

# Metagenomics of Parkinson’s disease implicates the gut microbiome in multiple disease mechanisms

Zachary D Wallen      Ayse Demirkan      Guy Twa      Gwendolyn Cohen  
Marissa N Dean      David G Standaert      Timothy Sampson      Haydeh Payami

## Contents

<b>Introduction</b>	<b>2</b>
<b>Bioinformatic processing of sequences</b>	<b>2</b>
Adapter trimming and quality trimming/filtering of sequences using BBDuk . . . . .	2
Removal of human host sequence reads using BBSplit/BBMap . . . . .	3
Removal of low complexity sequences using BBDuk . . . . .	3
Post quality control exclusions . . . . .	4
Taxonomic and functional profiling using MetaPhlan3 and HUMAnN3 . . . . .	4
<b>Setting up R environment for statistical analyses</b>	<b>6</b>
Create needed functions for analyses . . . . .	6
Load required R packages . . . . .	12
Create excel styles used for formatting output . . . . .	13
Make directories for output . . . . .	13
<b>Subject characteristics: testing PD vs NHC</b>	<b>13</b>
<b>Analyses of species and genera</b>	<b>55</b>
Preparing relative abundance and count data for downstream analyses . . . . .	55
Principal Component Analysis . . . . .	57
PERMANOVA and PERMDISP . . . . .	58
Enterotype analysis: PD vs NHC . . . . .	60
Differential abundance of species and genera . . . . .	64
MWAS . . . . .	64
MaAsLin2 and ANCOM-BC MWAS concordance . . . . .	71
Genus heterogeneity . . . . .	73
Species distributions and fold changes from MWAS . . . . .	75
Sex, age, and confounder analysis . . . . .	79
Correlation networks . . . . .	83
SparCC . . . . .	83
Community detection and creating node and edge files for network visualization . . . . .	86
<b>Analyses of gene families and pathways</b>	<b>89</b>
Preparing count data for downstream analyses . . . . .	89
Differential abundance of gene families and pathways . . . . .	89
MWAS . . . . .	89
Sporulation KO group association with constipation . . . . .	94
Gene family/pathway boxplots with fold changes from MWAS . . . . .	97

<b>Secondary analyses (replication driven)</b>	<b>116</b>
Replicating signal for SILVA v 132 Prevotella . . . . .	116
Replicating signal for opportunitstic pathogen cluster . . . . .	117
<b>R session information</b>	<b>119</b>

## Introduction

This file documents the workflow used for bioinformatics and code used to perform the statistical analyses detailed in the manuscript *Metagenomics of Parkinson’s disease implicates the gut microbiome in multiple disease mechanisms*. As a brief background, fecal samples from 492 PD and 242 control subjects were sent for shotgun metagenomic sequencing. Shotgun metagenomic sequences were acquired for all samples, quality controlled, decontaminated for human sequences, and filtered for low complexity sequences. Of these samples, 724 (490 PD and 234 controls) were taxonomically profiled using MetaPhlan3, functionally profiled for gene families and pathways using HUMAnN3, and included in statistical analyses where subject meta-data is taken into account. Taxonomic profiling resulted in species relative abundances (i.e., the % each species makes up out of all species that were detected in a sample) and species counts (i.e., estimation of how many times a species was observed in a sample; calculated by multiplying the relative abundances by the total sequence count for each sample), and functional profiling resulted in gene family and pathway counts, which were used in the statistical analyses detailed below. Raw shotgun metagenomic sequences were uploaded to NCBI SRA where they were decontaminated for human sequences by SRA. Sequences can be accessed under BioProject [PRJNA834801](https://www.ncbi.nlm.nih.gov/bioproject/PRJNA834801).

## Bioinformatic processing of sequences

Shotgun sequences were bioinformatically processed from raw sequences to taxonomic and functional (gene families and pathway) profiles using a pipeline that involved sequence QC, decontamination, and low complexity sequence filtering with [BBDuk](#) and [BBSplit](#), and taxonomic/functional profiling with [MetaPhlan3](#) and [HUMAnN3](#). Below describes the steps of the pipeline in brief, along with the shell code used to perform each step.

### Adapter trimming and quality trimming/filtering of sequences using BBDuk

Adapter (and PhiX) sequences were removed and quality trimming/filtering of sequence reads was performed using BBDuk with the following parameters:

- **ref=adapters,phix**: For removing common sequencing adapters and PhiX sequences.
- **ftm=5**: For trimming sequences to a length that is multiple of 5 (helps in removing extra base if one exists (i.e. if length of reads is 151 bp instead of 150 bp)).
- **tbo**: In addition to usual kmer adapter trimming, specifies to also trim adapters based on pair overlap detection using BBMerge.
- **tpe**: Specifies to make sure and trim both reads to the same length.
- **qtrim=r1**: Quality trim both 5’ and 3’ end of sequences.
- **trimq=25** : Quality score to trim up to on sequence ends.
- **minlen=50**: Filter out sequences that fall below 50 bp in length.

The script used to carry out the task above in the HPC environment is shown:

```

1 # perform adapter trimming and quality trimming/filtering for each sample
2 bbdduk.sh \
3 in=${FILE_NAME}_R1_001.fastq.gz \
4 in2=${FILE_NAME}_R2_001.fastq.gz \

```

```

5 out=Quality_Controlled_Sequences/${FILE_NAME}_R1_001.fastq.gz \
6 out2=Quality_Controlled_Sequences/${FILE_NAME}_R2_001.fastq.gz \
7 stats=Quality_Controlled_Sequences/${FILE_NAME}_stats.txt \
8 ftm=5 tpe tbo qtrim=r1 trimq=25 minlen=50 ref=adapters,phix \
9 -Xmx${MAX_MEM}g

```

## Removal of human host sequence reads using BBSplit/BBMap

Human sequence reads were removed from each sequence file by aligning reads to the most recent human genome reference using BBSplit/BBMap.

- The most recent human genome reference (GCA\_000001405.28\_GRCh38.p13\_genomic.fna) was downloaded from the [NCBI FTP site](#).
- BBSplit was ran with default parameters for both indexing the human genome reference file and for mapping sequences.

The script used to carry out the task above in the HPC environment is shown:

```

1 # index reference genome file
2 bbsplit.sh \
3 ref=GCA_000001405.28_GRCh38.p13_genomic.fna \
4 path=Decontaminated_Sequences \
5 t=${CPU_REQUEST} \
6 -Xmx${MAX_MEM}g
7
8 # perform decontamination for each sample
9 bbsplit.sh \
10 in1=Quality_Controlled_Sequences/${FILE_NAME}_R1_001.fastq.gz \
11 in2=Quality_Controlled_Sequences/${FILE_NAME}_R2_001.fastq.gz \
12 outu1=Decontaminated_Sequences/${FILE_NAME}_R1_001.fastq.gz \
13 outu2=Decontaminated_Sequences/${FILE_NAME}_R2_001.fastq.gz \
14 path=Decontaminated_Sequences \
15 basename=Decontaminated_Sequences/${FILE_NAME}._contam_#.fastq.gz \
16 refstats=Decontaminated_Sequences/${FILE_NAME}_refstats.log \
17 t=${CPU_REQUEST} \
18 -Xmx${MAX_MEM}g

```

## Removal of low complexity sequences using BBDuk

To remove low complexity sequences such as mononucleotide repeats, BBDuk entropy filtering was ran with following parameters:

- `entropy=0.01`: Entropy threshold for removing low complexity sequences. This value suggested by BBDuk author to remove only monomeric repeats (<http://seqanswers.com/forums/showthread.php?t=42776&page=7>).
- `entropywindow=50`: Calculate entropy using a sliding window of this length. Value is the default.
- `entropyk=5`: Calculate entropy using kmers of this length. Value is the default.

The script used to carry out the task above in the HPC environment is shown:

```

1 # concatenate paired sequence files for each sample
2 zcat Decontaminated_Sequences/${FILE_NAME}_R1_001.fastq.gz \
3 Decontaminated_Sequences/${FILE_NAME}_R2_001.fastq.gz \
4 > ${FILE_NAME}.fastq
5

```

```

6 # perform entropy filtering on concatenated paired sequence files
7 bbduk.sh \
8 in=${FILE_NAME}.fastq \
9 out=Low_Complexity_Filtered_Sequences/${FILE_NAME}.fastq.gz \
10 outm=Low_Complexity_Filtered_Sequences/Removed_Sequences/${FILE_NAME}.fastq.gz \
11 entropy=0.01 \
12 entropywindow=50 \
13 entropyk=5 \
14 -Xmx${MAX_MEM}g

```

## Post quality control exclusions

- Duplicate PD sample, N=1 (not uploaded to SRA)
- NHC samples whose subjects reported to have a neurological condition, N=8 (not uploaded to SRA)
- PD sample with high human DNA contamination and low sequence count (< 10M) after QC, N=1.
  - Note: sequences for this sample were made available on SRA repository
- This made the final set of quality controlled sequences range from 12M - 285M sequences per sample for 490 PD and 234 NHC.

## Taxonomic and functional profiling using MetaPhlAn3 and HUMAnN3

Quality controlled, decontaminated, and low complexity filtered sequences were profiled for taxonomic and functional content using MetaPhlAn v 3.0.14 that performs marker gene based taxonomic profiling to get relative abundances of microbial clades present in each sample, then using HUMAnN v 3.0.0 to determine the microbial gene family and metabolic pathway content present in each sample using the UniRef and MetaCyc databases.

- Taxonomic profiling with MetaPhlAn was performed once with default parameters, and then a second time adding the `--unknown_estimation` flag. This flag was enabled so the relative abundances of clades would take the unknown content of a samples metagenome (portion of sequenced metagenome that is not contained in the MetaPhlAn database) into account (important for calculating count data). When the second run of MetaPhlAn was complete, estimated counts for each clade were derived by multiplying the relative abundances with unknown estimation by the total read count reported by the bowtie2 intermediate file from running MetaPhlAn (formula:  $(\text{relative abundance} / 100) \times \text{nread from bowtie2}$ ).

The script used to carry out the task above in the HPC environment is shown:

```

1 # perform taxonomic profiling with default parameters for each sample
2 metaphlan \
3 Low_Complexity_Filtered_Sequences/${FILE_NAME}.fastq.gz \
4 --input_type fastq \
5 -t rel_ab \
6 --nproc $CPU_REQUEST \
7 --bowtie2out Taxonomic_Profiling/${FILE_NAME}_metaphlan_bowtie2.txt \
8 -o Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab.tsv
9
10 # perform taxonomic profiling adding unknown estimation for each sample
11 metaphlan \
12 Low_Complexity_Filtered_Sequences/${FILE_NAME}.fastq.gz \
13 --input_type fastq \
14 -t rel_ab \
15 --unknown_estimation \
16 --nproc $CPU_REQUEST \

```

```

17 --bowtie2out Taxonomic_Profiling/${FILE_NAME}_metaphlan_bowtie2.txt \
18 -o Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab_w_unknown.tsv
19
20 # calculate count data using relative abundances with unknown estimation
21 NREADS=$(grep '#nreads' Taxonomic_Profiling/${FILE_NAME}_metaphlan_bowtie2.txt | \
22     awk '{print $2}')
23
24 grep -v '^#' Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab_w_unknown.tsv | \
25 awk -v nreads="$NREADS" '{OFMT="%f";print $1,$2,$3,($3/100)*nreads,$4}' OFS='\t' | \
26 cat <(grep '^#' Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab_w_unknown.tsv) - | \
27 sed 's/#clade_name\tNCBI_tax_id\trelative_abundance\tadditional_species/
28 #clade_name\tNCBI_tax_id\trelative_abundance\tcounts\tadditional_species/' \
29 > Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab_w_counts.tsv
30
31 # extract count data
32 awk -F'\t' '{print $1,$2,$4}' OFS='\t' \
33 Taxonomic_Profiling/${FILE_NAME}_metaphlan_rel_ab_w_counts.tsv \
34 > Taxonomic_Profiling/${FILE_NAME}_metaphlan_counts.tsv
35
36 # merge individual sample files
37 merge_metaphlan_tables.py \
38 Taxonomic_Profiling/*_metaphlan_rel_ab.tsv > Taxonomic_Profiling/metaphlan_rel_ab.tsv
39 sed -i '2s/_metaphlan_rel_ab//g' Taxonomic_Profiling/metaphlan_rel_ab.tsv
40
41 merge_metaphlan_tables.py \
42 Taxonomic_Profiling/*_metaphlan_counts.tsv > Taxonomic_Profiling/metaphlan_counts.tsv
43 sed -i '2s/_metaphlan_counts//g' Taxonomic_Profiling/metaphlan_counts.tsv

