```

```
rownames(sample.list.16S)<-foo3
rename_fer<-function(x)
       {
       return(sapply(strsplit(unlist(strsplit(x," ")),";"),`[`,1))
              }

foo_new1<-lapply(swarms.16S,rename_fer)
#foo_new2<-melt(foo_new1) is too slow
library(future)
plan(multiprocess)
names(foo_new1)<-as.character(1:length(foo_new1))
res1 %<-% melt(foo_new1[1:200])
res2 %<-% melt(foo_new1[201:500])
res3 %<-% melt(foo_new1[501:1000])
res4 %<-% melt(foo_new1[1001:2000])
res5 %<-% melt(foo_new1[2001:3000])
res6 %<-% melt(foo_new1[3001:6000])
res7 %<-% melt(foo_new1[6001:8000])
res8 %<-% melt(foo_new1[8001:10000])
res9 %<-% melt(foo_new1[10001:13000])
res10 %<-% melt(foo new1[13001:14000])
res11 %<-% melt(foo_new1[14001:15000])
res12 %<-% melt(foo_new1[15001:20000])
res13 %<-% melt(foo_new1[20001: 40000])
res14 %<-% melt(foo_new1[40001: 50000])
res15 %<-% melt(foo_new1[50001: 60000])
res16 %<-% melt(foo_new1[60001: 70000])
res17 %<-% melt(foo_new1[70001: 80000])
res18 %<-% melt(foo_new1[80001: 90000])
res19 %<-% melt(foo_new1[90001: 100000])
res20 %<-% melt(foo_new1[100001:110000])
res21 %<-% melt(foo_new1[110001:120000])
res22 %<-% melt(foo_new1[120001:130000])
res23 %<-% melt(foo new1[130001:140000])
res24 %<-% melt(foo_new1[140001:150000])
res25 %<-% melt(foo_new1[150001:160000])
res26 %<-% melt(foo_new1[160001:170000])
res27 %<-% melt(foo_new1[170001:180000])
res28 %<-% melt(foo_new1[180001:190000])
res29 %<-% melt(foo_new1[190001:length(foo_new1)])

foo_new2<-rbind(
         res1,
      res2,
      res3,
      res4,
      res5,
      res6,
      res7,
      res8,
      res9,
      res10,
      res11,
      res12,
      res13,
      res14,
      res15,
      res16,
      res17,
      res18,
      res19,
      res20,
      res21,
      res22,
      res23,
      res24,
      res25,
      res26,
      res27,
      res28,
      res29)
         #
         #
         #
```

```
         #



save.image('~/Desktop/microbiome/datasets final5.Rdata')
rownames(foo_new2)<-foo_new2[,1]
foo_new2<-foo_new2[foo3,]
#
# Check
#
table(rownames(sample.list.16S)%in%foo3)
sample.list.16S[,4]<-foo_new2[rownames(sample.list.16S),2]

#foo<-NULL
#for (i in 1:length(swarms.16S))
#{
#foo<-rbind(foo,cbind(sapply(strsplit(unlist(strsplit(swarms.16S[i],"
")),";"),`[`,1),i))
#}
#rownames(foo)<-foo[,1]
#sample.list.16S[,4]<-i
#}
##
##
##
##
##
#library(foreach)
#library(doParallel)
##
###setup parallel backend to use many processors
##cores=detectCores()
#cl <- makeCluster(31) #not to overload your computer
#registerDoParallel(cl)
#culote<-function(x.swarms,j)
#{
#      foo<-sapply(strsplit(unlist(strsplit(x.swarms[j]," ")),";"),`[`,1)
#      x.list[names(x.list)%in%foo]<-j
#      return(x.list[names(x.list)%in%foo])
#}
#finalMatrix <- foreach(i=1:length(swarms.16S), .combine=rbind) %dopar% {
#   tempMatrix = culote(foo4,swarms.16S,i)
#   tempMatrix #Equivalent to finalMatrix = cbind(finalMatrix, tempMatrix)
#}
##stop cluster
#stopCluster(cl)




#
#
#
rownames(sample.list.ITS)<-foo2
rownames(sample.list.16S)<-foo3

rm(foo,foo2,foo3)
#
# 4. Chimeras
#
chimera.16S<-
read.table("/Users/ferninfm/Desktop/microbiome/16S/complete 16S 1f representatives.uchim
e")
chimera.ITS<-
read.table("/Users/ferninfm/Desktop/microbiome/ITS/08_swarm/complete_ITS2_1f_representat
ives.uchime")
#
#
#
rownames(chimera.16S)<-sapply(strsplit(as.matrix(chimera.16S$V2),";"),`[`,1)
rownames(chimera.ITS)<-sapply(strsplit(as.matrix(chimera.ITS$V2),"\\|"),`[`,1)
chimera.16S<-cbind(chimera.16S,V19=0)
chimera.ITS<-cbind(chimera.ITS,V19=0)
```

```
#
# ITS primera parte
#
#---------------
#
#-------------------
fer.subset<-function(x,sept1=" ",sept2="\\|")
{
       sapply(strsplit(unlist(strsplit(x,sept1)),sept2),`[`,1)
}


#####

foo<-sapply(swarms.ITS,fer.subset)
for (i in 1:length(swarms.ITS))
{
foo<-sapply(strsplit(unlist(strsplit(swarms.ITS[i]," ")),"\\|"),`[`,1)
chimera.ITS[rownames(chimera.ITS)%in%foo,19]<-i
}
otus.ITS<-chimera.ITS[,c(19,18)]
#
# 16S primera parte
#
chimera.16S[,19]<-foo_new2[rownames(chimera.16S),2]
otus.16S<-chimera.16S[,c(19,18)]
#--------
# To avoid repeating lengthy computations I did this to repair faulty output
# chimera.ITS<-chimera.ITS[!is.na(chimera.ITS$V1),]
# otus.ITS<-chimera.ITS[,c(19,18)]
#-----------------
#
# 5. Taxonomy
#
tax.ITS<-
read.table("/Users/ferninfm/Desktop/microbiome/ITS/11_LCA_qiime/taxonomy.tsv",sep="\t",h
eader=T)
rownames(tax.ITS)<-sapply(strsplit(as.matrix(tax.ITS$Feature.ID),"\\|"),`[`,1)

#source("https://bioconductor.org/biocLite.R")
#biocLite("phyloseq")


foo<-sapply(as.matrix(tax.ITS$Taxon),parse_taxonomy_qiime)
names(foo)<-rownames(tax.ITS)
tax.ITS<-build_tax_table(foo)
#
#
#
# 6. Build OTU_sample tables
#------------------------------
# ITS dataset
#-------------------------------
# test
table(rownames(tax.ITS)==rownames(otus.ITS))
#
tax.ITS<-cbind(otus.ITS,tax.ITS)
colnames(tax.ITS)<-c("otu","chim",colnames(tax.ITS)[-c(1,2)])
filtered.dataset.ITS<-aggregate(as.numeric(size)~otu+sample,data=sample.list.ITS,sum)
foo<-tax.ITS
rownames(foo)<-paste("otu",foo$otu,sep="_")
foo<-as.matrix(foo)
foo[is.na(foo)]<-"unidentified"
#
filtered.dataset.ITS<-
cbind(filtered.dataset.ITS,foo[paste("otu",filtered.dataset.ITS$otu,sep="_"),])
rownames(dataset.ITS)<-sapply(strsplit(rownames(dataset.ITS)," "),`[`,2)
filtered.dataset.ITS<-
cbind(dataset.ITS[filtered.dataset.ITS$sample,],filtered.dataset.ITS)
unfiltered.dataset.ITS<-filtered.dataset.ITS
filtered.dataset.ITS<-filtered.dataset.ITS[filtered.dataset.ITS$chim=="N",]
rm(chimera.ITS, dataset.ITS, otus.ITS, sample.list.ITS, sample.list.ITS.bkp, swarms.ITS,
tax.ITS)
#------------------------------
```

```
# 16S dataset
#-------------------------------
tax.16S<-
read.table("/Users/ferninfm/Desktop/microbiome/16S/taxonomy.tsv",sep="\t",header=T)
rownames(tax.16S)<-sapply(strsplit(as.matrix(tax.16S$Feature.ID),";"),`[`,1)
library(phyloseq)
foo<-sapply(as.matrix(tax.16S$Taxon),parse_taxonomy_qiime)
names(foo)<-rownames(tax.16S)
tax.16S<-build_tax_table(foo)
# preparatory fixing errors to avoid recomputing
#rownames(chimera.16S)<-sapply(strsplit(rownames(chimera.16S),";"),`[`,1)
#rownames(sample.list.16S)<-sapply(strsplit(rownames(sample.list.16S),";"),`[`,1)
#chimera.16S$V19<-sample.list.16S[rownames(chimera.16S),4] wrong
#otus.16S<-chimera.16S[,c(19,18)]
#
# test
#
table(rownames(tax.16S)==rownames(otus.16S))
#
tax.16S<-cbind(otus.16S,tax.16S)
colnames(tax.16S)<-c("otu","chim",colnames(tax.16S)[-c(1,2)])
filtered.dataset.16S<-aggregate(as.numeric(size)~otu+sample,data=sample.list.16S,sum)
colnames(filtered.dataset.16S)[3]<-"size"
#
foo<-tax.16S
rownames(foo)<-paste("otu",foo$otu,sep="_")
foo<-as.matrix(foo)
foo[is.na(foo)]<-"unidentified"
#
filtered.dataset.16S<-
cbind(filtered.dataset.16S,foo[paste("otu",filtered.dataset.16S$otu,sep="_"),])
filtered.dataset.16S<-cbind(dataset[filtered.dataset.16S$sample,],filtered.dataset.16S)
colnames(filtered.dataset.16S)[9]<-"size"
unfiltered.dataset.16S<-filtered.dataset.16S
filtered.dataset.16S<-filtered.dataset.16S[filtered.dataset.16S$chim=="N",]
rm(chimera.16S, dataset.16S, otus.16S, sample.list.16S, swarms.16S, tax.16S,foo,i,j)
#
#
#
#
filtered.dataset.ITS[,-7]->filtered.dataset.ITS # ERROR it was set at 6
filtered.dataset.16S[,-7]->filtered.dataset.16S
#colnames(filtered.dataset.ITS)[8]<-"size"
#
# 7. Filter by taxonomy
#
filtered.dataset.ITS<-filtered.dataset.ITS[filtered.dataset.ITS$Kingdom=="Fungi",]
filtered.dataset.16S<-
filtered.dataset.16S[filtered.dataset.16S$Kingdom!="unidentified",]
#
# 8. Make OTU tables
#
#table.otus.16S<-xtabs(size~otu+sample,data=filtered.dataset.16S)
#table.otus.ITS<-xtabs(size~otu+sample,data=filtered.dataset.ITS)
#


#
# 10. Clean_out and reorder levels
#
#filtered.dataset.ITS<-droplevels(filtered.dataset.ITS)
#filtered.dataset.16S<-droplevels(filtered.dataset.16S)
#
# Unnecesary
#filtered.dataset.ITS<-
cbind(as.factor(dataset[filtered.dataset.ITS$sample,6]),filtered.dataset.ITS)
#filtered.dataset.16S<-
cbind(factor(dataset[filtered.dataset.16S$sample,6]),filtered.dataset.16S)
#
#filtered.dataset.ITS$QUADRAT[filtered.dataset.ITS$QUADRAT=="A"]<-"7"
#filtered.dataset.16S$QUADRAT[filtered.dataset.16S$QUADRAT=="A"]<-"7"
#filtered.dataset.ITS$QUADRAT[filtered.dataset.ITS$QUADRAT=="B"]<-"8"
#filtered.dataset.16S$QUADRAT[filtered.dataset.16S$QUADRAT=="B"]<-"8"
#dataset$QUADRAT[dataset$QUADRAT=="A"]<-"7"
```