```

- Functional profiling with HUMAnN was performed with default parameters using the ChocoPhlAn database full\_chocophlan.v296\_201901b and UniRef database uniref90\_annotated\_v201901b\_full with sequences grouped at 90% identity.

```

1 # perform functional profiling with default parameters for each sample
2 humann \
3 --input Low_Complexity_Filtered_Sequences/${FILE_NAME}.fastq.gz \
4 --output Functional_Profiling \
5 --output-basename $FILE_NAME \
6 --nucleotide-database full_chocophlan.v296_201901b \
7 --protein-database uniref90_annotated_v201901b_full \
8 --metaphlan-options '-t rel_ab' \
9 --prescreen-threshold 0.01 \
10 --threads $CPU_REQUEST \
11 --verbose
12
13 # merge per sample tables by code provided by developers
14 humann_join_tables \
15 --input Functional_Profiling/ \
16 --output Functional_Profiling/humann_genefamilies.tsv \
17 --file_name genefamilies.tsv
18
19 humann_join_tables \
20 --input Functional_Profiling/ \
21 --output Functional_Profiling/humann_pathabundance.tsv \
22 --file_name pathabundance.tsv

```

- To reduce the number of gene families being analyzed, default HUMAnN gene families (UniRef90) were converted to KEGG ortholog (KO) groups using the `humann_regroup_table` script packaged with HUMAnN3 specifying the mapping file to be `map_ko_uniref90.txt.gz`. This was performed on gene family tables for each sample.
  - Note: on average per sample, 4-7% of gene families were able to be regrouped into KO groups
  - Per sample KO group abundances were then merged to create one file for all samples

```

1 # convert UniRef90 gene families to KO groups for each sample
2 humann_regroup_table \
3 -i Functional_Profiling/${FILE_NAME}_genefamilies.tsv \
4 -c full_mapping_v201901b/map_ko_uniref90.txt.gz \
5 -o Functional_Profiling/${FILE_NAME}_humann_KO_group_counts.tsv
6
7 # merge per sample tables
8 humann_join_tables \
9 --input Functional_Profiling/ \
10 --output Functional_Profiling/humann_KO_group_counts.tsv \
11 --file_name humann_KO_group_counts.tsv

```

- Pathway and KO group abundance tables were outputted in stratified format (contains abundances broken up by contributing species), therefore, pathway and KO group abundance files were subsetted for only the community level abundances using the following commands:

```

1 # extract only community level abundances
2 grep -v '|' Functional_Profiling/humann_pathabundance.tsv \
3 > Functional_Profiling/humann_pathabundance.tsv
4 grep -v '|' Functional_Profiling/humann_KO_group_counts.tsv \
5 > Functional_Profiling/humann_KO_group_counts.tsv

```

- Then, more informative names were given to KO groups using the `humann_rename_table` script packaged with HUMAnN3 specifying the naming file to be `map_ko_name.txt.gz`.

```

1 # add KO group names to file
2 humann_rename_table \
3 -i Functional_Profiling/humann_KO_group_counts.tsv \
4 -c full_mapping_v201901b/map_ko_name.txt.gz \
5 -o Functional_Profiling/humann_KO_group_counts.tsv

```

- The above taxonomic and functional profiling resulted in 4 tables: microbial clade relative abundances and estimated counts, KO group RPK abundances, and pathway RPK abundances. These tables, along with subject metadata, were used as input for the statistical analyses and generating figures, and can be found in the Source Data file provided with the manuscript.

## Setting up R environment for statistical analyses

### Create needed functions for analyses

```

1 ##### CREATE FUNCTIONS NEEDED FOR RUNNING CODE #####
2
3 # Function to suppress warnings and messages
4 # parameters:
5 #   x - the command to suppress messages/warnings of
6 suppress <- function(x){invisible(capture.output(suppressMessages(suppressWarnings(x))))}
7
8 # Function to perform ANCOM-BC and perform additional actions like grabbing summary stats

```

```

9 # and calculating bias-corrected abundances. Note: this function should work with older
10 # versions of ANCOM-BC that used the 'zero_cut' parameter or newer versions that use the
11 # 'prv_cut' parameter.
12 # parameters:
13 #     ps - phyloseq object with otu_table() and sample_data() data
14 #     formula - formula for model to be tested by ANCOM-BC. make sure categorical
15 #               variables have been dummy coded with 1 (for test group) and
16 #               0 (reference group) or else the N calculations will not work
17 #               correctly.
18 #     remaining parameters have been set to ANCOM-BC defaults (see ANCOM-BC documentation)
19 ANCOMBC.plus <- function(ps, formula,
20                          p_adj_method="holm",
21                          zero_cut=0.9,
22                          lib_cut=0,
23                          group=NULL,
24                          struc_zero=FALSE,
25                          neg_lb=FALSE,
26                          tol=1E-5,
27                          max_iter=100,
28                          conserve=FALSE,
29                          alpha=0.05,
30                          global=FALSE){
31   library(phyloseq)
32   library(ANCOMBC)
33   ci <- function(coef, se){
34     lower.ci <- coef - 1.96*se
35     upper.ci <- coef + 1.96*se
36     return(c(lower.ci=lower.ci,upper.ci=upper.ci))
37   }
38   # make sure samples are rows and features are columns
39   if (taxa_are_rows(ps)){
40     ps <- phyloseq(t(otu_table(ps)), sample_data(ps))
41   }
42   # run ANCOM-BC
43   if ('prv_cut' %in% names(as.list(args(ancombc)))){
44     suppressWarnings(
45       ancom.res <- ancombc(ps, formula, p_adj_method, 1-zero_cut, lib_cut,
46                           group, struc_zero, neg_lb, tol, max_iter,
47                           conserve, alpha, global)
48     )
49   }else{
50     suppressWarnings(
51       ancom.res <- ancombc(ps, formula, p_adj_method, zero_cut, lib_cut,
52                           group, struc_zero, neg_lb, tol, max_iter,
53                           conserve, alpha, global)
54     )
55   }
56
57   # calculate bias-corrected abundances
58   samp_frac <- ancom.res$samp_frac
59   samp_frac[is.na(samp_frac)] <- 0
60   ps.adj <- prune_taxa(rownames(ancom.res$feature_table), ps)
61   otu_table(ps.adj) <- log(otu_table(ps.adj) + 1) - samp_frac
62   # filter for samples with data for variables included in model

```

```

63 for (var in seq_len(ncol(ancom.res$res[[1]]))){
64   ps <- phyloseq(otu_table(ps),
65     sample_data(ps)[!is.na(sample_data(ps)[,strsplit(formula,
66       " \\+ ")[[1]][var]),])
67   ps.adj <- phyloseq(otu_table(ps.adj),
68     sample_data(ps.adj)[!is.na(sample_data(ps.adj)[,strsplit(formula,
69       " \\+ ")[[1]][var]),])
70 }
71 # get summary of results
72 res <- data.frame()
73 for (var in seq_len(ncol(ancom.res$res[[1]]))){
74   # get variable name
75   var.name1 <- strsplit(formula, " \\+ ")[[1]][var]
76   var.name2 <- colnames(ancom.res$res[[1]][var]
77   # get N in each group
78   if (length(table(sample_data(ps)[,var.name1])) == 2){
79     group.1.index <- sample_data(ps)[,var.name1] ==
80       names(table(sample_data(ps)[,var.name1]))[2]
81     group.1.index[is.na(group.1.index)] <- FALSE
82     group.2.index <- sample_data(ps)[,var.name1] ==
83       names(table(sample_data(ps)[,var.name1]))[1]
84     group.2.index[is.na(group.2.index)] <- FALSE
85     n1 <- colSums(otu_table(ps)[group.1.index,] > 0)
86     n2 <- colSums(otu_table(ps)[group.2.index,] > 0)
87     mean1 <- exp(colMeans(otu_table(ps.adj)[group.1.index,]))
88     mean2 <- exp(colMeans(otu_table(ps.adj)[group.2.index,]))
89   }else{
90     n1 <- rep(nsamples(ps), ntaxa(ps))
91     names(n1) <- taxa_names(ps)
92     n2 <- rep(NA, ntaxa(ps))
93     names(n2) <- taxa_names(ps)
94     mean1 <- exp(colMeans(otu_table(ps.adj)))
95     mean2 <- rep(NA, ntaxa(ps.adj))
96     names(mean2) <- taxa_names(ps.adj)
97   }
98   # calculate fold change and confidence interval of fold change
99   if (length(table(sample_data(ps)[,var.name1])) == 2){
100     FC <- exp(ancom.res$res[[1]][,var.name2])
101     FC.lower <- c()
102     FC.upper <- c()
103     for (coef in seq_along(ancom.res$res[[1]][,var.name2])){
104       FC.lower <- c(FC.lower, exp(ci(ancom.res$res[[1]][,var.name2][coef],
105         ancom.res$res[[2]][,var.name2][coef])['lower.ci']))
106       FC.upper <- c(FC.upper, exp(ci(ancom.res$res[[1]][,var.name2][coef],
107         ancom.res$res[[2]][,var.name2][coef])['upper.ci']))
108     }
109   }else{
110     FC <- NA
111     FC.lower <- NA
112     FC.upper <- NA
113   }
114   # summarize results for variable
115   rvar <- data.frame(Variable=var.name1,
116     Feature=rownames(ancom.res$feature_table),

```



```

117     N1=n1[rownames(ancom.res$feature_table)],
118     N2=n2[rownames(ancom.res$feature_table)],
119     Mean1=mean1[rownames(ancom.res$feature_table)],
120     Mean2=mean2[rownames(ancom.res$feature_table)],
121     Beta=ancom.res$res[[1]][,var.name2],
122     SE=ancom.res$res[[2]][,var.name2],
123     P=ancom.res$res[[4]][,var.name2],
124     FDR=ancom.res$res[[5]][,var.name2],
125     FC=FC, FC_lower=FC.lower, FC_upper=FC.upper,
126     check.names=FALSE)
127 res <- rbind(res, rvar[order(rvar$P),])
128 # add untested features if they exist
129 if (nrow(rvar) != ntaxa(ps)){
130     res <- rbind(res,
131                 data.frame(Variable=var.name1,
132                             Feature=taxa_names(ps)[!(taxa_names(ps) %in%
133                                                         rownames(ancom.res$feature_table))],
134                             N1=n1[taxa_names(ps)[!(taxa_names(ps) %in%
135                                                         rownames(ancom.res$feature_table))]],
136                             N2=n2[taxa_names(ps)[!(taxa_names(ps) %in%
137                                                         rownames(ancom.res$feature_table))]],
138                             Mean1=NA, Mean2=NA, Beta=NA, SE=NA, P=NA, FDR=NA,
139                             FC=NA, FC_lower=NA, FC_upper=NA,
140                             check.names=FALSE))
141 }
142 }
143 return(list(result.summary=res, ancom.output=ancom.res, bias.corrected.ps=ps.adj))
144 }
145
146 # Function to perform MaAsLin2 on phyloseq object and
147 # perform additional actions like grabbing summary stats
148 # parameters:
149 #     ps - phyloseq object with otu_table() and sample_data() data
150 #     output - path to directory for output
151 #     metadata - vector listing variable names in sample_data() to include in the model
152 # remaining parameters have been set to Maaslin2 defaults (see Maaslin2 documentation)
153 MaAsLin2.plus <- function(ps, output, metadata,
154                           min_abundance=0,
155                           min_prevalence=0.1,
156                           min_variance=0,
157                           normalization='TSS',
158                           transform='LOG',
159                           analysis_method='LM',
160                           max_significance=0.25,
161                           random_effects=NULL,
162                           fixed_effects=NULL,
163                           correction='BH',
164                           standardize=TRUE,
165                           cores=1,
166                           plot_heatmap=TRUE,
167                           plot_scatter=TRUE,
168                           heatmap_first_n=50,
169                           reference=NULL){
170 library(phyloseq)

```

```

171 library(Maaslin2)
172 ci <- function(coef, se){
173   lower.ci <- coef - 1.96*se
174   upper.ci <- coef + 1.96*se
175   return(c(lower.ci=lower.ci,upper.ci=upper.ci))
176 }
177 # make sure samples are rows and features are columns
178 if (taxa_are_rows(ps)){
179   ps <- phyloseq(t(otu_table(ps)), sample_data(ps))
180 }
181 # make sure only LOG is chosen for MaAsLin transformation
182 if (!(transform == "LOG")){
183   stop('function only supports LOG transformation at this time')
184 }
185 # run MaAsLin2
186 input_data <- data.frame(otu_table(ps))
187 input_metadata <- data.frame(sample_data(ps)[,colnames(sample_data(ps)) %in% metadata])
188 fits <- Maaslin2(input_data, input_metadata, output, min_abundance, min_prevalence,
189                 min_variance,normalization, transform, analysis_method, max_significance,
190                 random_effects, fixed_effects, correction, standardize, cores,
191                 plot_heatmap, plot_scatter, heatmap_first_n, reference)
192 # put back original feature names
193 for (feat in seq_along(fits$results$feature)){
194   fits$results$feature[feat] <- taxa_names(ps)[make.names(taxa_names(ps)) ==
195                                               fits$results$feature[feat]]
196 }
197 # filter for samples with data for variables included in model
198 for (var in seq_along(unique(fits$results$metadata))){
199   ps <- phyloseq(otu_table(ps),
200                 sample_data(ps)[!is.na(sample_data(ps)[,metadata[var]]),])
201 }
202 # get summary of results
203 res <- data.frame()
204 for (var in seq_along(unique(fits$results$metadata))){
205   # get variable name
206   var.name <- metadata[var]
207   # get N in each group
208   if (length(table(sample_data(ps)[,var.name])) == 2){
209     group.1.index <- sample_data(ps)[,var.name] ==
210                       names(table(sample_data(ps)[,var.name]))[2]
211     group.1.index[is.na(group.1.index)] <- FALSE
212     group.2.index <- sample_data(ps)[,var.name] ==
213                       names(table(sample_data(ps)[,var.name]))[1]
214     group.2.index[is.na(group.2.index)] <- FALSE
215     n1 <- colSums(otu_table(ps)[group.1.index,] > 0)
216     n2 <- colSums(otu_table(ps)[group.2.index,] > 0)
217     mean1 <- colMeans(otu_table(ps)[group.1.index,])
218     mean2 <- colMeans(otu_table(ps)[group.2.index,])
219   }else{
220     n1 <- rep(sum(table(sample_data(ps)[,var.name])), ntaxa(ps))
221     names(n1) <- taxa_names(ps)
222     n2 <- rep(NA, ntaxa(ps))
223     names(n2) <- taxa_names(ps)
224     mean1 <- colMeans(otu_table(ps))

```

```

225     mean2 <- rep(NA, ntaxa(ps))
226     names(mean2) <- taxa_names(ps)
227   }
228   # calculate fold change and confidence interval of fold change
229   if(length(table(sample_data(ps)[,var.name])) == 2){
230     FC <- 2^(fits$results$coef[fits$results$metadata == var.name])
231     FC.lower <- c()
232     FC.upper <- c()
233     for (coef in seq_along(fits$results$coef[fits$results$metadata == var.name])){
234       FC.lower <- c(FC.lower, 2^(ci(fits$results$coef[fits$results$metadata ==
235                                     var.name][coef],
236                                     fits$results$stderr[fits$results$metadata ==
237                                                         var.name][coef])['lower.ci']))
238       FC.upper <- c(FC.upper, 2^(ci(fits$results$coef[fits$results$metadata ==
239                                     var.name][coef],
240                                     fits$results$stderr[fits$results$metadata ==
241                                                         var.name][coef])['upper.ci']))
242     }
243   }else{
244     FC <- NA
245     FC.lower <- NA
246     FC.upper <- NA
247   }
248   # summarize results for variable
249   rvar <- data.frame(Variable=var.name,
250                     Feature=fits$results$feature[fits$results$metadata == var.name],
251                     N1=n1[fits$results$feature[fits$results$metadata == var.name]],
252                     N2=n2[fits$results$feature[fits$results$metadata == var.name]],
253                     Mean1=mean1[fits$results$feature[fits$results$metadata == var.name]],
254                     Mean2=mean2[fits$results$feature[fits$results$metadata == var.name]],
255                     Beta=fits$results$coef[fits$results$metadata == var.name],
256                     SE=fits$results$stderr[fits$results$metadata == var.name],
257                     P=fits$results$pval[fits$results$metadata == var.name],
258                     FDR=p.adjust(fits$results$pval[fits$results$metadata == var.name],
259                                   method=correction),
260                     FC=FC, FC_lower=FC.lower, FC_upper=FC.upper,
261                     check.names=FALSE)
262   res <- rbind(res, rvar[order(rvar$P),])
263   # add untested features if they exist
264   if (nrow(rvar) != ntaxa(ps)){
265     res <- rbind(res,
266                 data.frame(Variable=var.name,
267                           Feature=taxa_names(ps)[!(taxa_names(ps) %in%
268                                                         fits$results$feature[fits$results$metadata == var.name])],
269                           N1=n1[taxa_names(ps)[!(taxa_names(ps) %in%
270                                                         fits$results$feature[fits$results$metadata == var.name])]],
271                           N2=n2[taxa_names(ps)[!(taxa_names(ps) %in%
272                                                         fits$results$feature[fits$results$metadata == var.name])]],
273                           Mean1=mean1[taxa_names(ps)[!(taxa_names(ps) %in%
274                                                         fits$results$feature[fits$results$metadata == var.name])]],
275                           Mean2=mean2[taxa_names(ps)[!(taxa_names(ps) %in%
276                                                         fits$results$feature[fits$results$metadata == var.name])]],
277                           Beta=NA, SE=NA, P=NA, FDR=NA, FC=NA, FC_lower=NA, FC_upper=NA,
278                           check.names=FALSE)

```

```

279         )
280     }
281 }
282 return(list(result.summary=res, Maaslin2.output=fits))
283 }
284
285 # Function to compute log2 transform for transforming relative abundances
286 # parameters:
287 #   x - a vector or data.frame of relative abundances
288 log2.trans <- function(x) {
289   y <- replace(x, x == 0, min(x[x>0]) / 2)
290   return(log2(y))
291 }
292
293 # Function to convert numbers from strings back to numbers in excel output
294 # parameters:
295 #   df - the data.frame being outputted into excel
296 #   wb - the excel workbook object
297 #   sheet - the excel sheet name
298 # colnames - does df have column names, TRUE/FALSE
299 convertNum <- function(df, wb, sheet, colnames) {
300   library(foreach)
301   cn <- expand.grid(seq_len(ncol(df)), seq_len(nrow(df)))[,1]
302   rn <- expand.grid(seq_len(ncol(df)), seq_len(nrow(df)))[,2]
303   trash <- foreach(cn=cn, rn=rn) %dopar% {
304     if (!is.numeric(df[rn,cn]) && !is.na(as.numeric(as.character(df[rn,cn])))) {
305       row.offset <- ifelse(colnames, 1, 0)
306       openxlsx::writeData(wb, sheet, as.numeric(as.character(df[rn,cn])),
307                           startCol=cn, startRow=row.offset + rn)
308     }
309   }
310 }

```

## Load required R packages

```

1  ##### LOAD REQUIRED R PACKAGES #####
2
3  # standard data manipulation R packages
4  suppress(library(dplyr))
5  suppress(library(reshape2))
6  suppress(library(readxl))
7  suppress(library(phyloseq))
8  suppress(library(tibble))
9  suppress(library(openxlsx))
10 suppress(library(foreach))
11 suppress(library(data.table))
12 suppress(library(gridExtra))
13 suppress(library(scales))
14 # R packages used in analysis or plotting
15 suppress(library(ggplot2))
16 suppress(library(ggh4x))
17 suppress(library(ggfortify))
18 suppress(library(ggvenn))

```

```

19 suppress(library(ggrepel))
20 suppress(library(vegan))
21 suppress(library(pairwiseCI))
22 suppress(library(vcd))
23 suppress(library(ANCOMBC))
24 suppress(library(Maaslin2))
25 suppress(library(igraph))

```

## Create excel styles used for formatting output

```

1  ##### CREATE EXCEL STYLES USED FOR FORMATTING OUTPUT #####
2
3  bold <- createStyle(textDecoration="bold")
4  center <- createStyle(halign="center", valign="center", wrapText=TRUE)
5  horizontal_border_med <- createStyle(border="top", borderStyle="medium")
6  horizontal_border_thin <- createStyle(border="top", borderStyle="thin")

```

## Make directories for output

```

1  ##### CREATE OUTPUT DIRECTORIES #####
2
3  system('mkdir PDSHOTGUNAnalysis_out')
4  system('mkdir PDSHOTGUNAnalysis_out/1.Metadata')
5  system('mkdir PDSHOTGUNAnalysis_out/2.Gut_microbiome_composition')
6  system('mkdir PDSHOTGUNAnalysis_out/3.Taxonomic_associations')
7  system('mkdir PDSHOTGUNAnalysis_out/4.a.Network_analysis')
8  system('mkdir PDSHOTGUNAnalysis_out/4.b.Gephi_network_plots')
9  system('mkdir PDSHOTGUNAnalysis_out/5.Gene_pathway_associations')
10 system('mkdir PDSHOTGUNAnalysis_out/6.Secondary_analyses')

```

## Subject characteristics: testing PD vs NHC

To determine what subject metadata variables are significantly different between PD and NHC subjects, tested each variable for association with PD using Fisher's exact test (via `fisher.test` function) for categorical variables and Wilcoxon rank-sum test (via `wilcox.test` function) for quantitative variables. Odds ratios and confidence intervals were calculated via the `fisher.test` function. If any 2x2 tables of categorical variables contained 0, then the function `Prop.or` from the `pairwiseCI` R package specifying `CImethod='Woolf'` was used to calculate odds ratios and confidence intervals. Subject metadata can be found in the Source Data file included with the manuscript.

```

1  ##### SUBJECT CHARACTERISTICS PD VS NHC #####
2
3  # read in metadata
4  metadata <- data.frame(read_xlsx('Source_Data.xlsx', sheet='subject_metadata'))
5  rownames(metadata) <- metadata$sample_name
6
7  # make new data.frame for metadata, so we do not alter original data.frame
8  subject.data <- metadata
9
10 # create result data.frame
11 results <- data.frame()