```
#dataset$QUADRAT[dataset$QUADRAT=="B"]<-"8"
filtered.dataset.ITS<-droplevels(filtered.dataset.ITS)
filtered.dataset.16S<-droplevels(filtered.dataset.16S)
dataset<-droplevels(dataset)
dataset<-dataset[with(dataset,order(CODE,LOCALITY,QUADRAT)),]
#
filtered.dataset.ITS$sample<-
factor(as.character(filtered.dataset.ITS$sample),levels=rownames(dataset))
filtered.dataset.16S$sample<-
factor(as.character(filtered.dataset.16S$sample),levels=rownames(dataset))
#
#
# 11. Sort different levels ITS
#
filtered.dataset.ITS$Phylum<-factor(filtered.dataset.ITS$Phylum,levels=
       c("Ascomycota",
       "Basidiomycota",
       "Mortierellomycota",
       "Glomeromycota",
       "Mucoromycota",
       "Rozellomycota",
       "Chytridiomycota",
       "Entorrhizomycota",
       "Olpidiomycota",
       "Aphelidiomycota",
       "Entomophthoromycota",
       "Kickxellomycota",
       "Blastocladiomycota",
       "unidentified"))
#
colnames(filtered.dataset.ITS)[1]<-"Zone"
colnames(filtered.dataset.16S)[1]<-"Zone"

sort.list<-c(
"Ascomycota Archaeorhizomycetes Archaeorhizomycetales Archaeorhizomycetaceae",
"Ascomycota Archaeorhizomycetes unidentified unidentified",
"Ascomycota Arthoniomycetes Lichenostigmatales Phaeococcomycetaceae",
"Ascomycota Dothideomycetes Abrothallales Abrothallaceae",
"Ascomycota Dothideomycetes Botryosphaeriales Botryosphaeriaceae",
"Ascomycota Dothideomycetes Capnodiales Capnodiales_fam_Incertae_sedis",
"Ascomycota Dothideomycetes Capnodiales Cladosporiaceae",
"Ascomycota Dothideomycetes Capnodiales Extremaceae",
"Ascomycota Dothideomycetes Capnodiales Mycosphaerellaceae",
"Ascomycota Dothideomycetes Capnodiales Teratosphaeriaceae",
"Ascomycota Dothideomycetes Capnodiales unidentified",
"Ascomycota Dothideomycetes Dothideales Aureobasidiaceae",
"Ascomycota Dothideomycetes Dothideales Dothideaceae",
"Ascomycota Dothideomycetes Dothideales Dothideales_fam_Incertae_sedis",
"Ascomycota Dothideomycetes Dothideales Dothioraceae",
"Ascomycota Dothideomycetes Jahnulales Aliquandostipitaceae",
"Ascomycota Dothideomycetes Minutisphaerales Minutisphaeraceae",
"Ascomycota Dothideomycetes Mytilinidales Gloniaceae",
"Ascomycota Dothideomycetes Pleosporales Cucurbitariaceae",
"Ascomycota Dothideomycetes Pleosporales Didymellaceae",
"Ascomycota Dothideomycetes Pleosporales Didymosphaeriaceae",
"Ascomycota Dothideomycetes Pleosporales Lentitheciaceae",
"Ascomycota Dothideomycetes Pleosporales Leptosphaeriaceae",
"Ascomycota Dothideomycetes Pleosporales Lindgomycetaceae",
"Ascomycota Dothideomycetes Pleosporales Massarinaceae",
"Ascomycota Dothideomycetes Pleosporales Melanommataceae",
"Ascomycota Dothideomycetes Pleosporales Phaeosphaeriaceae",
"Ascomycota Dothideomycetes Pleosporales Pleomassariaceae",
"Ascomycota Dothideomycetes Pleosporales Pleosporaceae",
"Ascomycota Dothideomycetes Pleosporales Pleosporales_fam_Incertae_sedis",
"Ascomycota Dothideomycetes Pleosporales Sporormiaceae",
"Ascomycota Dothideomycetes Pleosporales unidentified",
"Ascomycota Dothideomycetes Tubeufiales Tubeufiaceae",
"Ascomycota Dothideomycetes Venturiales Sympoventuriaceae",
"Ascomycota Dothideomycetes Venturiales Venturiaceae",
"Ascomycota Dothideomycetes unidentified unidentified",
"Ascomycota Eurotiomycetes Chaetothyriales Chaetothyriales_fam_Incertae_sedis",
"Ascomycota Eurotiomycetes Chaetothyriales Cyphellophoraceae",
"Ascomycota Eurotiomycetes Chaetothyriales Herpotrichiellaceae",
"Ascomycota Eurotiomycetes Chaetothyriales Trichomeriaceae",
```

```
"Ascomycota Eurotiomycetes Chaetothyriales unidentified",
"Ascomycota Eurotiomycetes Eurotiales Aspergillaceae",
"Ascomycota Eurotiomycetes Eurotiales Thermoascaceae",
"Ascomycota Eurotiomycetes Eurotiales Trichocomaceae",
"Ascomycota Eurotiomycetes Onygenales Arthrodermataceae",
"Ascomycota Eurotiomycetes Verrucariales Verrucariaceae",
"Ascomycota Eurotiomycetes Verrucariales unidentified",
"Ascomycota Eurotiomycetes unidentified unidentified",
"Ascomycota GS37 GS37 unidentified",
"Ascomycota Geoglossomycetes Geoglossales Geoglossaceae",
"Ascomycota Geoglossomycetes Geoglossales unidentified",
"Ascomycota Lecanoromycetes Acarosporales Acarosporaceae",
"Ascomycota Lecanoromycetes Acarosporales unidentified",
"Ascomycota Lecanoromycetes Agyriales unidentified",
"Ascomycota Lecanoromycetes Caliciales Physciaceae",
"Ascomycota Lecanoromycetes Candelariales Candelariaceae",
"Ascomycota Lecanoromycetes Lecanorales Cladoniaceae",
"Ascomycota Lecanoromycetes Lecanorales Lecanoraceae",
"Ascomycota Lecanoromycetes Lecanorales Micareaceae",
"Ascomycota Lecanoromycetes Lecanorales Parmeliaceae",
"Ascomycota Lecanoromycetes Lecanorales Porpidiaceae",
"Ascomycota Lecanoromycetes Lecanorales Ramalinaceae",
"Ascomycota Lecanoromycetes Lecanorales Squamarinaceae",
"Ascomycota Lecanoromycetes Lecanorales Stereocaulaceae",
"Ascomycota Lecanoromycetes Lecanorales unidentified",
"Ascomycota Lecanoromycetes Lecanoromycetes_ord_Incertae_sedis
Lecanoromycetes_fam_Incertae_sedis",
"Ascomycota Lecanoromycetes Lecideales Lecideaceae",
"Ascomycota Lecanoromycetes Ostropales Stictidaceae",
"Ascomycota Lecanoromycetes Ostropales unidentified",
"Ascomycota Lecanoromycetes Peltigerales Peltigeraceae",
"Ascomycota Lecanoromycetes Pertusariales Icmadophilaceae",
"Ascomycota Lecanoromycetes Pertusariales Megasporaceae",
"Ascomycota Lecanoromycetes Rhizocarpales Rhizocarpaceae",
"Ascomycota Lecanoromycetes Teloschistales Teloschistaceae",
"Ascomycota Lecanoromycetes Trapeliales Trapeliaceae",
"Ascomycota Lecanoromycetes Trapeliales Trapeliales_fam_Incertae_sedis",
"Ascomycota Lecanoromycetes Umbilicariales Umbilicariaceae",
"Ascomycota Lecanoromycetes unidentified unidentified",
"Ascomycota Leotiomycetes Helotiales Dermateaceae",
"Ascomycota Leotiomycetes Helotiales Helotiaceae",
"Ascomycota Leotiomycetes Helotiales Helotiales_fam_Incertae_sedis",
"Ascomycota Leotiomycetes Helotiales Hyaloscyphaceae",
"Ascomycota Leotiomycetes Helotiales Leotiaceae",
"Ascomycota Leotiomycetes Helotiales Myxotrichaceae",
"Ascomycota Leotiomycetes Helotiales Rutstroemiaceae",
"Ascomycota Leotiomycetes Helotiales Sclerotiniaceae",
"Ascomycota Leotiomycetes Helotiales Vibrisseaceae",
"Ascomycota Leotiomycetes Helotiales unidentified",
"Ascomycota Leotiomycetes Phacidiales Phacidiaceae",
"Ascomycota Leotiomycetes Phacidiales unidentified",
"Ascomycota Leotiomycetes Rhytismatales Rhytismataceae",
"Ascomycota Leotiomycetes Rhytismatales unidentified",
"Ascomycota Leotiomycetes Thelebolales Pseudeurotiaceae",
"Ascomycota Leotiomycetes Thelebolales Thelebolaceae",
"Ascomycota Leotiomycetes Thelebolales unidentified",
"Ascomycota Leotiomycetes unidentified unidentified",
"Ascomycota Orbiliomycetes GS33 unidentified",
"Ascomycota Orbiliomycetes Orbiliales Orbiliaceae",
"Ascomycota Orbiliomycetes Orbiliales Orbiliales_fam_Incertae_sedis",
"Ascomycota Orbiliomycetes Orbiliales unidentified",
"Ascomycota Orbiliomycetes unidentified unidentified",
"Ascomycota Pezizomycetes Pezizales Ascobolaceae",
"Ascomycota Pezizomycetes Pezizales Pezizaceae",
"Ascomycota Pezizomycetes Pezizales Pyronemataceae",
"Ascomycota Pezizomycotina_cls_Incertae_sedis Pezizomycotina_ord_Incertae_sedis
Pezizomycotina_fam_Incertae_sedis",
"Ascomycota Saccharomycetes Saccharomycetales Debaryomycetaceae",
"Ascomycota Saccharomycetes Saccharomycetales Saccharomycetales_fam_Incertae_sedis",
"Ascomycota Saccharomycetes Saccharomycetales unidentified",
"Ascomycota Saccharomycetes unidentified unidentified",
"Ascomycota Sordariomycetes Chaetosphaeriales Chaetosphaeriaceae",
"Ascomycota Sordariomycetes Coniochaetales Coniochaetaceae",
"Ascomycota Sordariomycetes Coniochaetales unidentified",
```