```

```

12
13 # samples passing QC
14 results <- rbind(results,
15                 data.frame(Category='',
16                             Metadata="Number of subjects who passed sequence and metadata QC",
17                             `PD N`=table(subject.data$Case_status)['PD'],
18                             `PD summary stats`="-",
19                             `NHC N`=table(subject.data$Case_status)['Control'],
20                             `NHC summary stats`="-",
21                             `Total N`=length(na.omit(subject.data$Case_status)),
22                             P="-", `OR [95%CI]`="-", check.names=FALSE))
23 # age
24 P.t <- length(na.omit(subset(subject.data, Case_status == "PD")$Age_at_collection))
25 P.avg <- mean(na.omit(subset(subject.data, Case_status == "PD")$Age_at_collection))
26 P.sd <- sd(na.omit(subset(subject.data, Case_status == "PD")$Age_at_collection))
27 C.t <- length(na.omit(subset(subject.data, Case_status == "Control")$Age_at_collection))
28 C.avg <- mean(na.omit(subset(subject.data, Case_status == "Control")$Age_at_collection))
29 C.sd <- sd(na.omit(subset(subject.data, Case_status == "Control")$Age_at_collection))
30 p <- wilcox.test(subset(subject.data, Case_status == "PD")$Age_at_collection,
31                 subset(subject.data, Case_status == "Control")$Age_at_collection)$p.value
32 results <- rbind(results, data.frame(Category='',
33                                     Metadata = "Age",
34                                     `PD N`=P.t,
35                                     `PD summary stats`=paste(round(P.avg, 1),
36                                                             round(P.sd, 1),
37                                                             sep="±"),
38                                     `NHC N`=C.t,
39                                     `NHC summary stats`=paste(round(C.avg, 1),
40                                                             round(C.sd, 1),
41                                                             sep="±"),
42                                     `Total N`=P.t+C.t,
43                                     P=formatC(p, format="e", digits=1),
44                                     `OR [95%CI]`="-",
45                                     check.names=FALSE))
46 # sex
47 P.f <- table(subset(subject.data, Case_status == "PD")$Sex)['F']
48 P.m <- table(subset(subject.data, Case_status == "PD")$Sex)['M']
49 C.f <- table(subset(subject.data, Case_status == "Control")$Sex)['F']
50 C.m <- table(subset(subject.data, Case_status == "Control")$Sex)['M']
51 p <- fisher.test(matrix(c(P.m,P.f,C.m,C.f), nrow=2))$p.value
52 or <- fisher.test(matrix(c(P.m,P.f,C.m,C.f), nrow=2))$estimate
53 ci <- paste(round(fisher.test(matrix(c(P.m,P.f,C.m,C.f), nrow=2))$conf.int[1],1),
54             round(fisher.test(matrix(c(P.m,P.f,C.m,C.f), nrow=2))$conf.int[2],1), sep='-')
55 results <- rbind(results, data.frame(Category='',
56                                     Metadata = "Sex (N & % male)",
57                                     `PD N`=P.f+P.m,
58                                     `PD summary stats`=paste(P.m,
59                                                             " ", "(",
60                                                             round(P.m/(P.f+P.m)*100, 0),
61                                                             "%", ") ",
62                                                             sep=""),
63                                     `NHC N`=C.f+C.m,
64                                     `NHC summary stats`=paste(C.m,
65                                                             " ", "(",

```

```

66                                                                 round(C.m/(C.f+C.m)*100, 0),
67                                                                 "%",")",
68                                                                 sep=""),
69
70     `Total N`=P.f+P.m+C.f+C.m,
71     P=formatC(p, format="e", digits=1),
72     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
73     check.names=FALSE))
74
75 # hispanic or latino
76 P.y <- table(subset(subject.data, Case_status == "PD")$Hispanic_or_Latino)['Y']
77 P.n <- table(subset(subject.data, Case_status == "PD")$Hispanic_or_Latino)['N']
78 C.y <- table(subset(subject.data, Case_status == "Control")$Hispanic_or_Latino)['Y']
79 C.n <- table(subset(subject.data, Case_status == "Control")$Hispanic_or_Latino)['N']
80 if (is.na(P.n)){P.n <- 0}
81 if (is.na(P.y)){P.y <- 0}
82 if (is.na(C.n)){C.n <- 0}
83 if (is.na(C.y)){C.y <- 0}
84 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
85 if (any(c(P.y, P.n, C.y, C.n)==0)){
86   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
87   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
88     CImethod='Woolf')$conf.int[1],1),
89     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
90     CImethod='Woolf')$conf.int[2],1), sep='-')
91 }else{
92   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
93   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
94     nrow=2))$conf.int[1],1),
95     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
96     nrow=2))$conf.int[2],1), sep='-')
97 }
98 results <- rbind(results, data.frame(Category='Ancestry',
99   Metadata ="Hispanic or Latino",
100   `PD N`=P.n+P.y,
101   `PD summary stats`=paste(P.y,
102     " ", "(",
103     round(P.y/(P.n+P.y)*100, 0),
104     "%",")",
105     sep=""),
106   `NHC N`=C.n+C.y,
107   `NHC summary stats`=paste(C.y,
108     " ", "(",
109     round(C.y/(C.n+C.y)*100, 0),
110     "%",")",
111     sep=""),
112   `Total N`=P.n+P.y+C.n+C.y,
113   P=formatC(p, format="e", digits=1),
114   `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
115   check.names=FALSE))
116
117 # race
118 P.y <- table(subset(subject.data, Case_status == "PD")$Race)['White']
119 P.n <- sum(table(subset(subject.data, Case_status == "PD")$Race))-P.y
120 C.y <- table(subset(subject.data, Case_status == "Control")$Race)['White']
121 C.n <- sum(table(subset(subject.data, Case_status == "Control")$Race))-C.y
122 if (is.na(P.n)){P.n <- 0}

```

```

120 if (is.na(P.y)){P.y <- 0}
121 if (is.na(C.n)){C.n <- 0}
122 if (is.na(C.y)){C.y <- 0}
123 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
124 if (any(c(P.y, P.n, C.y, C.n)==0)){
125   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
126   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
127                     CImethod='Woolf')$conf.int[1],1),
128              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
129                     CImethod='Woolf')$conf.int[2],1), sep='-')
130 }else{
131   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
132   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
133                                   nrow=2))$conf.int[1],1),
134              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
135                                   nrow=2))$conf.int[2],1), sep='-')
136 }
137 results <- rbind(results, data.frame(Category='',
138                                     Metadata ="Race (N & % White)",
139                                     `PD N`=P.n+P.y,
140                                     `PD summary stats`=paste(P.y,
141                                                                " ", "(",
142                                                                round(P.y/(P.n+P.y)*100, 0),
143                                                                "%", ")\"",
144                                                                sep=""),
145                                     `NHC N`=C.n+C.y,
146                                     `NHC summary stats`=paste(C.y,
147                                                                " ", "(",
148                                                                round(C.y/(C.n+C.y)*100, 0),
149                                                                "%", ")\"",
150                                                                sep=""),
151                                     `Total N`=P.n+P.y+C.n+C.y,
152                                     P=formatC(p, format="e", digits=1),
153                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
154                                     check.names=FALSE))
155 # jewish ancestry
156 P.y <- table(subset(subject.data, Case_status == "PD")$Jewish_ancestry)['Y']
157 P.n <- table(subset(subject.data, Case_status == "PD")$Jewish_ancestry)['N']
158 C.y <- table(subset(subject.data, Case_status == "Control")$Jewish_ancestry)['Y']
159 C.n <- table(subset(subject.data, Case_status == "Control")$Jewish_ancestry)['N']
160 if (is.na(P.n)){P.n <- 0}
161 if (is.na(P.y)){P.y <- 0}
162 if (is.na(C.n)){C.n <- 0}
163 if (is.na(C.y)){C.y <- 0}
164 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
165 if (any(c(P.y, P.n, C.y, C.n)==0)){
166   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
167   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
168                     CImethod='Woolf')$conf.int[1],1),
169              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
170                     CImethod='Woolf')$conf.int[2],1), sep='-')
171 }else{
172   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
173   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),

```



```

174         nrow=2))$conf.int[1],1),
175     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
176         nrow=2))$conf.int[2],1), sep='-')
177 }
178 results <- rbind(results, data.frame(Category='',
179     Metadata ="Jewish",
180     `PD N`=P.n+P.y,
181     `PD summary stats`=paste(P.y,
182         " ", "(",
183         round(P.y/(P.n+P.y)*100, 0),
184         "%", ") ",
185         sep=""),
186     `NHC N`=C.n+C.y,
187     `NHC summary stats`=paste(C.y,
188         " ", "(",
189         round(C.y/(C.n+C.y)*100, 0),
190         "%", ") ",
191         sep=""),
192     `Total N`=P.n+P.y+C.n+C.y,
193     P=formatC(p, format="e", digits=1),
194     `OR [95%CI]`=paste(round(or, 1), ' [',ci,'] ',sep=''),
195     check.names=FALSE))
196 # bmi
197 P.t <- length(na.omit(subset(subject.data, Case_status == "PD")$BMI))
198 P.avg <- mean(na.omit(subset(subject.data, Case_status == "PD")$BMI))
199 P.sd <- sd(na.omit(subset(subject.data, Case_status == "PD")$BMI))
200 C.t <- length(na.omit(subset(subject.data, Case_status == "Control")$BMI))
201 C.avg <- mean(na.omit(subset(subject.data, Case_status == "Control")$BMI))
202 C.sd <- sd(na.omit(subset(subject.data, Case_status == "Control")$BMI))
203 p <- wilcox.test(subset(subject.data, Case_status == "PD")$BMI,
204     subset(subject.data, Case_status == "Control")$BMI)$p.value
205 results <- rbind(results, data.frame(Category='Weight',
206     Metadata ="BMI",
207     `PD N`=P.t,
208     `PD summary stats`=paste(round(P.avg, 1),
209         round(P.sd, 1), sep="±"),
210     `NHC N`=C.t,
211     `NHC summary stats`=paste(round(C.avg, 1),
212         round(C.sd, 1), sep="±"),
213     `Total N`=P.t+C.t,
214     P=formatC(p, format="e", digits=1),
215     `OR [95%CI]`= "-",
216     check.names=FALSE))
217 # lost 10lbs in past year
218 P.y <- table(subset(subject.data, Case_status == "PD")$Loss_10lbs_in_last_year)['Y']
219 P.n <- table(subset(subject.data, Case_status == "PD")$Loss_10lbs_in_last_year)['N']
220 C.y <- table(subset(subject.data, Case_status == "Control")$Loss_10lbs_in_last_year)['Y']
221 C.n <- table(subset(subject.data, Case_status == "Control")$Loss_10lbs_in_last_year)['N']
222 if (is.na(P.n)){P.n <- 0}
223 if (is.na(P.y)){P.y <- 0}
224 if (is.na(C.n)){C.n <- 0}
225 if (is.na(C.y)){C.y <- 0}
226 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
227 if (any(c(P.y, P.n, C.y, C.n)==0)){

```

```

228 or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
229 ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
230               CImethod='Woolf')$conf.int[1],1),
231            round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
232               CImethod='Woolf')$conf.int[2],1), sep='-')
233 }else{
234 or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
235 ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
236               nrow=2))$conf.int[1],1),
237            round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
238               nrow=2))$conf.int[2],1), sep='-')
239 }
240 results <- rbind(results, data.frame(Category='',
241                                   Metadata ="Lost >10 pounds in past year",
242                                   `PD N`=P.n+P.y,
243                                   `PD summary stats`=paste(P.y,
244                                                           " ", "(",
245                                                           round(P.y/(P.n+P.y)*100, 0),
246                                                           "%", ") ",
247                                                           sep=""),
248                                   `NHC N`=C.n+C.y,
249                                   `NHC summary stats`=paste(C.y,
250                                                           " ", "(",
251                                                           round(C.y/(C.n+C.y)*100, 0),
252                                                           "%", ") ",
253                                                           sep=""),
254                                   `Total N`=P.n+P.y+C.n+C.y,
255                                   P=formatC(p, format="e", digits=1),
256                                   `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
257                                   check.names=FALSE))
258 # gained 10lbs in past year
259 P.y <- table(subset(subject.data, Case_status == "PD")$Gained_10lbs_in_last_year)['Y']
260 P.n <- table(subset(subject.data, Case_status == "PD")$Gained_10lbs_in_last_year)['N']
261 C.y <- table(subset(subject.data, Case_status == "Control")$Gained_10lbs_in_last_year)['Y']
262 C.n <- table(subset(subject.data, Case_status == "Control")$Gained_10lbs_in_last_year)['N']
263 if (is.na(P.n)){P.n <- 0}
264 if (is.na(P.y)){P.y <- 0}
265 if (is.na(C.n)){C.n <- 0}
266 if (is.na(C.y)){C.y <- 0}
267 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
268 if (any(c(P.y, P.n, C.y, C.n)==0)){
269 or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
270 ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
271               CImethod='Woolf')$conf.int[1],1),
272            round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
273               CImethod='Woolf')$conf.int[2],1), sep='-')
274 }else{
275 or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
276 ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
277               nrow=2))$conf.int[1],1),
278            round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
279               nrow=2))$conf.int[2],1), sep='-')
280 }
281 results <- rbind(results, data.frame(Category='',

```

```

282     Metadata ="Gained >10 pounds in past year",
283     `PD N`=P.n+P.y,
284     `PD summary stats`=paste(P.y,
285                               " ", "(",
286                               round(P.y/(P.n+P.y)*100, 0),
287                               "%", ")"),
288                               sep=""),
289     `NHC N`=C.n+C.y,
290     `NHC summary stats`=paste(C.y,
291                               " ", "(",
292                               round(C.y/(C.n+C.y)*100, 0),
293                               "%", ")"),
294                               sep=""),
295     `Total N`=P.n+P.y+C.n+C.y,
296     P=formatC(p, format="e", digits=1),
297     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
298     check.names=FALSE))
299 # fruits or vegetables daily
300 P.y <- table(subset(subject.data,
301                   Case_status == "PD")$How_often_do_you_eat_FRUITS_or_VEGETABLES) ["At least once a day"]
302 P.n <- sum(table(subset(subject.data,
303                   Case_status == "PD")$How_often_do_you_eat_FRUITS_or_VEGETABLES))-P.y
304 C.y <- table(subset(subject.data,
305                   Case_status == "Control")$How_often_do_you_eat_FRUITS_or_VEGETABLES) ["At least once a day"]
306 C.n <- sum(table(subset(subject.data,
307                   Case_status == "Control")$How_often_do_you_eat_FRUITS_or_VEGETABLES))-C.y
308 if (is.na(P.n)){P.n <- 0}
309 if (is.na(P.y)){P.y <- 0}
310 if (is.na(C.n)){C.n <- 0}
311 if (is.na(C.y)){C.y <- 0}
312 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
313 if (any(c(P.y, P.n, C.y, C.n)==0)){
314   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
315   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
316                             CImethod='Wolf')$conf.int[1],1),
317               round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
318                             CImethod='Wolf')$conf.int[2],1), sep='-')
319 }else{
320   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
321   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
322                                       nrow=2))$conf.int[1],1),
323               round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
324                                       nrow=2))$conf.int[2],1), sep='-')
325 }
326 results <- rbind(results, data.frame(Category='Diet',
327                                     Metadata ="Fruits or vegetables daily",
328                                     `PD N`=P.n+P.y,
329                                     `PD summary stats`=paste(P.y,
330                                                               " ", "(",
331                                                               round(P.y/(P.n+P.y)*100, 0),
332                                                               "%", ")"),
333                                                               sep=""),
334                                     `NHC N`=C.n+C.y,
335                                     `NHC summary stats`=paste(C.y,

```

```

336         " ", "(" ,
337         round(C.y/(C.n+C.y)*100, 0),
338         "%", ")" ,
339         sep="") ,
340     `Total N`=P.n+P.y+C.n+C.y,
341     P=formatC(p, format="e", digits=1),
342     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
343     check.names=FALSE))
344 # poultry, beef, pork, seafood, eggs daily
345 P.y <- table(subset(subject.data,
346   Case_status == "PD")$How_often_do_you_eat_POULTRY_BEEF_PORK_SEAFOOD_EGGS["At least once a day"])
347 P.n <- sum(table(subset(subject.data,
348   Case_status == "PD")$How_often_do_you_eat_POULTRY_BEEF_PORK_SEAFOOD_EGGS))-P.y
349 C.y <- table(subset(subject.data,
350   Case_status == "Control")$How_often_do_you_eat_POULTRY_BEEF_PORK_SEAFOOD_EGGS["At least once a day"])
351 C.n <- sum(table(subset(subject.data,
352   Case_status == "Control")$How_often_do_you_eat_POULTRY_BEEF_PORK_SEAFOOD_EGGS))-C.y
353 if (is.na(P.n)){P.n <- 0}
354 if (is.na(P.y)){P.y <- 0}
355 if (is.na(C.n)){C.n <- 0}
356 if (is.na(C.y)){C.y <- 0}
357 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
358 or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
359 if (any(c(P.y, P.n, C.y, C.n)==0)){
360   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
361   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
362     CImethod='Wolf')$conf.int[1],1),
363     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
364     CImethod='Wolf')$conf.int[2],1), sep='-')
365 }else{
366   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
367   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
368     nrow=2))$conf.int[1],1),
369     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
370     nrow=2))$conf.int[2],1), sep='-')
371 }
372 results <- rbind(results, data.frame(Category='',
373   Metadata ="Poultry, beef, pork, seafood, eggs daily",
374   `PD N`=P.n+P.y,
375   `PD summary stats`=paste(P.y,
376     " ", "(" ,
377     round(P.y/(P.n+P.y)*100, 0),
378     "%", ")" ,
379     sep="") ,
380   `NHC N`=C.n+C.y,
381   `NHC summary stats`=paste(C.y,
382     " ", "(" ,
383     round(C.y/(C.n+C.y)*100, 0),
384     "%", ")" ,
385     sep="") ,
386   `Total N`=P.n+P.y+C.n+C.y,
387   P=formatC(p, format="e", digits=1),
388   `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
389   check.names=FALSE))

```

```

390 # nuts daily
391 P.y <- table(subset(subject.data,
392   Case_status == "PD")$How_often_do_you_eat_NUTS)["At least once a day"]
393 P.n <- sum(table(subset(subject.data,
394   Case_status == "PD")$How_often_do_you_eat_NUTS))-P.y
395 C.y <- table(subset(subject.data,
396   Case_status == "Control")$How_often_do_you_eat_NUTS)["At least once a day"]
397 C.n <- sum(table(subset(subject.data,
398   Case_status == "Control")$How_often_do_you_eat_NUTS))-C.y
399 if (is.na(P.n)){P.n <- 0}
400 if (is.na(P.y)){P.y <- 0}
401 if (is.na(C.n)){C.n <- 0}
402 if (is.na(C.y)){C.y <- 0}
403 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
404 if (any(c(P.y, P.n, C.y, C.n)==0)){
405   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
406   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
407     CImethod='Wolf')$conf.int[1],1),
408     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
409     CImethod='Wolf')$conf.int[2],1), sep='-')
410 }else{
411   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
412   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
413     nrow=2))$conf.int[1],1),
414     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
415     nrow=2))$conf.int[2],1), sep='-')
416 }
417 results <- rbind(results, data.frame(Category='',
418   Metadata ="Nuts daily",
419   `PD N`=P.n+P.y,
420   `PD summary stats`=paste(P.y,
421     " ", "(",
422     round(P.y/(P.n+P.y)*100, 0),
423     "%", ")\"",
424     sep=""),
425   `NHC N`=C.n+C.y,
426   `NHC summary stats`=paste(C.y,
427     " ", "(",
428     round(C.y/(C.n+C.y)*100, 0),
429     "%", ")\"",
430     sep=""),
431   `Total N`=P.n+P.y+C.n+C.y,
432   P=formatC(p, format="e", digits=1),
433   `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
434   check.names=FALSE))
435 # yogurt at least a few times a week
436 P.y <- table(subset(subject.data,
437   Case_status == "PD")$How_often_do_you_eat_YOGURT)["At least once a day"]+
438   table(subset(subject.data,
439   Case_status == "PD")$How_often_do_you_eat_YOGURT)["Few times a week"]
440 P.n <- sum(table(subset(subject.data,
441   Case_status == "PD")$How_often_do_you_eat_YOGURT))-P.y
442 C.y <- table(subset(subject.data,
443   Case_status == "Control")$How_often_do_you_eat_YOGURT)["At least once a day"]+

```

```

444     table(subset(subject.data,
445       Case_status == "Control")$How_often_do_you_eat_YOGURT["Few times a week"])
446 C.n <- sum(table(subset(subject.data,
447   Case_status == "Control")$How_often_do_you_eat_YOGURT))-C.y
448 if (is.na(P.n)){P.n <- 0}
449 if (is.na(P.y)){P.y <- 0}
450 if (is.na(C.n)){C.n <- 0}
451 if (is.na(C.y)){C.y <- 0}
452 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
453 if (any(c(P.y, P.n, C.y, C.n)==0)){
454   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
455   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
456     CImethod='Wolf')$conf.int[1],1),
457     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
458     CImethod='Wolf')$conf.int[2],1), sep='-')
459 }else{
460   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
461   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
462     nrow=2))$conf.int[1],1),
463     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
464     nrow=2))$conf.int[2],1), sep='-')
465 }
466 results <- rbind(results, data.frame(Category='',
467   Metadata="Yogurt at least a few times a week",
468   `PD N`=P.n+P.y,
469   `PD summary stats`=paste(P.y,
470     " ", "(",
471     round(P.y/(P.n+P.y)*100, 0),
472     "%", ")\"",
473     sep=""),
474   `NHC N`=C.n+C.y,
475   `NHC summary stats`=paste(C.y,
476     " ", "(",
477     round(C.y/(C.n+C.y)*100, 0),
478     "%", ")\"",
479     sep=""),
480   `Total N`=P.n+P.y+C.n+C.y,
481   P=formatC(p, format="e", digits=1),
482   `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
483   check.names=FALSE))
484 # grains daily
485 P.y <- table(subset(subject.data,
486   Case_status == "PD")$How_often_do_you_eat_GRAINS["At least once a day"])
487 P.n <- sum(table(subset(subject.data,
488   Case_status == "PD")$How_often_do_you_eat_GRAINS))-P.y
489 C.y <- table(subset(subject.data,
490   Case_status == "Control")$How_often_do_you_eat_GRAINS["At least once a day"])
491 C.n <- sum(table(subset(subject.data,
492   Case_status == "Control")$How_often_do_you_eat_GRAINS))-C.y
493 if (is.na(P.n)){P.n <- 0}
494 if (is.na(P.y)){P.y <- 0}
495 if (is.na(C.n)){C.n <- 0}
496 if (is.na(C.y)){C.y <- 0}
497 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value

```

```

498 if (any(c(P.y, P.n, C.y, C.n)==0)){
499   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
500   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
501                       CImethod='Woolf')$conf.int[1],1),
502              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
503                       CImethod='Woolf')$conf.int[2],1), sep='-')
504 }else{
505   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
506   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
507                                   nrow=2))$conf.int[1],1),
508              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
509                                   nrow=2))$conf.int[2],1), sep='-')
510 }
511 results <- rbind(results, data.frame(Category='',
512                                   Metadata ="Grains daily",
513                                   `PD N`=P.n+P.y,
514                                   `PD summary stats`=paste(P.y,
515                                                           " ", "(" ,
516                                                           round(P.y/(P.n+P.y)*100, 0),
517                                                           "%", ") ",
518                                                           sep=""),
519                                   `NHC N`=C.n+C.y,
520                                   `NHC summary stats`=paste(C.y,
521                                                           " ", "(" ,
522                                                           round(C.y/(C.n+C.y)*100, 0),
523                                                           "%", ") ",
524                                                           sep=""),
525                                   `Total N`=P.n+P.y+C.n+C.y,
526                                   P=formatC(p, format="e", digits=1),
527                                   `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
528                                   check.names=FALSE))
529 # alcohol
530 P.y <- table(subset(subject.data, Case_status == "PD")$Do_you_drink_alcohol) ['Y']
531 P.n <- table(subset(subject.data, Case_status == "PD")$Do_you_drink_alcohol) ['N']
532 C.y <- table(subset(subject.data, Case_status == "Control")$Do_you_drink_alcohol) ['Y']
533 C.n <- table(subset(subject.data, Case_status == "Control")$Do_you_drink_alcohol) ['N']
534 if (is.na(P.n)){P.n <- 0}
535 if (is.na(P.y)){P.y <- 0}
536 if (is.na(C.n)){C.n <- 0}
537 if (is.na(C.y)){C.y <- 0}
538 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
539 if (any(c(P.y, P.n, C.y, C.n)==0)){
540   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
541   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
542                       CImethod='Woolf')$conf.int[1],1),
543              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
544                       CImethod='Woolf')$conf.int[2],1), sep='-')
545 }else{
546   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
547   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
548                                   nrow=2))$conf.int[1],1),
549              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
550                                   nrow=2))$conf.int[2],1), sep='-')
551 }