```
"Ascomycota Sordariomycetes Diaporthales Diaporthaceae",
"Ascomycota Sordariomycetes Diaporthales Gnomoniaceae",
"Ascomycota Sordariomycetes Diaporthales Valsaceae",
"Ascomycota Sordariomycetes Diaporthales unidentified",
"Ascomycota Sordariomycetes Glomerellales Glomerellaceae",
"Ascomycota Sordariomycetes Glomerellales Plectosphaerellaceae",
"Ascomycota Sordariomycetes Hypocreales Bionectriaceae",
"Ascomycota Sordariomycetes Hypocreales Clavicipitaceae",
"Ascomycota Sordariomycetes Hypocreales Cordycipitaceae",
"Ascomycota Sordariomycetes Hypocreales Hypocreaceae",
"Ascomycota Sordariomycetes Hypocreales Hypocreales_fam_Incertae_sedis",
"Ascomycota Sordariomycetes Hypocreales Nectriaceae",
"Ascomycota Sordariomycetes Hypocreales Ophiocordycipitaceae",
"Ascomycota Sordariomycetes Hypocreales Stachybotryaceae",
"Ascomycota Sordariomycetes Hypocreales unidentified",
"Ascomycota Sordariomycetes Magnaporthales Magnaporthaceae",
"Ascomycota Sordariomycetes Microascales Microascaceae",
"Ascomycota Sordariomycetes Microascales Microascales_fam_Incertae_sedis",
"Ascomycota Sordariomycetes Myrmecridiales Myrmecridiaceae",
"Ascomycota Sordariomycetes Phomatosporales Phomatosporaceae",
"Ascomycota Sordariomycetes Sordariales Chaetomiaceae",
"Ascomycota Sordariomycetes Sordariales Lasiosphaeriaceae",
"Ascomycota Sordariomycetes Sordariales Sordariaceae",
"Ascomycota Sordariomycetes Sordariales Sordariales_fam_Incertae_sedis",
"Ascomycota Sordariomycetes Sordariales unidentified",
"Ascomycota Sordariomycetes Xylariales Amphisphaeriaceae",
"Ascomycota Sordariomycetes Xylariales Apiosporaceae",
"Ascomycota Sordariomycetes Xylariales Bartaliniaceae",
"Ascomycota Sordariomycetes Xylariales Beltraniaceae",
"Ascomycota Sordariomycetes Xylariales Hyponectriaceae",
"Ascomycota Sordariomycetes Xylariales Microdochiaceae",
"Ascomycota Sordariomycetes Xylariales unidentified",
"Ascomycota Sordariomycetes unidentified unidentified",
"Ascomycota Taphrinomycetes Taphrinales Protomycetaceae",
"Ascomycota Taphrinomycetes Taphrinales Taphrinaceae",
"Ascomycota Xylonomycetes GS34 unidentified",
"Ascomycota unidentified unidentified unidentified",
"Basidiomycota Agaricomycetes Agaricales Agaricaceae",
"Basidiomycota Agaricomycetes Agaricales Amanitaceae",
"Basidiomycota Agaricomycetes Agaricales Bolbitiaceae",
"Basidiomycota Agaricomycetes Agaricales Clavariaceae",
"Basidiomycota Agaricomycetes Agaricales Cortinariaceae",
"Basidiomycota Agaricomycetes Agaricales Crepidotaceae",
"Basidiomycota Agaricomycetes Agaricales Entolomataceae",
"Basidiomycota Agaricomycetes Agaricales Hydnangiaceae",
"Basidiomycota Agaricomycetes Agaricales Hygrophoraceae",
"Basidiomycota Agaricomycetes Agaricales Hymenogastraceae",
"Basidiomycota Agaricomycetes Agaricales Inocybaceae",
"Basidiomycota Agaricomycetes Agaricales Lycoperdaceae",
"Basidiomycota Agaricomycetes Agaricales Lyophyllaceae",
"Basidiomycota Agaricomycetes Agaricales Pleurotaceae",
"Basidiomycota Agaricomycetes Agaricales Psathyrellaceae",
"Basidiomycota Agaricomycetes Agaricales Strophariaceae",
"Basidiomycota Agaricomycetes Agaricales Tricholomataceae",
"Basidiomycota Agaricomycetes Agaricales unidentified",
"Basidiomycota Agaricomycetes Amylocorticiales Amylocorticiaceae",
"Basidiomycota Agaricomycetes Atheliales Atheliaceae",
"Basidiomycota Agaricomycetes Atheliales unidentified",
"Basidiomycota Agaricomycetes Auriculariales unidentified",
"Basidiomycota Agaricomycetes Boletales Boletaceae",
"Basidiomycota Agaricomycetes Boletales Boletales_fam_Incertae_sedis",
"Basidiomycota Agaricomycetes Boletales Coniophoraceae",
"Basidiomycota Agaricomycetes Boletales Paxillaceae",
"Basidiomycota Agaricomycetes Boletales Rhizopogonaceae",
"Basidiomycota Agaricomycetes Boletales Suillaceae",
"Basidiomycota Agaricomycetes Boletales Tapinellaceae",
"Basidiomycota Agaricomycetes Cantharellales Cantharellales_fam_Incertae_sedis",
"Basidiomycota Agaricomycetes Cantharellales Ceratobasidiaceae",
"Basidiomycota Agaricomycetes Cantharellales Clavulinaceae",
"Basidiomycota Agaricomycetes Cantharellales unidentified",
"Basidiomycota Agaricomycetes Corticiales Corticiaceae",
"Basidiomycota Agaricomycetes GS29 unidentified",
"Basidiomycota Agaricomycetes Geastrales Geastraceae",
"Basidiomycota Agaricomycetes Hymenochaetales Hymenochaetaceae",
```

```
"Basidiomycota Agaricomycetes Hymenochaetales Hymenochaetales_fam_Incertae_sedis",
"Basidiomycota Agaricomycetes Hymenochaetales Rickenellaceae",
"Basidiomycota Agaricomycetes Hymenochaetales Schizoporaceae",
"Basidiomycota Agaricomycetes Hymenochaetales Tubulicrinaceae",
"Basidiomycota Agaricomycetes Jaapiales Jaapiaceae",
"Basidiomycota Agaricomycetes Polyporales Coriolaceae",
"Basidiomycota Agaricomycetes Polyporales Fomitopsidaceae",
"Basidiomycota Agaricomycetes Polyporales Ganodermataceae",
"Basidiomycota Agaricomycetes Polyporales Hyphodermataceae",
"Basidiomycota Agaricomycetes Polyporales Meripilaceae",
"Basidiomycota Agaricomycetes Polyporales Meruliaceae",
"Basidiomycota Agaricomycetes Polyporales Podoscyphaceae",
"Basidiomycota Agaricomycetes Russulales Albatrellaceae",
"Basidiomycota Agaricomycetes Russulales Bondarzewiaceae",
"Basidiomycota Agaricomycetes Russulales Peniophoraceae",
"Basidiomycota Agaricomycetes Russulales Russulaceae",
"Basidiomycota Agaricomycetes Russulales Stereaceae",
"Basidiomycota Agaricomycetes Russulales unidentified",
"Basidiomycota Agaricomycetes Sebacinales Sebacinaceae",
"Basidiomycota Agaricomycetes Sebacinales Serendipitaceae",
"Basidiomycota Agaricomycetes Sebacinales unidentified",
"Basidiomycota Agaricomycetes Thelephorales Thelephoraceae",
"Basidiomycota Agaricomycetes Trechisporales Hydnodontaceae",
"Basidiomycota Agaricomycetes Trechisporales unidentified",
"Basidiomycota Agaricomycetes Tremellodendropsidales unidentified",
"Basidiomycota Agaricomycetes unidentified unidentified",
"Basidiomycota Agaricostilbomycetes Agaricostilbales Chionosphaeraceae",
"Basidiomycota Agaricostilbomycetes Agaricostilbales Kondoaceae",
"Basidiomycota Cystobasidiomycetes Erythrobasidiales
Erythrobasidiales_fam_Incertae_sedis",
"Basidiomycota Cystobasidiomycetes Erythrobasidiales unidentified",
"Basidiomycota Exobasidiomycetes Entylomatales Entylomataceae",
"Basidiomycota Exobasidiomycetes Exobasidiales Exobasidiaceae",
"Basidiomycota Exobasidiomycetes Exobasidiales unidentified",
"Basidiomycota Exobasidiomycetes Georgefischeriales Tilletiariaceae",
"Basidiomycota GS27 GS27 unidentified",
"Basidiomycota Malasseziomycetes Malasseziales Malasseziaceae",
"Basidiomycota Microbotryomycetes Leucosporidiales Leucosporidiaceae",
"Basidiomycota Microbotryomycetes Leucosporidiales unidentified",
"Basidiomycota Microbotryomycetes Microbotryomycetes_ord_Incertae_sedis Chrysozymaceae",
"Basidiomycota Microbotryomycetes Sporidiobolales Sporidiobolaceae",
"Basidiomycota Microbotryomycetes Sporidiobolales unidentified",
"Basidiomycota Microbotryomycetes unidentified unidentified",
"Basidiomycota Pucciniomycetes Platygloeales Eocronartiaceae",
"Basidiomycota Pucciniomycetes Septobasidiales Septobasidiaceae",
"Basidiomycota Pucciniomycetes unidentified unidentified",
"Basidiomycota Spiculogloeomycetes Spiculogloeales Spiculogloeaceae",
"Basidiomycota Tremellomycetes Cystofilobasidiales
Cystofilobasidiales_fam_Incertae_sedis",
"Basidiomycota Tremellomycetes Cystofilobasidiales Mrakiaceae",
"Basidiomycota Tremellomycetes Filobasidiales Filobasidiaceae",
"Basidiomycota Tremellomycetes Filobasidiales Piskurozymaceae",
"Basidiomycota Tremellomycetes Filobasidiales unidentified",
"Basidiomycota Tremellomycetes Holtermanniales Holtermanniales_fam_Incertae_sedis",
"Basidiomycota Tremellomycetes Tremellales Bulleraceae",
"Basidiomycota Tremellomycetes Tremellales Bulleribasidiaceae",
"Basidiomycota Tremellomycetes Tremellales Phaeotremellaceae",
"Basidiomycota Tremellomycetes Tremellales Rhynchogastremataceae",
"Basidiomycota Tremellomycetes Tremellales Tremellaceae",
"Basidiomycota Tremellomycetes Tremellales Trimorphomycetaceae",
"Basidiomycota Tremellomycetes Tremellales unidentified",
"Basidiomycota Tremellomycetes Trichosporonales Tetragoniomycetaceae",
"Basidiomycota Tremellomycetes Trichosporonales Trichosporonaceae",
"Basidiomycota Tremellomycetes unidentified unidentified",
"Basidiomycota Ustilaginomycetes Urocystidales Urocystidaceae",
"Basidiomycota Ustilaginomycetes Ustilaginales Ustilaginaceae",
"Basidiomycota unidentified unidentified unidentified",
"Aphelidiomycota Aphelidiomycetes GS16 unidentified",
"Blastocladiomycota unidentified unidentified unidentified",
"Chytridiomycota Chytridiomycetes Chytridiales Chytridiaceae",
"Chytridiomycota Chytridiomycetes unidentified unidentified",
"Chytridiomycota Lobulomycetes Lobulomycetales Lobulomycetaceae",
"Chytridiomycota Rhizophydiomycetes Rhizophydiales Alphamycetaceae",
"Chytridiomycota Rhizophydiomycetes Rhizophydiales Terramycetaceae",
```

```
"Chytridiomycota Spizellomycetes Spizellomycetales Spizellomycetaceae",
"Chytridiomycota unidentified unidentified unidentified",
"Entomophthoromycota Basidiobolomycetes Basidiobolales Basidiobolaceae",
"Entorrhizomycota Entorrhizomycetes Entorrhizales Entorrhizaceae",
"Glomeromycota Archaeosporomycetes Archaeosporales Ambisporaceae",
"Glomeromycota Archaeosporomycetes Archaeosporales Archaeosporaceae",
"Glomeromycota Archaeosporomycetes Archaeosporales unidentified",
"Glomeromycota Glomeromycetes Diversisporales Acaulosporaceae",
"Glomeromycota Glomeromycetes Diversisporales Diversisporaceae",
"Glomeromycota Glomeromycetes Gigasporales Gigasporaceae",
"Glomeromycota Glomeromycetes Glomerales Claroideoglomeraceae",
"Glomeromycota Glomeromycetes Glomerales Glomeraceae",
"Glomeromycota Glomeromycetes Glomerales unidentified",
"Glomeromycota unidentified unidentified unidentified",
"Kickxellomycota Harpellomycetes Harpellales Legeriomycetaceae",
"Mortierellomycota Mortierellomycetes Mortierellales Mortierellaceae",
"Mortierellomycota Mortierellomycetes Mortierellales unidentified",
"Mortierellomycota unidentified unidentified unidentified",
"Mucoromycota Endogonomycetes Endogonales Endogonaceae",
"Mucoromycota Endogonomycetes Endogonales unidentified",
"Mucoromycota Endogonomycetes GS22 unidentified",
"Mucoromycota Mucoromycetes Mucorales Mucoraceae",
"Mucoromycota Umbelopsidomycetes Umbelopsidales Umbelopsidaceae",
"Olpidiomycota Olpidiomycetes Olpidiales Olpidiaceae",
"Olpidiomycota Olpidiomycetes Olpidiales unidentified",
"Rozellomycota Rozellomycotina_cls_Incertae_sedis GS06 unidentified",
"Rozellomycota Rozellomycotina_cls_Incertae_sedis GS11 unidentified",
"Rozellomycota unidentified unidentified unidentified",
"unidentified unidentified unidentified unidentified")
#
# 11.2 rename unidentified
#
filtered.dataset.ITS$Class<-
factor(filtered.dataset.ITS$Class,levels=c(as.character(with(data.frame(x=sapply(strspli
t(sort.list," "),`[`,2)),x[!duplicated(x)&x!="unidentified"])),"unidentified"))
filtered.dataset.ITS$Order<-
factor(filtered.dataset.ITS$Order,levels=c(as.character(with(data.frame(x=sapply(strspli
t(sort.list," "),`[`,3)),x[!duplicated(x)&x!="unidentified"])),"unidentified"))
filtered.dataset.ITS$Family<-
factor(filtered.dataset.ITS$Family,levels=c(as.character(with(data.frame(x=sapply(strspl
it(sort.list," "),`[`,4)),x[!duplicated(x)&x!="unidentified"])),"unidentified"))
#
#
# 11.3. tabulate. OTUs, filter table.OTUS and dataset by number of reads (must be a bove
50)
#
#
#
table.otus.ITS<-
tabulate.and.print(data=filtered.dataset.ITS,7,9,8,category=colnames(filtered.dataset.IT
S)[i],print=FALSE,normalize=FALSE)
table.otus.ITS<-table.otus.ITS[[1]][,1:4427]
filtered.dataset.ITS<-
filtered.dataset.ITS[filtered.dataset.ITS$otu%in%colnames(table.otus.ITS),]
#
# 12. Import to phyloseq ITS
#
library(phyloseq)
foo<-filtered.dataset.ITS[!duplicated(filtered.dataset.ITS$otu),11:17] #postrepaired
rownames(foo)<-filtered.dataset.ITS[!duplicated(filtered.dataset.ITS$otu),9]
#
rownames(foo)<-gsub("[[:blank:]]","",rownames(foo)) #post repaired
phyloseq.ITS<-
phyloseq(otu_table(table.otus.ITS,taxa_are_rows=F),sample_data(dataset),tax_table(as.mat
rix(foo[order(as.numeric(rownames(foo))),])))
#
# aposteriori fix tax_table(phyloseq.ITS)<-
tax_table(as.matrix(foo[order(as.numeric(rownames(foo))),]))
#
#
# 13. 16S
#
# Reorder OTUs import to phyloseq 16S
```

```
filtered.dataset.16S$otu<-
factor(filtered.dataset.16S$otu,levels=order(sort(as.numeric(filtered.dataset.16S$otu)))
)
#table.otus.16S<-
tabulate.and.print(data=filtered.dataset.16S,7,9,8,category=colnames(filtered.dataset.16
S)[i],print=FALSE,normalize=FALSE)
table.otus.16S<-xtabs(size~otu+sample,data=filtered.dataset.16S)
foo<-filtered.dataset.16S[!duplicated(filtered.dataset.16S$otu),11:17]
rownames(foo)<-filtered.dataset.16S[!duplicated(filtered.dataset.16S$otu),9]
rownames(foo)<-gsub("[[:blank:]]","",rownames(foo)) #post repaired
class(table.otus.16S)<-"matrix"
library(ape)
seqs.16S<-
read.dna("/Users/ferninfm/Desktop/microbiome/16S/complete_16S_1f_representatives.fas",fo
rmat="fasta")
#!!! missing strsplit names seqs.16S
filtered.seqs.16S<-seqs.16S[sort(as.numeric(rownames(foo)))]
names(filtered.seqs.16S)<-paste("otu",sort(as.numeric(rownames(foo))),sep="_")
write.dna(filtered.seqs.16S,"~/Desktop/microbiome/16S/filtered_16S_representatives.fas",
format="fasta")
#
# FORGOT TO ADD TREE
#
phyloseq.16S<-
phyloseq(otu_table(table.otus.16S,taxa_are_rows=T),sample_data(dataset),tax_table(as.mat
rix(foo[order(as.numeric(rownames(foo))),])))
# Fix names tax_table(phyloseq.16S)<-
tax_table(as.matrix(foo[order(as.numeric(rownames(foo))),]))
#
tree.16Sp<-read.tree("~/Desktop/microbiome/16S/16S_NS2_fasttree")
tree.16Sp$tip.label<-sapply(strsplit(tree.16Sp$tip.label,"_"),`[`,2)
require(phangorn)
phy_tree(phyloseq.16S)<-phy_tree(tree.16Sp)
#phy tree(phyloseq.16S)<-midpoint(phy tree(phyloseq.16S))

# Now 16S
#



# 11. Save dataset
save.image('~/Desktop/microbiome/datasets_final_all4.Rdata')
```

d. Vorlagen digitale Datenverarbeitung. Datenanalyse mit R und Markdown

````
---
title: "NPHT longterm monitoring project. Template for data analyses"
author: "Fernando Fernandez Mendoza"
date: "4/9/2019"
output:
  html_document: default
  word_document: default
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

# Results
## Description of the dataset

Amplicon sequencing of fungal ITS. Sequencing intensity across samples. Barplot heighth
represents the raw number of reads, light colored fraction represents chimeric reads
excluded from the analysis. Yellow highlights underepresented sample points excluded
from the analysis: four samples from UN, two from IG and one from from Furca pass (FU);
together the mean two from the transect head (K), two from the middle section (M) and
three from the base (T)
```{r}
load('/Users/ferninfm/Desktop/05_Manuscripts/02_microbiome/datasets_phyloseq.Rdata')
load('/Users/ferninfm/Desktop/05_Manuscripts/02_microbiome/filtered_unfiltered_backups.R
data')
library(knitr)
library(xtable)
````
````

```
library(ggplot2)
library(reshape2)
library(grid)
library(phyloseq)
library(vegan)
library(lme4)
library(stargazer)
library(edgeR)
source("/Users/ferninfm/Desktop/05_Manuscripts/02_microbiome/functions.r")
#
colnames(unfiltered.dataset.ITS)[8]<-"size"
aggr<-aggregate(size~sample,data=filtered.dataset.ITS,sum)
aggr2<-aggregate(size~sample,data=unfiltered.dataset.ITS,sum)
pops<-rbind(aggr[,2],aggr2[,2])
pops[2,]<-pops[2,]-pops[1,]
colnames(pops)<-aggr[,1]
out_samples_ITS<-colnames(pops)[pops[1,]<=100000]
pops<-rbind(pops,as.numeric(pops[1,]<=100000)*pops[1,])
pops[1,]<-as.numeric(pops[1,]>=100000)*pops[1,]
#
#
#
# FIXING OF OBERETTES?
#
#
sample_data(phyloseq.ITS)[c("OB1-c6da",
"OB1-c6dul",
"OB1-c6dur",
"OB2-c6da",
"OB2-c6dul",
"OB2-c6dur",
"OB3-c7da",
"OB3-c7dul",
"OB3-c7dur",
"OB1-c4da",
"OB1-c4dul",
"OB1-c4dur",
"OB2-c4da",
"OB2-c4dul",
"OB2-c4dur",
"OB3-c4da",
"OB3-c4dul",
"OB3-c4dur",
"OB1-c1da",
"OB1-c1dli",
"OB1-c1dre",
"OB2-c1da",
"OB2-c1dur",
"OB3-c1da",
"OB3-c1dul",
"OB3-c1dur"),6]<-c("T",
"T",
"T",
"T",
"T",
"T",
"T",
"T",
"T",
"M",
"M",
"M",
"M",
"M",
"M",
"M",
"M",
"M",
"K",
"K",
"K",
"K",
"K",
"K",
```

```
"K",
"K")
#
sample_data(phyloseq.16S)[c("OB1-c6da",
"OB1-c6dul",
"OB1-c6dur",
"OB2-c6da",
"OB2-c6dul",
"OB2-c6dur",
"OB3-c7da",
"OB3-c7dul",
"OB3-c7dur",
"OB1-c4da",
"OB1-c4dul",
"OB1-c4dur",
"OB2-c4da",
"OB2-c4dul",
"OB2-c4dur",
"OB3-c4da",
"OB3-c4dul",
"OB3-c4dur",
"OB1-c1da",
"OB1-c1dli",
"OB1-c1dre",
"OB2-c1da",
"OB2-c1dur",
"OB3-c1da",
"OB3-c1dul",
"OB3-c1dur"),6]<-c("T",
"T",
"T",
"T",
"T",
"T",
"T",
"T",
"M",
"M",
"M",
"M",
"M",
"M",
"M",
"M",
"K",
"K",
"K",
"K",
"K",
"K",
"K",
"K")
#
# Subset samples for national Park
#
sub_NP<-
c("IG1A1C","IG1A2B","IG1BTD","IG1C3C","IG1CKA","IG2A1C","IG2AKC","IG2C4D","IG2C5D","IG2C
KB","IG2CTD","IG3A1A","IG3A3A","IG3A5A","IG3A6A-
C","IG3AKP","IG3BKB","IG4A2A","IG4A4C","IG4A6C","IG4A8C","IG4ATC","IG4CBD","IG5A3C","IG5
A4C","IG5ATA-
C","IG5C2D","IG5C4D","IG5CKD","SE1A4A","SE1A6D","SE1A7D","SE1ATA","SE1C1B","SE1C3D","SE1
C5B","SE1C7A","SE2A2D","SE2A3D","SE2AKB","SE2ATC","SE2C2C","SE2C5A","SE2C5D","SE2CTC","S
E3A1B","SE3A4B","SE3AKB","SE3ATB","SE3BKB","SE3BTA","SE3C1A","SE3C1B","SE3C4D","UN1A2C",
"UN1BKD","UN1BTA","UN1C4A","UN1C6D","UN1C7A","UN2A6A","UN2ATB","UN2BKA","UN2C1C","UN2C3A
","UN2C6C","UN3A6C","UN3BTD","UN3C4D","UN3C5A-
C","UN4A1A","UN4A6C","UN4BKA","UN4BTC","UN4C4A","UN5A3A","UN5A6D","UN5BTD","UN5C1A","UN5
CKC","UN6A3C","UN6A5C","UN6BTA","UN6C5A")
#
# Prune for NP
#
phyloseq.ITS<-prune_samples(sub_NP,phyloseq.ITS)
phyloseq.ITS<-prune_taxa(colSums(phyloseq.ITS@otu_table)!=0,phyloseq.ITS)
```

```
phyloseq.16S<-prune_samples(sub_NP,phyloseq.16S)
# Watchout for inconsistency
phyloseq.16S<-prune_taxa(rowSums(phyloseq.16S@otu_table)!=0,phyloseq.16S)
phyloseq.16S<-
prune_taxa(taxa_names(phyloseq.16S)[phyloseq.16S@tax_table[,3]!="Chloroplast"],phyloseq.
16S)
pops<-pops[,sub_NP]
# PLOT
theme_set(theme_bw())
ggplot(
       melt(t(pops)),
       aes(x=Var1, y=value, fill=factor(Var2, levels=c("2","1","3")))) +
       geom_bar(stat="identity", width=.8) +
       ylab("Number of reads") +
       xlab("Sample sorted by locality, transect and position") +
       guides(fill=FALSE) +
       scale_fill_manual(values = c("#b2abd2","#542788","#e08214")) +
       theme_bw() +
       theme(axis.text.x = element_text(angle = 45, hjust = 1,size=4))
```

Description of Both datasets before filtering

```{r}
phyloseq.ITS
sum(phyloseq.ITS@otu_table)
phyloseq.16S
sum(phyloseq.16S@otu_table)
```

Uniqueness of OTUs on the ITS data matrix, chimeric sequences are excluded. Each point is an OTU mapped in space in fuction of its total size (the total number of reads across all samples) and the number of samples in which it is present. The blue discontinuous line represent the median value of samples in which individual OTUs are found

```{r}
ggplot(
       data.frame(
              sample.sums=colSums(!(otu_table(phyloseq.ITS)==0)),
              read.sums=colSums(otu_table(phyloseq.ITS))),
              aes(y=read.sums,x=sample.sums)) +
        geom_point(
              size=2, alpha = 0.3, colour = "#fc8d59") +
       theme_bw() +
       scale_y_continuous(
              trans='log10') +
       labs(
              x="Number of Samples",
              y="Cumulative number of reads") +
       geom_vline(
              xintercept=median(colSums(!(otu_table(phyloseq.ITS)==0))),
linetype="dotted",color = "blue", size=0.5)
#plot(colSums(!(otu_table(phyloseq.ITS)==0)),pch=16,col="#fc8d5920",ylab="Number of
Samples",xlab="Number of OTUs")
#abline (h=median(colSums(!(otu_table(phyloseq.ITS)==0))),col="blue",lty=3)
```

Stacked Density profiles of sequencing depth per OTU in the ITS dataset. Color code separates OTUs found in a single sample from those present in more than one.

SUBSET Phyloseq Objects by UNUSABLE SAMPLES

```{r}
phyloseq.ITS<-
subset_samples(phyloseq.ITS,!(rownames(sample_data(phyloseq.ITS))%in%out_samples_ITS))
#
# Plot
#
df<-data.frame(
weight=colSums(otu_table(phyloseq.ITS)),TF=colSums((otu_table(phyloseq.ITS)!=0))==1)
#
ggplot(df,
aes(log(weight,base=10),color=TF,fill=TF)) +
geom_density(adjust=0.5,stat="density",position="stack") +
scale_color_manual (values=c("#e08214","#542788")) +
```