```

```

552 results <- rbind(results, data.frame(Category='',
553                                     Metadata ="Alcohol",
554                                     `PD N`=P.n+P.y,
555                                     `PD summary stats`=paste(P.y,
556                                                             " ", "(",
557                                                             round(P.y/(P.n+P.y)*100, 0),
558                                                             "%", ")"),
559                                     sep=""),
560                                     `NHC N`=C.n+C.y,
561                                     `NHC summary stats`=paste(C.y,
562                                                             " ", "(",
563                                                             round(C.y/(C.n+C.y)*100, 0),
564                                                             "%", ")"),
565                                     sep=""),
566                                     `Total N`=P.n+P.y+C.n+C.y,
567                                     P=formatC(p, format="e", digits=1),
568                                     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
569                                     check.names=FALSE))
570 # do you smoke
571 P.y <- table(subset(subject.data, Case_status == "PD")$Do_you_smoke) ['Y']
572 P.n <- table(subset(subject.data, Case_status == "PD")$Do_you_smoke) ['N']
573 C.y <- table(subset(subject.data, Case_status == "Control")$Do_you_smoke) ['Y']
574 C.n <- table(subset(subject.data, Case_status == "Control")$Do_you_smoke) ['N']
575 if (is.na(P.n)){P.n <- 0}
576 if (is.na(P.y)){P.y <- 0}
577 if (is.na(C.n)){C.n <- 0}
578 if (is.na(C.y)){C.y <- 0}
579 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
580 if (any(c(P.y, P.n, C.y, C.n)==0)){
581   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
582   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
583                           CImethod='Woolf')$conf.int[1],1),
584              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
585                           CImethod='Woolf')$conf.int[2],1), sep='-')
586 }else{
587   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
588   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
589                                     nrow=2))$conf.int[1],1),
590              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
591                                     nrow=2))$conf.int[2],1), sep='-')
592 }
593 results <- rbind(results, data.frame(Category='',
594                                     Metadata ="Tobacco",
595                                     `PD N`=P.n+P.y,
596                                     `PD summary stats`=paste(P.y,
597                                                             " ", "(",
598                                                             round(P.y/(P.n+P.y)*100, 0),
599                                                             "%", ")"),
600                                     sep=""),
601                                     `NHC N`=C.n+C.y,
602                                     `NHC summary stats`=paste(C.y,
603                                                             " ", "(",
604                                                             round(C.y/(C.n+C.y)*100, 0),
605                                                             "%", ")"),

```



```

606                                     sep=""),
607     `Total N`=P.n+P.y+C.n+C.y,
608     P=formatC(p, format="e", digits=1),
609     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
610     check.names=FALSE))
611 # caffeine
612 P.y <- table(subset(subject.data,
613     Case_status == "PD")$Do_you_drink_caffeinated_beverages)['Y']
614 P.n <- table(subset(subject.data,
615     Case_status == "PD")$Do_you_drink_caffeinated_beverages)['N']
616 C.y <- table(subset(subject.data,
617     Case_status == "Control")$Do_you_drink_caffeinated_beverages)['Y']
618 C.n <- table(subset(subject.data,
619     Case_status == "Control")$Do_you_drink_caffeinated_beverages)['N']
620 if (is.na(P.n)){P.n <- 0}
621 if (is.na(P.y)){P.y <- 0}
622 if (is.na(C.n)){C.n <- 0}
623 if (is.na(C.y)){C.y <- 0}
624 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
625 if (any(c(P.y, P.n, C.y, C.n)==0)){
626     or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
627     ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
628         CImethod='Wolf')$conf.int[1],1),
629         round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
630         CImethod='Wolf')$conf.int[2],1), sep='-')
631 }else{
632     or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
633     ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
634         nrow=2))$conf.int[1],1),
635         round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
636         nrow=2))$conf.int[2],1), sep='-')
637 }
638 results <- rbind(results, data.frame(Category='',
639     Metadata ="Caffeine",
640     `PD N`=P.n+P.y,
641     `PD summary stats`=paste(P.y,
642         " ", "(",
643         round(P.y/(P.n+P.y)*100, 0),
644         "%", ")\"",
645         sep=""),
646     `NHC N`=C.n+C.y,
647     `NHC summary stats`=paste(C.y,
648         " ", "(",
649         round(C.y/(C.n+C.y)*100, 0),
650         "%", ")\"",
651         sep=""),
652     `Total N`=P.n+P.y+C.n+C.y,
653     P=formatC(p, format="e", digits=1),
654     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
655     check.names=FALSE))
656 # constipation day of stool collection
657 P.y <- table(subset(subject.data,
658     Case_status == "PD")$Day_of_stool_collection_constipation)['Y']
659 P.n <- table(subset(subject.data,

```

```

660     Case_status == "PD")$Day_of_stool_collection_constipation)['N']
661 C.y <- table(subset(subject.data,
662     Case_status == "Control")$Day_of_stool_collection_constipation)['Y']
663 C.n <- table(subset(subject.data,
664     Case_status == "Control")$Day_of_stool_collection_constipation)['N']
665 if (is.na(P.n)){P.n <- 0}
666 if (is.na(P.y)){P.y <- 0}
667 if (is.na(C.n)){C.n <- 0}
668 if (is.na(C.y)){C.y <- 0}
669 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
670 if (any(c(P.y, P.n, C.y, C.n)==0)){
671   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
672   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
673     CImethod='Wolf')$conf.int[1],1),
674     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
675     CImethod='Wolf')$conf.int[2],1), sep='-')
676 }else{
677   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
678   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
679     nrow=2))$conf.int[1],1),
680     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
681     nrow=2))$conf.int[2],1), sep='-')
682 }
683 results <- rbind(results,
684     data.frame(Category='GI health on day of stool collection',
685     Metadata ="Constipation (no bowel movement) in >=3 days prior to stool collection",
686     `PD N`=P.n+P.y,
687     `PD summary stats`=paste(P.y,
688     " ", "(",
689     round(P.y/(P.n+P.y)*100, 0),
690     "%", ") ",
691     sep=""),
692     `NHC N`=C.n+C.y,
693     `NHC summary stats`=paste(C.y,
694     " ", "(",
695     round(C.y/(C.n+C.y)*100, 0),
696     "%", ") ",
697     sep=""),
698     `Total N`=P.n+P.y+C.n+C.y,
699     P=formatC(p, format="e", digits=1),
700     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
701     check.names=FALSE))
702 # diarrhea day of stool collection
703 P.y <- table(subset(subject.data,
704     Case_status == "PD")$Day_of_stool_collection_diarrhea)['Y']
705 P.n <- table(subset(subject.data,
706     Case_status == "PD")$Day_of_stool_collection_diarrhea)['N']
707 C.y <- table(subset(subject.data,
708     Case_status == "Control")$Day_of_stool_collection_diarrhea)['Y']
709 C.n <- table(subset(subject.data,
710     Case_status == "Control")$Day_of_stool_collection_diarrhea)['N']
711 if (is.na(P.n)){P.n <- 0}
712 if (is.na(P.y)){P.y <- 0}
713 if (is.na(C.n)){C.n <- 0}

```

```

714 if (is.na(C.y)){C.y <- 0}
715 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
716 if (any(c(P.y, P.n, C.y, C.n)==0)){
717   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
718   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
719                     CImethod='Woolf')$conf.int[1],1),
720              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
721                     CImethod='Woolf')$conf.int[2],1), sep='-')
722 }else{
723   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
724   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
725                                     nrow=2))$conf.int[1],1),
726              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
727                                     nrow=2))$conf.int[2],1), sep='-')
728 }
729 results <- rbind(results, data.frame(Category='',
730                                     Metadata ="Diarrhea",
731                                     `PD N`=P.n+P.y,
732                                     `PD summary stats`=paste(P.y,
733                                                                " ", "(" ,
734                                                                round(P.y/(P.n+P.y)*100, 0),
735                                                                "%", ") ",
736                                                                sep=""),
737                                     `NHC N`=C.n+C.y,
738                                     `NHC summary stats`=paste(C.y,
739                                                                " ", "(" ,
740                                                                round(C.y/(C.n+C.y)*100, 0),
741                                                                "%", ") ",
742                                                                sep=""),
743                                     `Total N`=P.n+P.y+C.n+C.y,
744                                     P=formatC(p, format="e", digits=1),
745                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
746                                     check.names=FALSE))
747 # abdominal pain day of stool collection
748 P.y <- table(subset(subject.data,
749                   Case_status == "PD")$Day_of_stool_collection_abdominal_pain)['Y']
750 P.n <- table(subset(subject.data,
751                   Case_status == "PD")$Day_of_stool_collection_abdominal_pain)['N']
752 C.y <- table(subset(subject.data,
753                   Case_status == "Control")$Day_of_stool_collection_abdominal_pain)['Y']
754 C.n <- table(subset(subject.data,
755                   Case_status == "Control")$Day_of_stool_collection_abdominal_pain)['N']
756 if (is.na(P.n)){P.n <- 0}
757 if (is.na(P.y)){P.y <- 0}
758 if (is.na(C.n)){C.n <- 0}
759 if (is.na(C.y)){C.y <- 0}
760 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
761 if (any(c(P.y, P.n, C.y, C.n)==0)){
762   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
763   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
764                     CImethod='Woolf')$conf.int[1],1),
765              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
766                     CImethod='Woolf')$conf.int[2],1), sep='-')
767 }else{

```

```

768 or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
769 ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
770               nrow=2))$conf.int[1],1),
771           round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
772               nrow=2))$conf.int[2],1), sep='-')
773 }
774 results <- rbind(results, data.frame(Category='',
775               Metadata ="Abdominal pain",
776               `PD N`=P.n+P.y,
777               `PD summary stats`=paste(P.y,
778               " ", "(" ,
779               round(P.y/(P.n+P.y)*100, 0),
780               "%", ") ",
781               sep=""),
782               `NHC N`=C.n+C.y,
783               `NHC summary stats`=paste(C.y,
784               " ", "(" ,
785               round(C.y/(C.n+C.y)*100, 0),
786               "%", ") ",
787               sep=""),
788               `Total N`=P.n+P.y+C.n+C.y,
789               P=formatC(p, format="e", digits=1),
790               `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
791               check.names=FALSE))
792 # excess gas day of stool collection
793 P.y <- table(subset(subject.data,
794               Case_status == "PD")$Day_of_stool_collection_excess_gas)['Y']
795 P.n <- table(subset(subject.data,
796               Case_status == "PD")$Day_of_stool_collection_excess_gas)['N']
797 C.y <- table(subset(subject.data,
798               Case_status == "Control")$Day_of_stool_collection_excess_gas)['Y']
799 C.n <- table(subset(subject.data,
800               Case_status == "Control")$Day_of_stool_collection_excess_gas)['N']
801 if (is.na(P.n)){P.n <- 0}
802 if (is.na(P.y)){P.y <- 0}
803 if (is.na(C.n)){C.n <- 0}
804 if (is.na(C.y)){C.y <- 0}
805 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
806 if (any(c(P.y, P.n, C.y, C.n)==0)){
807   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
808   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
809               CImethod='Woolf')$conf.int[1],1),
810           round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
811               CImethod='Woolf')$conf.int[2],1), sep='-')
812 }else{
813   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
814   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
815               nrow=2))$conf.int[1],1),
816           round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
817               nrow=2))$conf.int[2],1), sep='-')
818 }
819 results <- rbind(results, data.frame(Category='',
820               Metadata ="Excess gas",
821               `PD N`=P.n+P.y,