```
scale_fill_manual (labels = c(" +1 sample", " 1
sample"),values=c("#e0821488","#54278888")) +
guides(fill = guide_legend(reverse=TRUE)) +
theme(legend.position=c(.85,.85),legend.title=element_blank()) +
geom_vline(aes(xintercept=log(300,base=10)), color="#542788", linetype="dashed", size=1)
+
xlab("logarithm of OTU sizes") + guides(color=FALSE)
#
# SUBSET Phyloseq Objects by UNUSABLE SAMPLES
#
# SUBSET Phyloseq object by UNDERREPRESENTED / SIngleton OTUS
# Criterion OTUS found in a single sample & which have less than 300 reads for ITS
#
#
badTaxa<-
names(colSums(otu_table(phyloseq.ITS)[,colSums(!(otu_table(phyloseq.ITS)==0))==1])<=100)
[(colSums(otu_table(phyloseq.ITS)[,colSums(!(otu_table(phyloseq.ITS)==0))==1])<=300)]
goodTaxa <- setdiff(taxa_names(phyloseq.ITS), badTaxa)
#
# PRUNE TAXA ITS
#
phyloseq.ITS<-prune_taxa(goodTaxa,phyloseq.ITS)
#
rm(badTaxa,goodTaxa)
```

Amplicon sequencing of the bacterial 16S variable region. Sequencing intensity across samples. Barplot heighth represents the raw number of reads, light colored fraction represents chimeric reads excluded from the analysis. Yellow highlights underepresented sample points excluded from the analysis: one sample from IG and one from OB

```{r}
aggr<-aggregate(size~sample,data=filtered.dataset.16S,sum)
aggr2<-aggregate(size~sample,data=unfiltered.dataset.16S,sum)
rownames(aggr)<-aggr[,1]
rownames(aggr2)<-aggr2[,1]
aggr2<-aggr2[rownames(aggr),]
pops<-rbind(aggr[,2],aggr2[,2])
pops[2,]<-pops[2,]-pops[1,]
colnames(pops)<-aggr[,1]
out_samples_16S<-colnames(pops)[pops[1,]<=100000]
pops<-rbind(pops,as.numeric(pops[1,]<=100000)*pops[1,])
pops[1,]<-as.numeric(pops[1,]>=100000)*pops[1,]
pops<-pops[,sub_NP]
#
#
#
ggplot(melt(t(pops)), aes(x=Var1, y=value, fill=factor(Var2, levels=c("2","1","3")))) +
  geom_bar(stat="identity", width=.8) +
  ylab("Number of reads") +
  xlab("Sample sorted by locality, transect and position") +
  guides(fill=FALSE) +
  scale_fill_manual(values = c("#b2abd2","#542788","#e08214")) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, size=4))
```

Uniqueness of OTUs on the 16S data matrix, chimeric sequences are excluded. Each point is an OTU mapped in space in fuction of its total size (the total number of reads across all samples) and the number of samples in which it is present. The blue discontinuous line represent the median value of samples in which individual OTUs are found

```{r}
ggplot(
       data.frame(
       sample.sums=rowSums(!(otu_table(phyloseq.16S)==0)),
       read.sums=rowSums(otu_table(phyloseq.16S))),
       aes(y=read.sums,x=sample.sums
       )
       ) +
       geom_point(size=2, alpha = 0.3, colour = "#fc8d59") +
       theme_bw() +
       scale_y_continuous(trans='log10') +
       labs(x="Number of Samples",y="Cumulative number of reads") +
       geom_vline(
```

```
              xintercept=median(rowSums(!(otu_table(phyloseq.16S)==0))),
linetype="dotted",color = "blue", size=0.5)
#plot(rowSums(!(otu_table(phyloseq.16S)==0)),pch=16,col="#fc8d5920",ylab="Number of
Samples",xlab="Number of OTUs")
#abline (h=median(rowSums(!(otu table(phyloseq.16S)==0))),col="blue",lty=3)
```

Stacked Density profiles of sequencing depth per OTU in the 16S dataset. Color code separates OTUs found in a single sample from those present in more than one sample.

```{r}
phyloseq.16S<-
subset_samples(phyloseq.16S,!(rownames(sample_data(phyloseq.16S))%in%out_samples_16S))
#
# Plot
#
df<-data.frame(
weight=rowSums(otu_table(phyloseq.16S)),TF=rowSums((otu_table(phyloseq.16S)!=0))==1)
#
ggplot(df,
aes(log(weight,base=10),color=TF,fill=TF)) +
geom_density(adjust=0.5,stat="density",position="stack") +
scale_color_manual (values=c("#e08214","#542788")) +
scale_fill_manual (labels = c(" +1 sample", " 1
sample"),values=c("#e0821488","#54278888")) +
guides(fill = guide_legend(reverse=TRUE)) +
theme(legend.position=c(.85,.85),legend.title=element_blank()) +
geom_vline(aes(xintercept=log(100,base=10)), color="#542788", linetype="dashed", size=1)
+
xlab("logarithm of OTU sizes") + guides(color=FALSE)
#
# SUBSET Phyloseq object by UNDERREPRESENTED / SIngleton OTUS 16S
# Criterion OTUS found in a single sample & which have less than 100 reads for 16S
#
#
badTaxa<-
names(rowSums(otu_table(phyloseq.16S)[rowSums(!(otu_table(phyloseq.16S)==0))==1,])<=100)
[(rowSums(otu_table(phyloseq.16S)[rowSums(!(otu_table(phyloseq.16S)==0))==1,])<=100)]
goodTaxa <- setdiff(taxa_names(phyloseq.16S), badTaxa)
#
# PRUNE TAXA 16S
#
phyloseq.16S<-prune_taxa(goodTaxa,phyloseq.16S)
#/Volumes/FFM_Genomes
rm(badTaxa,goodTaxa)
```

##Description of both dataset after filtering

```{r}
phyloseq.16S<-prune_taxa(rowSums(phyloseq.16S@otu_table)!=0,phyloseq.16S)
phyloseq.ITS<-prune_taxa(colSums(phyloseq.ITS@otu_table)!=0,phyloseq.ITS)
phyloseq.ITS
sum(phyloseq.ITS@otu_table)
phyloseq.16S
sum(phyloseq.16S@otu_table)
```

##OTU sharing profile

```{r}
venn_phyloseq<-
function(phyloseq_object,margin="LOCALITY",reads=FALSE,colorinos=c(2,3,4))
  {
  require(limma)
  require(tidyverse)
  require(ggplot2)
  require(ggforce)
  foo_venn <- merge_samples(phyloseq_object,margin)
  cat.names <- rownames(sample_data(foo_venn))
  if (taxa_are_rows(phyloseq_object)==TRUE)
         {
       foo_venn<-t(otu_table(foo_venn))
       }
  if (taxa_are_rows(phyloseq_object)==FALSE)
       {
```

```
       foo_venn<-t(otu_table(foo_venn))
       }
  if (reads==FALSE)
         {
       foo venn<-vennCounts((foo venn))
       }
  if (reads==TRUE)
       {
       foo_venn<-vennCounts(foo_venn,)
  }
#
# https://scriptsandstatistics.wordpress.com/2018/04/26/how-to-plot-venn-diagrams-using-
r-ggplot2-and-ggforce/
#
class(foo_venn) <- "matrix"
df.vdc <- as.data.frame(foo_venn)[-1,] %>%
  mutate(x = c(0, 1.2, 0.8, -1.2, -0.8, 0, 0),
         y = c(1.2, -0.6, 0.5, -0.6, 0.5, -1, 0))
df.venn <- data.frame(x = c(0, 0.866, -0.866),
                      y = c(1, -0.5, -0.5),
                      labels = cat.names)
p1<- ggplot(df.venn) +
  geom_circle(aes(x0 = x, y0 = y, r = 1.5, fill = labels), alpha = .3, size = 1, colour
= 'grey') +
  coord_fixed() +
  theme_void() +
  theme(legend.position = 'bottom') +
  scale_fill_manual(values = colorinos) +
  scale_colour_manual(values = colorinos, guide = FALSE) +
  labs(fill = NULL) +
  annotate("text", x = df.vdc$x, y = df.vdc$y, label = df.vdc$Counts, size = 5)
p1
}

venn_phyloseq(phyloseq.ITS,"LOCALITY",colorinos=c(2,3,4))

```

```{r}
venn_phyloseq(phyloseq.ITS,"CODE",colorinos=c("#542788","#e08214","#f7f7f7"))
```

```{r}
venn_phyloseq(phyloseq.16S,"LOCALITY")

```
```{r}
venn_phyloseq(phyloseq.16S,"CODE",colorinos=c("#542788","#e08214","#f7f7f7"))

```
Taxonomic profile of the ITS dataset

```{r}
# COMNPUTE DATASET
cumulative.ITS <-
data.frame(abundance=colSums(otu_table(phyloseq.ITS)),tax_table(phyloseq.ITS))
cumulative.16S <-
data.frame(abundance=rowSums(otu_table(phyloseq.16S)),tax_table(phyloseq.16S))
norm.ITS <- transform_sample_counts(phyloseq.ITS, function(x){x/sum(x)})
norm.16S <- transform_sample_counts(phyloseq.16S, function(x){x/sum(x)})
#
#
```
Taxonomic composition of the fungal dataset at Class level
```{r}
erase.me<-aggregate(abundance~Class,cumulative.ITS,FUN=sum)
erase.me$Class<-
factor(erase.me$Class,levels=erase.me[order(erase.me[,2],decreasing=T),1])
#
ggplot(erase.me,
       aes(x=Class,y=abundance,fill=Class)) +
       geom_col() +
       theme(legend.position='none') +
       theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
```

Taxonomic composition fungal samples at Class level

```{r, fig.width = 12, fig.height = 5}
foo.plot<-plot_bar(norm.ITS,x="Sample",y="Abundance",fill="Class") +
geom_bar(aes(color=Class, fill=Class), stat="identity", position="stack")
foo.plot + theme(legend.position="none")
```

```{r, fig.width = 12, fig.height = 5}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```
Taxonomic composition of the fungal dataset at Order level
```{r}
erase.me<-aggregate(abundance~Order,cumulative.ITS,FUN=sum)
erase.me$Order<-
factor(erase.me$Order,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[c(1,2,4:41),],aes(x=Order,y=abundance,fill=Order)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))

```
Taxonomic composition fungal samples at Order level

```{r, fig.width = 12, fig.height = 5}
foo.plot<-plot_bar(norm.ITS,x="Sample",y="Abundance",fill="Order") +
geom_bar(aes(color=Order, fill=Order), stat="identity", position="stack") +
theme(legend.position="bottom")
foo.plot + theme(legend.position="none")
```
```{r, fig.width = 12, fig.height = 10}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```


```{r}
erase.me<-aggregate(abundance~Family,cumulative.ITS,FUN=sum)
erase.me$Family<-
factor(erase.me$Family,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:41,],aes(x=Family,y=abundance,fill=Family)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```
```{r}
erase.me<-aggregate(abundance~Genus,cumulative.ITS,FUN=sum)
erase.me$Genus<-
factor(erase.me$Genus,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:41,],aes(x=Genus,y=abundance,fill=Genus)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```
```{r}
erase.me<-aggregate(abundance~Species,cumulative.ITS,FUN=sum)
erase.me$Species<-
factor(erase.me$Species,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:41,],aes(x=Species,y=abundance,fill=Species)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```
The 40 most sequenced fungal OTUs
Appendix 1. ITS
```{r, results='asis'}
erase.me<-cbind(tax_table(phyloseq.ITS),total_reads=colSums(otu_table(phyloseq.ITS)))
erase.me<-head(erase.me[order(as.numeric(erase.me[,8]),decreasing=TRUE),],40)
stargazer(erase.me,type="html")
```

```
```
##Community structure ITS
Non-normalized
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two
```{r}
ITS.ord <- ordinate(phyloseq.ITS, "NMDS", "bray",k=3)
#
p1<-plot_ordination(phyloseq.ITS, ITS.ord, type="taxa", axes=1:2, color="Phylum", title="taxa")
p1+ facet_wrap(~Phylum)
```
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two
```{r}
p1<-plot_ordination(phyloseq.ITS, ITS.ord, type="samples", axes=1:2, color="LOCALITY", shape="CODE", title="samples")
p1 + geom_point(size=2)
```
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two. Faceted by sampling locality
```{r}
p1 + facet_wrap(~LOCALITY) + geom_point(size=2)
```
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two. Faceted by position in the gradient
```{r}
p1 + facet_wrap(~CODE) + geom_point(size=2)
```
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three
```{r}
 #

p2<-plot_ordination(phyloseq.ITS, ITS.ord, type="samples", axes=2:3, color="LOCALITY", shape="CODE", title="samples")
p2 + geom_point(size=2)
```
Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three. Faceted by Sampling locality
```{r}
p2 + facet_wrap(~LOCALITY) + geom_point(size=2)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three. Faceted by position in the gradient
```{r}
p2 + facet_wrap(~CODE) + geom_point(size=2)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three
```{r}
p3<-plot_ordination(phyloseq.ITS, ITS.ord, type="samples", axes=c(1,3), color="LOCALITY", shape="CODE", title="samples")
p3 + geom_point(size=2)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three. Faceted by LOCALITY
```{r}
p3 + facet_wrap(~LOCALITY) + geom_point(size=2)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three. Faceted by position in the gradient
```{r}
p3 + facet_wrap(~CODE) + geom_point(size=2)
rm(p1,p2,p3)
```
Normalized
```
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two