```

```

822     `PD summary stats`=paste(P.y,
823                               " ", "(",
824                               round(P.y/(P.n+P.y)*100, 0),
825                               "%", ")"),
826                               sep=""),
827     `NHC N`=C.n+C.y,
828     `NHC summary stats`=paste(C.y,
829                               " ", "(",
830                               round(C.y/(C.n+C.y)*100, 0),
831                               "%", ")"),
832                               sep=""),
833     `Total N`=P.n+P.y+C.n+C.y,
834     P=formatC(p, format="e", digits=1),
835     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
836     check.names=FALSE))
837 # bloating day of stool collection
838 P.y <- table(subset(subject.data,
839                   Case_status == "PD")$Day_of_stool_collection_bloating) ['Y']
840 P.n <- table(subset(subject.data,
841                   Case_status == "PD")$Day_of_stool_collection_bloating) ['N']
842 C.y <- table(subset(subject.data,
843                   Case_status == "Control")$Day_of_stool_collection_bloating) ['Y']
844 C.n <- table(subset(subject.data,
845                   Case_status == "Control")$Day_of_stool_collection_bloating) ['N']
846 if (is.na(P.n)){P.n <- 0}
847 if (is.na(P.y)){P.y <- 0}
848 if (is.na(C.n)){C.n <- 0}
849 if (is.na(C.y)){C.y <- 0}
850 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
851 if (any(c(P.y, P.n, C.y, C.n)==0)){
852   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
853   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
854                         CImethod='Wolf')$conf.int[1],1),
855              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
856                         CImethod='Wolf')$conf.int[2],1), sep='-')
857 }else{
858   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
859   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
860                                     nrow=2))$conf.int[1],1),
861              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
862                                     nrow=2))$conf.int[2],1), sep='-')
863 }
864 results <- rbind(results, data.frame(Category='',
865                                     Metadata ="Bloating",
866                                     `PD N`=P.n+P.y,
867                                     `PD summary stats`=paste(P.y,
868                                                                 " ", "(",
869                                                                 round(P.y/(P.n+P.y)*100, 0),
870                                                                 "%", ")"),
871                                     sep=""),
872                                     `NHC N`=C.n+C.y,
873                                     `NHC summary stats`=paste(C.y,
874                                                                 " ", "(",
875                                                                 round(C.y/(C.n+C.y)*100, 0),

```

```

876                                     "%",")",
877                                     sep=""),
878     `Total N`=P.n+P.y+C.n+C.y,
879     P=formatC(p, format="e", digits=1),
880     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
881     check.names=FALSE))
882 # GI discomfort day of stool collection
883 P.y <- table(subset(subject.data,
884     Case_status == "PD")$Day_of_stool_collection_digestion_issue)['Y']
885 P.n <- table(subset(subject.data,
886     Case_status == "PD")$Day_of_stool_collection_digestion_issue)['N']
887 C.y <- table(subset(subject.data,
888     Case_status == "Control")$Day_of_stool_collection_digestion_issue)['Y']
889 C.n <- table(subset(subject.data,
890     Case_status == "Control")$Day_of_stool_collection_digestion_issue)['N']
891 if (is.na(P.n)){P.n <- 0}
892 if (is.na(P.y)){P.y <- 0}
893 if (is.na(C.n)){C.n <- 0}
894 if (is.na(C.y)){C.y <- 0}
895 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
896 if (any(c(P.y, P.n, C.y, C.n)==0)){
897     or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
898     ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
899         CImethod='Woolf')$conf.int[1],1),
900         round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
901             CImethod='Woolf')$conf.int[2],1), sep='-')
902 }else{
903     or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
904     ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
905         nrow=2))$conf.int[1],1),
906         round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
907             nrow=2))$conf.int[2],1), sep='-')
908 }
909 results <- rbind(results,
910     data.frame(Category=' ',
911         Metadata ="GI discomfort on day of stool collection (yes to any of the five items)",
912         `PD N`=P.n+P.y,
913         `PD summary stats`=paste(P.y,
914             " ", "(",
915             round(P.y/(P.n+P.y)*100, 0),
916             "%",")",
917             sep=""),
918         `NHC N`=C.n+C.y,
919         `NHC summary stats`=paste(C.y,
920             " ", "(",
921             round(C.y/(C.n+C.y)*100, 0),
922             "%",")",
923             sep=""),
924         `Total N`=P.n+P.y+C.n+C.y,
925         P=formatC(p, format="e", digits=1),
926         `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
927         check.names=FALSE))
928 # bristol stool chart
929 P.t <- length(na.omit(subset(subject.data, Case_status == "PD")$Bristol_stool_chart))

```

```

930 P.avg <- mean(na.omit(subset(subject.data, Case_status == "PD")$Bristol_stool_chart))
931 P.sd <- sd(na.omit(subset(subject.data, Case_status == "PD")$Bristol_stool_chart))
932 C.t <- length(na.omit(subset(subject.data, Case_status == "Control")$Bristol_stool_chart))
933 C.avg <- mean(na.omit(subset(subject.data, Case_status == "Control")$Bristol_stool_chart))
934 C.sd <- sd(na.omit(subset(subject.data, Case_status == "Control")$Bristol_stool_chart))
935 p <- wilcox.test(subset(subject.data, Case_status == "PD")$Bristol_stool_chart,
936                 subset(subject.data, Case_status == "Control")$Bristol_stool_chart)$p.value
937 results <- rbind(results, data.frame(Category=' ',
938                                     Metadata="Bristol stool chart",
939                                     `PD N`=P.t,
940                                     `PD summary stats`=paste(round(P.avg, 1),
941                                                                round(P.sd, 1), sep="±"),
942                                     `NHC N`=C.t,
943                                     `NHC summary stats`=paste(round(C.avg, 1),
944                                                                round(C.sd, 1), sep="±"),
945                                     `Total N`=P.t+C.t,
946                                     P=formatC(p, format="e", digits=1),
947                                     `OR [95%CI]`="-", check.names=FALSE))
948 # constipation in the past 3 months
949 P.y <- table(subset(subject.data, Case_status == "PD")$Constipation)[`Y`]
950 P.n <- table(subset(subject.data, Case_status == "PD")$Constipation)[`N`]
951 C.y <- table(subset(subject.data, Case_status == "Control")$Constipation)[`Y`]
952 C.n <- table(subset(subject.data, Case_status == "Control")$Constipation)[`N`]
953 if (is.na(P.n)){P.n <- 0}
954 if (is.na(P.y)){P.y <- 0}
955 if (is.na(C.n)){C.n <- 0}
956 if (is.na(C.y)){C.y <- 0}
957 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
958 if (any(c(P.y, P.n, C.y, C.n)==0)){
959   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
960   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
961                           CImethod='Woolf')$conf.int[1],1),
962              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
963                           CImethod='Woolf')$conf.int[2],1), sep='-')
964 }else{
965   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
966   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
967                                       nrow=2))$conf.int[1],1),
968              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
969                                       nrow=2))$conf.int[2],1), sep='-')
970 }
971 results <- rbind(results,
972                 data.frame(Category='GI health in 3 months prior to stool collection',
973                           Metadata="Constipation (< 3 bowel movements per week)",
974                           `PD N`=P.n+P.y,
975                           `PD summary stats`=paste(P.y,
976                                                    " ", "(",
977                                                    round(P.y/(P.n+P.y)*100, 0),
978                                                    "%", ")",
979                                                    sep=""),
980                           `NHC N`=C.n+C.y,
981                           `NHC summary stats`=paste(C.y,
982                                                    " ", "(",
983                                                    round(C.y/(C.n+C.y)*100, 0),

```

```

984                                     "%",")",
985                                     sep=""),
986     `Total N`=P.n+P.y+C.n+C.y,
987     P=formatC(p, format="e", digits=1),
988     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
989     check.names=FALSE))
990 # diarrheal (once a week or more)
991 P.y <- table(subset(subject.data, Case_status == "PD")$Diarrhea)['Y']
992 P.n <- table(subset(subject.data, Case_status == "PD")$Diarrhea)['N']
993 C.y <- table(subset(subject.data, Case_status == "Control")$Diarrhea)['Y']
994 C.n <- table(subset(subject.data, Case_status == "Control")$Diarrhea)['N']
995 if (is.na(P.n)){P.n <- 0}
996 if (is.na(P.y)){P.y <- 0}
997 if (is.na(C.n)){C.n <- 0}
998 if (is.na(C.y)){C.y <- 0}
999 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1000 if (any(c(P.y, P.n, C.y, C.n)==0)){
1001   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
1002   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1003                       CImethod='Wolf')$conf.int[1],1),
1004              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1005                       CImethod='Wolf')$conf.int[2],1), sep='-')
1006 }else{
1007   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1008   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1009                                     nrow=2))$conf.int[1],1),
1010              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1011                                     nrow=2))$conf.int[2],1), sep='-')
1012 }
1013 results <- rbind(results, data.frame(Category='',
1014                                     Metadata ="Diarrheal (once a week or more)",
1015                                     `PD N`=P.n+P.y,
1016                                     `PD summary stats`=paste(P.y,
1017                                                                " ", "(",
1018                                                                round(P.y/(P.n+P.y)*100, 0),
1019                                                                "%", ")",
1020                                                                sep=""),
1021                                     `NHC N`=C.n+C.y,
1022                                     `NHC summary stats`=paste(C.y,
1023                                                                " ", "(",
1024                                                                round(C.y/(C.n+C.y)*100, 0),
1025                                                                "%", ")",
1026                                                                sep=""),
1027                                     `Total N`=P.n+P.y+C.n+C.y,
1028                                     P=formatC(p, format="e", digits=1),
1029                                     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
1030                                     check.names=FALSE))
1031 # colitis
1032 P.y <- table(subset(subject.data, Case_status == "PD")$Colitis)['Y']
1033 P.n <- table(subset(subject.data, Case_status == "PD")$Colitis)['N']
1034 C.y <- table(subset(subject.data, Case_status == "Control")$Colitis)['Y']
1035 C.n <- table(subset(subject.data, Case_status == "Control")$Colitis)['N']
1036 if (is.na(P.n)){P.n <- 0}
1037 if (is.na(P.y)){P.y <- 0}