```{r, fig.width=8, fig.height=8}
ITS.ord <- ordinate(norm.ITS, "NMDS", "bray",k=3)
#
p1<-plot_ordination(norm.ITS, ITS.ord, type="taxa", axes=1:2, color="Phylum",
title="taxa")
p1+ facet_wrap(~Phylum) + theme(legend.position="none")
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two

```{r, fig.width=10, fig.height=10}
p1<-plot_ordination(norm.ITS, ITS.ord, type="taxa", axes=1:2, color="Order",
title="taxa")
p1 + facet_wrap(~Class) + theme(legend.position="none")
```

```{r, fig.width = 12, fig.height = 10}
legend <- cowplot::get_legend(p1)
grid.newpage()
grid.draw(legend)
```

```{r}
p1<-plot_ordination(norm.ITS, ITS.ord, type="samples", axes=1:2, color="LOCALITY",
shape="CODE", title="samples")
p1 + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two. Faceted by sampling locality

```{r}
p1 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two. Faceted by position in the gradient

```{r}
p1 + facet_wrap(~CODE) + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three

```{r}
 #

p2<-plot_ordination(norm.ITS, ITS.ord, type="samples", axes=2:3, color="LOCALITY",
shape="CODE", title="samples")
p2 + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three. Faceted by Sampling locality

```{r}
p2 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three. Faceted by position in the gradient

```{r}
p2 + facet_wrap(~CODE) + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three

```{r}
p3<-plot_ordination(norm.ITS, ITS.ord, type="samples", axes=c(1,3), color="LOCALITY",
shape="CODE", title="samples")
p3 + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three. Faceted by LOCALITY

```{r}
p3 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the ITS OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and three. Faceted by position in the gradient

```{r}
p3 + facet_wrap(~CODE) + geom_point(size=4)
rm(p1,p2,p3)
```

##Taxonomic composition 16S

Taxonomic composition of the bacterial dataset at Class level

```{r}
# FIRST CUMULATIVE PLOT 16S
erase.me<-aggregate(abundance~Phylum,cumulative.16S,FUN=sum)
erase.me$Phylum<-
factor(erase.me$Phylum,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[1:50,],aes(x=Phylum,y=abundance,fill=Phylum)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```{r, fig.width = 12, fig.height = 5}
foo.plot<-plot_bar(norm.16S,x="Sample",y="Abundance",fill="Phylum") +
geom_bar(aes(color=Phylum, fill=Phylum), stat="identity", position="stack")
foo.plot + theme(legend.position="none")
```

```{r, fig.width = 12, fig.height = 5}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```

```{r}
# FIRST CUMULATIVE PLOT 16S
erase.me<-aggregate(abundance~Class,cumulative.16S,FUN=sum)
erase.me$Class<-
factor(erase.me$Class,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[1:50,],aes(x=Class,y=abundance,fill=Class)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

Taxonomic composition bacterial samples at Class level

```{r, fig.width = 12, fig.height = 5}
foo.plot<-plot_bar(norm.16S,x="Sample",y="Abundance",fill="Class") +
geom_bar(aes(color=Class, fill=Class), stat="identity", position="stack")
foo.plot + theme(legend.position="none")
```

```{r, fig.width = 12, fig.height = 5}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```

Taxonomic composition of the bacterial dataset at Order level

```{r}
erase.me<-aggregate(abundance~Order,cumulative.16S,FUN=sum)
erase.me$Order<-
factor(erase.me$Order,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:51,],aes(x=Order,y=abundance,fill=Order)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))

```

Taxonomic composition bacterial samples at Order level

```{r, fig.width = 12, fig.height = 5}
```

```
foo.plot<-plot_bar(norm.16S,x="Sample",y="Abundance",fill="Order") +
geom_bar(aes(color=Order, fill=Order), stat="identity", position="stack")
foo.plot + theme(legend.position="none")
```
```{r, fig.width = 15, fig.height = 15}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```

```{r}
erase.me<-aggregate(abundance~Family,cumulative.16S,FUN=sum)
erase.me$Family<-
factor(erase.me$Family,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:51,],aes(x=Family,y=abundance,fill=Family)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```{r}
erase.me<-aggregate(abundance~Genus,cumulative.16S,FUN=sum)
erase.me$Genus<-
factor(erase.me$Genus,levels=erase.me[order(erase.me[,2],decreasing=T),1])
erase.me<-erase.me[order(erase.me$abundance,decreasing=TRUE),]
#
ggplot(erase.me[2:51,],aes(x=Genus,y=abundance,fill=Genus)) + geom_col() +
theme(legend.position='none')+ theme(axis.text.x = element_text(angle = 90, hjust = 1))
```
```{r, results='asis'}
erase.me<-cbind(tax_table(phyloseq.16S),total_reads=rowSums(otu_table(phyloseq.16S)))
erase.me<-head(erase.me[order(as.numeric(erase.me[,8]),decreasing=TRUE),],40)
  stargazer(erase.me,type="html")
```
##Community structure 16S
Non-normalized
Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two
```{r}
ord.16S <- ordinate(phyloseq.16S, "NMDS", "bray",k=3)
#
p1<-plot_ordination(phyloseq.16S, ord.16S, type="taxa", axes=1:2, color="Phylum",
title="taxa")
p1 + geom_point(size=2) + theme(legend.position="none")+ facet_wrap(~Phylum)
legend <- cowplot::get_legend(p1)
grid.newpage()
grid.draw(legend)
```


Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two
```{r}
p1<-plot_ordination(phyloseq.16S, ord.16S, type="samples", axes=1:2, color="LOCALITY",
shape="CODE", title="samples")
p1 + geom_point(size=2)
```
Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two. Faceted by sampling locality
```{r}
p1 + facet_wrap(~LOCALITY) + geom_point(size=2)
```
Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two. Faceted by position in the gradient
```{r}
p1 + facet_wrap(~CODE) + geom_point(size=2)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three
```{r}
p2<-plot_ordination(phyloseq.16S, ord.16S, type="samples", axes=1:2, color="LOCALITY",
shape="CODE", title="samples")
```

```
p2 + geom_point(size=2)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by Sampling locality

```{r}
p2 + facet_wrap(~LOCALITY) + geom_point(size=2)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by position in the gradient

```{r}
p2 + facet_wrap(~CODE) + geom_point(size=2)
rm(p1,p2)
```

Normalized

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes one and two

```{r}
ord.16S <- ordinate(norm.16S, "NMDS", "bray",k=3)
#
p1<-plot_ordination(norm.16S, ord.16S, type="taxa", axes=1:2, color="Phylum", title="taxa")
p1 + geom_point(size=4) + theme(legend.position="none")+ facet_wrap(~Phylum)
```

```{r}
legend <- cowplot::get_legend(p1)
grid.newpage()
grid.draw(legend)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two

```{r}
p1<-plot_ordination(norm.16S, ord.16S, type="samples", axes=1:2, color="LOCALITY", shape="CODE", title="samples")
p1 + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two

```{r}
p1<-plot_ordination(norm.16S, ord.16S, type="samples", axes=2:3, color="LOCALITY", shape="CODE", title="samples")
p1 + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two. Faceted by sampling locality

```{r}
p1 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two. Faceted by position in the gradient

```{r}
p1 + facet_wrap(~CODE) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three

```{r}
p2<-plot_ordination(norm.16S, ord.16S, type="samples", axes=1:2, color="LOCALITY", shape="CODE", title="samples")
p2 + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by Sampling locality

```{r}
p2 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by position in the gradient

```{r}
p2 + facet_wrap(~CODE) + geom_point(size=4)
rm(p1,p2)
```

##Phylogenetic structure of the bacterial community

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) based on weighted Unifrac distances, K=2. Axes one and two

```{r, fig.width=10,fig.height=10}
#
#phy_tree(norm.16S)<-root(phy_tree(norm.16S),"46270",resolve.root=TRUE)
unifrac.16S <- ordinate(norm.16S,"MDS","UniFrac", weighted=TRUE, normalized=TRUE, parallel=TRUE, fast=TRUE)

#
p1<-plot_ordination(phyloseq.16S, unifrac.16S, type="taxa", axes=1:2, color="Phylum", title="taxa")
p1 + geom_point(size=2) + theme(legend.position="none") + facet_wrap(~Phylum)
```

```{r, fig.width=15,fig.height=15}
p1<-plot_ordination(phyloseq.16S, unifrac.16S, type="taxa", axes=1:2, color="Class", title="taxa")
p1 + geom_point(size=2) + theme(legend.position="none") + facet_wrap(~Class)
```

```{r}
legend <- cowplot::get_legend(p1)
grid.newpage()
grid.draw(legend)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two

```{r}
p1<-plot_ordination(norm.16S, unifrac.16S, type="samples", axes=1:2, color="LOCALITY", shape="CODE", title="samples")
p1 + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two

```{r}
p1 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes one and two. Faceted by position in the gradient

```{r}
p1 + facet_wrap(~CODE) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=3. Axes two and three

```{r}
p2<-plot_ordination(norm.16S, unifrac.16S, type="samples", axes=1:2, color="LOCALITY", shape="CODE", title="samples")
p2 + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by Sampling locality

```{r}
p2 + facet_wrap(~LOCALITY) + geom_point(size=4)
```

Ordination plot of the 16S OTU table. Ordination method non-metric multidimensional scaling (MetaMDS) with predefined K=2. Axes two and three. Faceted by position in the gradient

```{r}
p2 + facet_wrap(~CODE) + geom_point(size=4)
rm(p1,p2)
```

```
#Diversity plots
###ITS

Diversity ITS
```{r, fig.width = 9}
      alpha_meas = c(
            "Observed",
            "Chao1",
            "ACE",
            "Shannon",
            "Simpson",
            "InvSimpson")

      # The global plot
      p1<-plot_richness(
            phyloseq.ITS,
            "CODE",
            "LOCALITY",
            measures=alpha_meas)
      p1 + theme bw() + geom boxplot(
            data=p1$data, aes(x=CODE, y=value, color=NULL), alpha=0.1)
      #
      # Save for later
      #$
      divs.ITS<-p1$data
      #$
      divs.ITS<-
cbind(divs.ITS,reads=rowSums(otu_table(phyloseq.ITS))[divs.ITS$samples])
```

###Diversities per locality

Diversity ITS
```{r, fig.width = 9, fig.height = 10}
      p1 + theme_bw() + facet_wrap(~LOCALITY*variable,ncol=6, scales = "free") +
geom_boxplot(
            data=p1$data, aes(x=CODE, y=value, color=NULL, alpha=0.1))
```
%
%--------------------------------%
%
Diversity 16S
```{r, fig.width = 9}
      p2<-plot_richness(
            phyloseq.16S,
            "CODE",
            "LOCALITY",
            measures=alpha_meas)
      p2 + theme_bw() + geom_boxplot(
            data=p2$data, aes(x=CODE, y=value, color=NULL), alpha=0.1)
      #$
      # For later lm4 and other mixed models
      divs.16S<-p2$data
      #$
      divs.16S<-
cbind(divs.16S,reads=colSums(otu_table(phyloseq.16S))[divs.16S$samples])
      #$
```
Diversity 16S
```{r, fig.width = 9, fig.height = 10}

      p2 + theme bw() + facet wrap(~LOCALITY*variable,ncol=6, scales = "free") +
geom_boxplot(
            data=p2$data, aes(x=CODE, y=value, color=NULL, alpha=0.1))
```

##Modelling diversity
ITS
33_Div_modelling_ITS, fig.height=4, echo=FALSE, messages=FALSE, warning=FALSE,
prompt=FALSE, results = 'asis
####Test Normality
```{r, results='asis'}
#
```
```

```
#
#
# RESCALE READS
#
divs.ITS<-cbind(divs.ITS,norm.reads=((divs.ITS$reads-
min(divs.ITS$reads))/(max(divs.ITS$reads)-min(divs.ITS$reads))))
divs.16S<-cbind(divs.16S,norm.reads=((divs.16S$reads-
min(divs.16S$reads))/(max(divs.16S$reads)-min(divs.16S$reads))))
#
# Save backup
#
#save.image("/Users/ferninfm/Desktop/microbiome/datasets_Manuscript_filtered.Rdata")
#
#      Test NORMALITY
#
library(lme4)
x<-list()
for (i in levels(divs.ITS$variable))
       {
       x[[i]]<-shapiro.test(divs.ITS$value[divs.ITS$variable==i])
       }
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[3,1]
foo<-foo[1:2,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:2]
colnames(foo)<-names(x)
stargazer(title=paste("Normality test of Diversity estimates across the ITS dataset.
",tit,sep=""),foo,type="html")
```
```{r, results='asis'}
#
#
#
# RESCALE READS
#
divs.ITS<-cbind(divs.ITS,norm.reads=((divs.ITS$reads-
min(divs.ITS$reads))/(max(divs.ITS$reads)-min(divs.ITS$reads))))
divs.16S<-cbind(divs.16S,norm.reads=((divs.16S$reads-
min(divs.16S$reads))/(max(divs.16S$reads)-min(divs.16S$reads))))
#
# Save backup
#
#save.image("/Users/ferninfm/Desktop/microbiome/datasets_Manuscript_filtered.Rdata")
#
#      Test NORMALITY
#
library(lme4)
x<-list()
for (i in levels(divs.ITS$variable))
       {
       x[[i]]<-shapiro.test(log10(divs.ITS$value[divs.ITS$variable==i]))
       }
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[3,1]
foo<-foo[1:2,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:2]
colnames(foo)<-names(x)
stargazer(title=paste("Normality test of the logarithm of diversity estimates across the
ITS dataset. ",tit,sep=""),foo,type="html")
```
####TEST HOMOCEDASTICITY (One way rank anova)
```{r, results='asis'}
#
x<-list()
for (i in levels(divs.ITS$variable))
       {
       x[[i]]<-
kruskal.test(divs.ITS$value[divs.ITS$variable==i]~as.numeric(factor(divs.ITS$CODE[divs.I
TS$variable==i])))
       }
```

```
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[4,1]
foo<-foo[1:3,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:3]
colnames(foo)<-names(x)
stargazer(title=paste("Heterocedasticity in Diversity estimates across the ITS dataset.
",tit,".",sep=""),foo[c(2,1,3),],type="html")
```

Heterocedasticity Bartlett test
```{r, results='asis'}
for (i in levels(divs.ITS$variable))
       {
       x[[i]]<-
bartlett.test(divs.ITS$value[divs.ITS$variable==i],divs.ITS$CODE[divs.ITS$variable==i])
       }
#
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[5,1]
foo<-foo[1:3,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:3]
colnames(foo)<-names(x)
stargazer(title=paste("Heterocedasticity in Diversity estimates across the ITS dataset.
",tit,".",sep=""),foo[c(2,1,3),],type="html")
```

PAIRWISE DIFFERENCE OF MEANS
```{r, results='asis'}
x<-list()
for (i in levels(divs.ITS$variable))
       {
       x[[i]]<-
pairwise.wilcox.test(divs.ITS$value[divs.ITS$variable==i],divs.ITS$CODE[divs.ITS$variabl
e==i])
       }
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[1,1]
foo<-foo[c(3,4,6),]
foo<-substr(foo,1,6)
rownames(foo)<-c("KM","KT","MT")
colnames(foo)<-names(x)
stargazer(title=paste("Difference in statistical distribution Richness and diversity
estimates in the ITS dataset among positions in the snowbed slope gradient. ",tit,".
Table values represent the Holm adjusted probability that diversity and richness
estimates for K,M,T positions across gradients belong to the same
distribution.",sep=""),foo,type="html")
```

Modelling Richness and diversity with read depth as a covariate and LOCALITY as a random effect
```{r, results='asis'}

mods<-list()
for (i in levels(divs.ITS$variable))
       {
       mods[[i]]<-lmer(value~as.numeric(factor(CODE))+norm.reads+(1|LOCALITY),
       data=divs.ITS[divs.ITS$variable==i,],REML=FALSE)
       }

stargazer(mods,title="Influence of the position in the Slope gradient on Richness and
Diversity metrics for the ITS dataset. Linear mixed-effects models accounting for
individual localities as blocks and sampling intensity as
covariate",dep.var.labels=names(mods),covariate.labels=c("Slope","Intensity",
"Constant"),type="html")
```

```{r, results='asis'}
nmods<-list()
```

```
for (i in levels(divs.ITS$variable))
       {
       nmods[[i]]<-lmer(value~norm.reads+(1|LOCALITY),
       data=divs.ITS[divs.ITS$variable==i,],REML=FALSE)
       }

stargazer(nmods,title="Influence of the position in the Slope gradient on Richness and
Diversity metrics for the ITS dataset. Summary of Null models accounting for localities
as blocks and sampling intensity as a
covariate",dep.var.labels=names(mods),covariate.labels=c("Intensity",
"Constant"),type="html")
```

```{r, results='asis'}
anomods<-mods
for (i in names(anomods)) anomods[[i]]<-anova(mods[[i]],nmods[[i]])
names(anomods)<-names(mods)
#
#
#
foo<-unlist(lapply(anomods,as.matrix))
dim(foo)<-c(length(foo)/6,6)
foo<-t(foo)
foo1<-mat.or.vec(12,8)
for (i in 1:6) foo1[((i*2)-1):(i*2),]<-rbind(foo[i,((1:8*2)-1)],foo[i,((1:8*2))])
colnames(foo1)<-names(anomods[[1]])
rownames(foo1)<-
c(names(anomods),paste(names(anomods),".null",sep=""))[c(7,1,8,2,9,3,10,4,11,5,12,6)]
stargazer(title="Support for influence of the position in the gradient on diversity
metrics for the ITS dataset.Summary of ANOVA camparisons between null and complete
models.",foo1,type="html")
rm(foo,foo1)
```
16S

Test NORMALITY
34_Div_modelling_16S, fig.height=4, echo=FALSE, warning=FALSE, messages=FALSE,
prompt=FALSE,results='asis'
```{r, results='asis'}
#
x<-list()
for (i in levels(divs.16S$variable))
       {
       x[[i]]<-shapiro.test(divs.16S$value[divs.16S$variable==i])
       }
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[3,1]
foo<-foo[1:2,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:2]
colnames(foo)<-names(x)
stargazer(title=paste("Normality test of Diversity estimates across the 16S dataset.
",tit,sep=""),foo,type="html")
```

TEST HOMOCEDASTICITY (One way rank anova)
```{r, results='asis'}
x<-list()
for (i in levels(divs.16S$variable))
       {
       x[[i]]<-
kruskal.test(divs.16S$value[divs.16S$variable==i]~as.numeric(factor(divs.16S$CODE[divs.1
6S$variable==i])))
       }

foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[4,1]
foo<-foo[1:3,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:3]
colnames(foo)<-names(x)
```

```
stargazer(title=paste("Heterocedasticity in Diversity estimates across the 16S dataset.
",tit,".",sep=""),foo[c(2,1,3),],type="html")
```
Bartlett's test
```{r, results='asis'}
for (i in levels(divs.16S$variable))
       {
       x[[i]]<-
bartlett.test(divs.16S$value[divs.16S$variable==i],divs.16S$CODE[divs.16S$variable==i])
       }
#
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[5,1]
foo<-foo[1:3,]
foo<-substr(foo,1,6)
rownames(foo)<-names(x[[1]])[1:3]
colnames(foo)<-names(x)
stargazer(title=paste("Heterocedasticity in Diversity estimates across the 16S dataset.
",tit,".",sep=""),foo[c(2,1,3),],type="html")
```
PAIRWISE DIFFERENCE OF MEANS
```{r, results='asis'}
x<-list()
for (i in levels(divs.16S$variable))
       {
       x[[i]]<-
pairwise.wilcox.test(divs.16S$value[divs.16S$variable==i],divs.16S$CODE[divs.16S$variabl
e==i])
       }
foo<-unlist(lapply(x,as.matrix))
dim(foo)<-c(length(foo)/6,6)
tit<-foo[1,1]
foo<-foo[c(3,4,6),]
foo<-substr(foo,1,6)
rownames(foo)<-c("KM","KT","MT")
colnames(foo)<-names(x)
stargazer(title=paste("Difference in statistical distribution Richness and diversity
estimates in the 16S dataset among positions in the snowbed slope gradient. ",tit,".
Table values represent the Holm adjusted probability that diversity and richness
estimates for K,M,T positions across gradients belong to the same
distribution.",sep=""),foo,type="html")
```


Modelling Richness and diversity with read depth as a covariate and LOCALITY as a
random effect
```{r, results='asis'}
mods<-list()
for (i in levels(divs.16S$variable))
       {
       mods[[i]]<-lmer(value~as.numeric(factor(CODE))+norm.reads+(1|LOCALITY),
       data=divs.16S[divs.16S$variable==i,],REML=FALSE)
       }
stargazer(mods,title="Influence of the position in the Slope gradient on Richness and
Diversity metrics for the 16S dataset.  Linear mixed-effects models accounting for
individual localities as blocks and sampling intensity as
covariate",dep.var.labels=names(mods),covariate.labels=c("Slope","Intensity",
"Constant"),type="html")
```
Uno
```{r, results='asis'}
nmods<-list()
for (i in levels(divs.16S$variable))
       {
       nmods[[i]]<-lmer(value~norm.reads+(1|LOCALITY),
       data=divs.16S[divs.16S$variable==i,],REML=FALSE)
       }

stargazer(nmods,title="Influence of the position in the Slope gradient on Richness and
Diversity metrics for the 16S dataset. Summary of Null models accounting for localities
as blocks and sampling intensity as a
```

```
covariate",dep.var.labels=names(mods),covariate.labels=c("Intensity",
"Constant"),type="html")
```
Dos
```{r, results='asis'}
anomods<-mods
for (i in names(anomods)) anomods[[i]]<-anova(mods[[i]],nmods[[i]])
names(anomods)<-names(mods)
#
#
#
foo<-unlist(lapply(anomods,as.matrix))
dim(foo)<-c(length(foo)/6,6)
foo<-t(foo)
foo1<-mat.or.vec(12,8)
for (i in 1:6) foo1[((i*2)-1):(i*2),]<-rbind(foo[i,((1:8*2)-1)],foo[i,((1:8*2))])
colnames(foo1)<-names(anomods[[1]])
rownames(foo1)<-
c(names(anomods),paste(names(anomods),".null",sep=""))[c(7,1,8,2,9,3,10,4,11,5,12,6)]
stargazer(title="Support for influence of the position in the gradient on diversity
metrics for the 16S dataset.Summary of ANOVA camparisons between null and complete
models.",foo1,type="html")
rm(foo,foo1)
```
Modelling Taxonomic composition using a Differencial Gene expression framework
ITS
Fold diferences in sequencing proportion between locations in the gradient at OTU level grouped by genus. Only OTUs with a variance above 1e-5 are shown.
```{r}
##############################################################################
#' Convert phyloseq OTU count data into DGEList for edgeR package
#'
#' Further details.
#'
#' @param physeq (Required).  A \code{\link{phyloseq-class}} or
#'  an \code{\link{otu_table-class}} object.
#'  The latter is only appropriate if \code{group} argument is also a
#'  vector or factor with length equal to \code{nsamples(physeq)}.
#'
#' @param group (Required). A character vector or factor giving the experimental
#'  group/condition for each sample/library. Alternatively, you may provide
#'  the name of a sample variable. This name should be among the output of
#'  \code{sample_variables(physeq)}, in which case
#'  \code{get_variable(physeq, group)} would return either a character vector or factor.
#'  This is passed on to \code{\link[edgeR]{DGEList}},
#'  and you may find further details or examples in its documentation.
#'
#' @param method (Optional). The label of the edgeR-implemented normalization to use.
#'  See \code{\link[edgeR]{calcNormFactors}} for supported options and details.
#'  The default option is \code{"RLE"}, which is a scaling factor method
#'  proposed by Anders and Huber (2010).
#'  At time of writing, the \link[edgeR]{edgeR} package supported
#'  the following options to the \code{method} argument:
#'
#'  \code{c("TMM", "RLE", "upperquartile", "none")}.
#'
#' @param ... Additional arguments passed on to \code{\link[edgeR]{DGEList}}
#'
#' @examples
#'
phyloseq_to_edgeR = function(physeq, group, method="RLE", ...){
  require("edgeR")
  require("phyloseq")
  # Enforce orientation.
  if( !taxa_are_rows(physeq) ){ physeq <- t(physeq) }
  x = as(otu_table(physeq), "matrix")
  # Add one to protect against overflow, log(0) issues.
  x = x + 1
  # Check `group` argument
  if( identical(all.equal(length(group), 1), TRUE) & nsamples(physeq) > 1 ){
    # Assume that group was a sample variable name (must be categorical)
    group = get_variable(physeq, group)
  }
  # Define gene annotations (`genes`) as tax_table
```

```
  taxonomy = tax_table(physeq, errorIfNULL=FALSE)
  if( !is.null(taxonomy) ){
    taxonomy = data.frame(as(taxonomy, "matrix"))
  }
  # Now turn into a DGEList
  y = DGEList(counts=x, group=group, genes=taxonomy, remove.zeros = TRUE, ...)
  # Calculate the normalization factors
  z = calcNormFactors(y, method=method)
  # Check for division by zero inside `calcNormFactors`
  if( !all(is.finite(z$samples$norm.factors)) ){
    stop("Something wrong with edgeR::calcNormFactors on this data,
         non-finite $norm.factors, consider changing `method` argument")
  }
  # Estimate dispersions
  return(estimateTagwiseDisp(estimateCommonDisp(z)))
}
##############################################################################
#
# Differential expression
#
#
#
#
#norm.ITS <- transform_sample_counts(phyloseq.ITS, function(x){x/sum(x)})
plot(density((log10(apply(otu_table(phyloseq.ITS), 2, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very high variance")
```
ITS_ genus
```{r, results='asis'}
culo<-phyloseq.ITS
#
# Prune Taxa
#
varianceThreshold = 1e5
keepOTUs <- names(which(apply(otu_table(culo), 2, var) > varianceThreshold))
culo = prune_taxa(keepOTUs, culo)

#
dge<-phyloseq_to_edgeR(culo, group="CODE")
dispersions<-estimateDisp(dge)

#
#
et <- exactTest(dge,pair=c("K","T"))
#

# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.05
sigtab <- res[(res$FDR < alpha), ]
#
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))
#
sigtab <- cbind(as(sigtab, "data.frame"), as(tax_table(culo)[rownames(sigtab), ],
"matrix"))
#
```
```{r}
#
#
theme_set(theme_bw())
scale_fill_discrete <- function(palname = "Set1", ...) {
    scale_fill_brewer(palette = palname, ...)
}
sigtabgen = subset(sigtab, Order!="unidentified"&Genus!="unidentified")
# ORDER order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Genus, function(x) max(x))
```

```
x = sort(x, TRUE)
sigtabgen$Genus = factor(as.character(sigtabgen$Genus), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Genus, y = logFC, color = Order)) +
geom_point(size=4) +
  theme(axis.text.x = element text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```
```{r}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```
Fold diferences in sequencing proportion between locations in the gradient with OTUs
aggregated at Order level. Only Orders with a variance above 1e-5 are used.'
```{r}
##############################################################################
#
# Differential expression
#
#
#
#
culo<-tax_glom(phyloseq.ITS,"Order")
plot(density((log10(apply(otu_table( culo), 2, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very low variance")
```
```{r, results='asis'}
#
# Prune Taxa
#
varianceThreshold = 1e4
keepOTUs <- names(which(apply(otu_table( culo), 2, var) > varianceThreshold))
 culo = prune taxa(keepOTUs,  culo)

#
dge<-phyloseq_to_edgeR( culo, group="CODE")
dispersions<-estimateDisp(dge)
#
#
et <- exactTest(dge,pair=c("K","T"))
#
# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.05
sigtab <- res[(res$FDR < alpha), ]
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))
sigtab <- cbind(as(sigtab, "data.frame"), as(tax_table( culo)[rownames(sigtab), ],
"matrix"))
```
```{r}
#
#
#
theme_set(theme_bw())
sigtabgen = subset(sigtab, Order!="unidentified")
# ORDER class
x = tapply(sigtabgen$logFC, sigtabgen$Class, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Class = factor(as.character(sigtabgen$Class), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Order, y = logFC, color =  Class)) +
  geom_point(size=4) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```
```{r}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
```

```
grid.draw(legend)
```
Fold diferences in sequencing proportion between locations in the gradient with OTUs aggregated at Genus level. Only Genera with a variance above 1e-5 are used.
```{r}
# Differential expression
#

#
#
#
culo<-tax_glom(phyloseq.ITS,"Genus")
plot(density((log10(apply(otu_table( culo), 2, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very low variance")
```

```{r, results='asis'}
#
# Prune Taxa
#
varianceThreshold = 1e4
keepOTUs <- names(which(apply(otu_table( culo), 2, var) > varianceThreshold))
 culo = prune_taxa(keepOTUs,  culo)

#
dge<-phyloseq_to_edgeR( culo, group="CODE")
dispersions<-estimateDisp(dge)

#
#
et <- exactTest(dge,pair=c("K","T"))
# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.001
sigtab <- res[(res$FDR < alpha), ]
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))
#
sigtab <- cbind(as(sigtab, "data.frame"), as(tax_table( culo)[rownames(sigtab), ],
"matrix"))
```
```{r}
#
#
theme_set(theme_bw())
sigtabgen = subset(sigtab, Order!="unidentified"&Genus!="unidentified")
# ORDER Order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Genus = factor(as.character(sigtabgen$Genus), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Genus, y = logFC, color =  Class)) +
  geom_point(size=4) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```
```{r}
legend <- cowplot::get legend(foo.plot)
grid.newpage()
grid.draw(legend)
```


Modelling Taxonomic composition using a Differencial Gene expression framework

16 S
Fold diferences in sequencing proportion between locations in the gradient at OTU level grouped by genus. Only OTUs with a variance above 1e-7.5 are shown. 16S dataset

```{r}
plot(density((log10(apply(otu_table(phyloseq.16S), 1, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very low variance")
```

```{r, results='asis'}
culo<-phyloseq.16S
#
# Prune Taxa
#
varianceThreshold = 1e4
keepOTUs <- names(which(apply(otu_table(culo), 1, var) > varianceThreshold))
culo = prune_taxa(keepOTUs, culo)

#
dge<-phyloseq_to_edgeR(culo, group="CODE")
dispersions<-estimateDisp(dge)
#
#
#
et <- exactTest(dge,pair=c("K","T"))
#


#
# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.001
sigtab <- res[(res$FDR < alpha), ]
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))
sigtab <- cbind(as(sigtab, "data.frame"), as(tax table(culo)[rownames(sigtab), ],
"matrix"))
#
```
```{r}
#
#
theme_set(theme_bw())
sigtabgen = subset(sigtab, Order!="unidentified"&Genus!="unidentified")
# ORDER order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Genus = factor(as.character(sigtabgen$Genus), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Genus, y = logFC, color = Order)) +
geom_point(size=3) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```

```{r}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```

Fold diferences in sequencing proportion between locations in the gradient with OTUs aggregated at Order level. Only Orders with a variance above 1e-7.5 are used. 16S dataset

```{r}
tax_glom_16S_order<-tax_glom(phyloseq.16S,"Order")
culo<-tax_glom_16S_order
plot(density((log10(apply(otu_table( culo), 1, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very low variance")
```

```{r, results='asis'}
#
# Prune Taxa
#
varianceThreshold = 1e-6
keepOTUs <- names(which(apply(otu_table( culo), 1, var) > varianceThreshold))
 culo = prune_taxa(keepOTUs,  culo)

#
dge<-phyloseq_to_edgeR( culo, group="CODE")
#
#
dispersions<-estimateDisp(dge)
et <- exactTest(dge,pair=c(1,3))

# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.001
sigtab <- res[(res$FDR < alpha), ]
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))

sigtab <- cbind(as(sigtab, "data.frame"), as(tax_table( culo)[rownames(sigtab), ],
"matrix"))
```
```{r}
#
#
theme_set(theme_bw())
sigtabgen = subset(sigtab, Order!="unidentified")
# ORDER class
x = tapply(sigtabgen$logFC, sigtabgen$Class, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Class = factor(as.character(sigtabgen$Class), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Order, y = logFC, color =  Class)) +
geom_point(size=3) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```

```{r}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```
Fold diferences in sequencing proportion between locations in the gradient with OTUs aggregated at Genus level. Only Genera with a variance above 1e-7.5 are used.
```{r}
tax_glom_16S_genus<-tax_glom(phyloseq.16S,"Genus")
culo<-tax_glom_16S_genus
plot(density((log10(apply(otu_table( culo), 1, var)))),
     xlab="log10(variance)",
     main="A large fraction of OTUs have very low variance")
```

```{r, results='asis'}
# Prune Taxa
#
varianceThreshold = 1e-7
keepOTUs <- names(which(apply(otu_table( culo), 1, var) > varianceThreshold))
 culo = prune_taxa(keepOTUs,  culo)

#
dge<-phyloseq_to_edgeR( culo, group="CODE")
dispersions<-estimateDisp(dge)
#
#
et <- exactTest(dge,pair=c(1,3))
```

```
# Extract values from test results
tt <- topTags(et, n=nrow(dge$table), adjust.method="BH", sort.by="PValue")
res <- tt@.Data[[1]]
alpha <- 0.001
sigtab <- res[(res$FDR < alpha), ]
stargazer(as.matrix(sigtab),type="html",title=paste("Common dispersion
=",dispersions$common.dispersion))

sigtab <- cbind(as(sigtab, "data.frame"), as(tax_table( culo)[rownames(sigtab), ],
"matrix"))
```
```{r}
#
#
#
theme_set(theme_bw())
sigtabgen = subset(sigtab, Order!="unidentified"&Genus!="unidentified")
# ORDER Order
x = tapply(sigtabgen$logFC, sigtabgen$Order, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Order = factor(as.character(sigtabgen$Order), levels = names(x))
# Genus order
x = tapply(sigtabgen$logFC, sigtabgen$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Genus = factor(as.character(sigtabgen$Genus), levels = names(x))
foo.plot<-ggplot(sigtabgen, aes(x = Genus, y = logFC, color =  Class)) +
geom_point(size=3) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust = 0.5))
foo.plot + theme(legend.position="none")
```
```{r}
legend <- cowplot::get_legend(foo.plot)
grid.newpage()
grid.draw(legend)
```

Appendix 1. ITS
```{r, results='asis'}
erase.me<-cbind(tax_table(phyloseq.ITS),total_reads=colSums(otu_table(phyloseq.ITS)))
erase.me<-erase.me[order(as.numeric(erase.me[,8]),decreasing=TRUE),]
stargazer(erase.me,type="text")
```
Appendix 2. 16S
```{r, results='asis'}
erase.me<-cbind(tax_table(phyloseq.16S),total_reads=rowSums(otu_table(phyloseq.16S)))
erase.me<-erase.me[order(as.numeric(erase.me[,8]),decreasing=TRUE),]
stargazer(erase.me,type="text")
```

Medeninhaber und Herausgeber, Verleger:
Nationalparkrat Hohe Tauern
Kirchplatz 2, 9971 Matrei
Tel.: +43 (0) 4875 / 5112 | E-Mail: nationalparkrat@hohetauern.at

www.hohetauern.at