```



```

1038 if (is.na(C.n)){C.n <- 0}
1039 if (is.na(C.y)){C.y <- 0}
1040 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1041 if (any(c(P.y, P.n, C.y, C.n)==0)){
1042   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1043   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1044                       CImethod='Woolf')$conf.int[1],1),
1045               round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1046                       CImethod='Woolf')$conf.int[2],1), sep='-')
1047 }else{
1048   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1049   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1050                                   nrow=2))$conf.int[1],1),
1051               round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1052                                   nrow=2))$conf.int[2],1), sep='-')
1053 }
1054 results <- rbind(results, data.frame(Category='GI disease',
1055                                     Metadata ="Colitis",
1056                                     `PD N`=P.n+P.y,
1057                                     `PD summary stats`=paste(P.y,
1058                                                                " ", "(" ,
1059                                                                round(P.y/(P.n+P.y)*100, 0),
1060                                                                "%", ") ",
1061                                                                sep=""),
1062                                     `NHC N`=C.n+C.y,
1063                                     `NHC summary stats`=paste(C.y,
1064                                                                " ", "(" ,
1065                                                                round(C.y/(C.n+C.y)*100, 0),
1066                                                                "%", ") ",
1067                                                                sep=""),
1068                                     `Total N`=P.n+P.y+C.n+C.y,
1069                                     P=formatC(p, format="e", digits=1),
1070                                     `OR [95%CI]`=paste(round(or, 1), ' [',ci,'] ',sep=''),
1071                                     check.names=FALSE))
1072 # IBS
1073 P.y <- table(subset(subject.data, Case_status == "PD")$IBS) ['Y']
1074 P.n <- table(subset(subject.data, Case_status == "PD")$IBS) ['N']
1075 C.y <- table(subset(subject.data, Case_status == "Control")$IBS) ['Y']
1076 C.n <- table(subset(subject.data, Case_status == "Control")$IBS) ['N']
1077 if (is.na(P.n)){P.n <- 0}
1078 if (is.na(P.y)){P.y <- 0}
1079 if (is.na(C.n)){C.n <- 0}
1080 if (is.na(C.y)){C.y <- 0}
1081 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1082 if (any(c(P.y, P.n, C.y, C.n)==0)){
1083   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1084   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1085                       CImethod='Woolf')$conf.int[1],1),
1086               round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1087                       CImethod='Woolf')$conf.int[2],1), sep='-')
1088 }else{
1089   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1090   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1091                                   nrow=2))$conf.int[1],1),

```

```

1092         round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1093                                 nrow=2))$conf.int[2],1), sep='-')
1094     }
1095     results <- rbind(results, data.frame(Category='',
1096                                         Metadata ="Irritable bowel syndrome",
1097                                         `PD N`=P.n+P.y,
1098                                         `PD summary stats`=paste(P.y,
1099                                                                    " ", "(",
1100                                                                    round(P.y/(P.n+P.y)*100, 0),
1101                                                                    "%", ")"),
1102                                                                    sep=""),
1103                                         `NHC N`=C.n+C.y,
1104                                         `NHC summary stats`=paste(C.y,
1105                                                                    " ", "(",
1106                                                                    round(C.y/(C.n+C.y)*100, 0),
1107                                                                    "%", ")"),
1108                                                                    sep=""),
1109                                         `Total N`=P.n+P.y+C.n+C.y,
1110                                         P=formatC(p, format="e", digits=1),
1111                                         `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
1112                                         check.names=FALSE))
1113     # Crohn's disease
1114     P.y <- table(subset(subject.data, Case_status == "PD")$Crohns_disease)['Y']
1115     P.n <- table(subset(subject.data, Case_status == "PD")$Crohns_disease)['N']
1116     C.y <- table(subset(subject.data, Case_status == "Control")$Crohns_disease)['Y']
1117     C.n <- table(subset(subject.data, Case_status == "Control")$Crohns_disease)['N']
1118     if (is.na(P.n)){P.n <- 0}
1119     if (is.na(P.y)){P.y <- 0}
1120     if (is.na(C.n)){C.n <- 0}
1121     if (is.na(C.y)){C.y <- 0}
1122     p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1123     if (any(c(P.y, P.n, C.y, C.n)==0)){
1124         or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1125         ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1126                                CImethod='Woolf')$conf.int[1],1),
1127                    round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1128                                CImethod='Woolf')$conf.int[2],1), sep='-')
1129     }else{
1130         or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1131         ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1132                                             nrow=2))$conf.int[1],1),
1133                    round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1134                                             nrow=2))$conf.int[2],1), sep='-')
1135     }
1136     results <- rbind(results, data.frame(Category='',
1137                                         Metadata ="Crohn's disease",
1138                                         `PD N`=P.n+P.y,
1139                                         `PD summary stats`=paste(P.y,
1140                                                                    " ", "(",
1141                                                                    round(P.y/(P.n+P.y)*100, 1),
1142                                                                    "%", ")"),
1143                                                                    sep=""),
1144                                         `NHC N`=C.n+C.y,
1145                                         `NHC summary stats`=paste(C.y,

```

```

1146         " ", "(" ,
1147         round(C.y/(C.n+C.y)*100, 1),
1148         "%", ")" ,
1149         sep=""),
1150     `Total N`=P.n+P.y+C.n+C.y,
1151     P=formatC(p, format="e", digits=1),
1152     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
1153     check.names=FALSE))
1154 # IBD
1155 P.y <- table(subset(subject.data, Case_status == "PD")$IBD) ['Y']
1156 P.n <- table(subset(subject.data, Case_status == "PD")$IBD) ['N']
1157 C.y <- table(subset(subject.data, Case_status == "Control")$IBD) ['Y']
1158 C.n <- table(subset(subject.data, Case_status == "Control")$IBD) ['N']
1159 if (is.na(P.n)){P.n <- 0}
1160 if (is.na(P.y)){P.y <- 0}
1161 if (is.na(C.n)){C.n <- 0}
1162 if (is.na(C.y)){C.y <- 0}
1163 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1164 if (any(c(P.y, P.n, C.y, C.n)==0)){
1165     or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1166     ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1167         CImethod='Woolf')$conf.int[1],1),
1168         round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1169         CImethod='Woolf')$conf.int[2],1), sep='-')
1170 }else{
1171     or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1172     ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1173         nrow=2))$conf.int[1],1),
1174         round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1175         nrow=2))$conf.int[2],1), sep='-')
1176 }
1177 results <- rbind(results, data.frame(Category='',
1178     Metadata ="Inflammatory bowel disease",
1179     `PD N`=P.n+P.y,
1180     `PD summary stats`=paste(P.y,
1181         " ", "(" ,
1182         round(P.y/(P.n+P.y)*100, 0),
1183         "%", ")" ,
1184         sep=""),
1185     `NHC N`=C.n+C.y,
1186     `NHC summary stats`=paste(C.y,
1187         " ", "(" ,
1188         round(C.y/(C.n+C.y)*100, 0),
1189         "%", ")" ,
1190         sep=""),
1191     `Total N`=P.n+P.y+C.n+C.y,
1192     P=formatC(p, format="e", digits=1),
1193     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
1194     check.names=FALSE))
1195 # ulcers in past 3 months
1196 P.y <- table(subset(subject.data, Case_status == "PD")$Ulcer_past_3_months) ['Y']
1197 P.n <- table(subset(subject.data, Case_status == "PD")$Ulcer_past_3_months) ['N']
1198 C.y <- table(subset(subject.data, Case_status == "Control")$Ulcer_past_3_months) ['Y']
1199 C.n <- table(subset(subject.data, Case_status == "Control")$Ulcer_past_3_months) ['N']

```

```

1200 if (is.na(P.n)){P.n <- 0}
1201 if (is.na(P.y)){P.y <- 0}
1202 if (is.na(C.n)){C.n <- 0}
1203 if (is.na(C.y)){C.y <- 0}
1204 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1205 if (any(c(P.y, P.n, C.y, C.n)==0)){
1206   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1207   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1208                     CImethod='Woolf')$conf.int[1],1),
1209              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1210                     CImethod='Woolf')$conf.int[2],1), sep='-')
1211 }else{
1212   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1213   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1214                                     nrow=2))$conf.int[1],1),
1215              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1216                                     nrow=2))$conf.int[2],1), sep='-')
1217 }
1218 results <- rbind(results, data.frame(Category='',
1219                                     Metadata ="Ulcers in the past 3 months",
1220                                     `PD N`=P.n+P.y,
1221                                     `PD summary stats`=paste(P.y,
1222                                                                " ", "(" ,
1223                                                                round(P.y/(P.n+P.y)*100, 0),
1224                                                                "%", ") ",
1225                                                                sep=""),
1226                                     `NHC N`=C.n+C.y,
1227                                     `NHC summary stats`=paste(C.y,
1228                                                                " ", "(" ,
1229                                                                round(C.y/(C.n+C.y)*100, 1),
1230                                                                "%", ") ",
1231                                                                sep=""),
1232                                     `Total N`=P.n+P.y+C.n+C.y,
1233                                     P=formatC(p, format="e", digits=1),
1234                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
1235                                     check.names=FALSE))
1236 # SIBO
1237 P.y <- table(subset(subject.data, Case_status == "PD")$SIBO) ['Y']
1238 P.n <- table(subset(subject.data, Case_status == "PD")$SIBO) ['N']
1239 C.y <- table(subset(subject.data, Case_status == "Control")$SIBO) ['Y']
1240 C.n <- table(subset(subject.data, Case_status == "Control")$SIBO) ['N']
1241 if (is.na(P.n)){P.n <- 0}
1242 if (is.na(P.y)){P.y <- 0}
1243 if (is.na(C.n)){C.n <- 0}
1244 if (is.na(C.y)){C.y <- 0}
1245 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1246 if (any(c(P.y, P.n, C.y, C.n)==0)){
1247   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1248   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1249                     CImethod='Woolf')$conf.int[1],1),
1250              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1251                     CImethod='Woolf')$conf.int[2],1), sep='-')
1252 }else{
1253   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate

```

```

1254   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1255                                     nrow=2))$conf.int[1],1),
1256              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1257                                     nrow=2))$conf.int[2],1), sep='-')
1258 }
1259 results <- rbind(results, data.frame(Category='',
1260                                     Metadata ="Small instestinal bacterial overgrowth",
1261                                     `PD N`=P.n+P.y,
1262                                     `PD summary stats`=paste(P.y,
1263                                                                " ", "(",
1264                                                                round(P.y/(P.n+P.y)*100, 1),
1265                                                                "%", ")"),
1266                                                                sep=""),
1267                                     `NHC N`=C.n+C.y,
1268                                     `NHC summary stats`=paste(C.y,
1269                                                                " ", "(",
1270                                                                round(C.y/(C.n+C.y)*100, 0),
1271                                                                "%", ")"),
1272                                                                sep=""),
1273                                     `Total N`=P.n+P.y+C.n+C.y,
1274                                     P=formatC(p, format="e", digits=1),
1275                                     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
1276                                     check.names=FALSE))
1277 # Celiac disease
1278 P.y <- table(subset(subject.data, Case_status == "PD")$Celiac_disease)['Y']
1279 P.n <- table(subset(subject.data, Case_status == "PD")$Celiac_disease)['N']
1280 C.y <- table(subset(subject.data, Case_status == "Control")$Celiac_disease)['Y']
1281 C.n <- table(subset(subject.data, Case_status == "Control")$Celiac_disease)['N']
1282 if (is.na(P.n)){P.n <- 0}
1283 if (is.na(P.y)){P.y <- 0}
1284 if (is.na(C.n)){C.n <- 0}
1285 if (is.na(C.y)){C.y <- 0}
1286 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1287 if (any(c(P.y, P.n, C.y, C.n)==0)){
1288   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
1289   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1290                           CImethod='Woolf')$conf.int[1],1),
1291              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1292                           CImethod='Woolf')$conf.int[2],1), sep='-')
1293 }else{
1294   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1295   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1296                                       nrow=2))$conf.int[1],1),
1297              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1298                                       nrow=2))$conf.int[2],1), sep='-')
1299 }
1300 results <- rbind(results, data.frame(Category='',
1301                                     Metadata ="Celiac disease",
1302                                     `PD N`=P.n+P.y,
1303                                     `PD summary stats`=paste(P.y,
1304                                                                " ", "(",
1305                                                                round(P.y/(P.n+P.y)*100, 1),
1306                                                                "%", ")"),
1307                                                                sep=""),

```

```

1308     `NHC N`=C.n+C.y,
1309     `NHC summary stats`=paste(C.y,
1310                               " ", "(",
1311                               round(C.y/(C.n+C.y)*100, 0),
1312                               "%", ")\"",
1313                               sep=""),
1314     `Total N`=P.n+P.y+C.n+C.y,
1315     P=formatC(p, format="e", digits=1),
1316     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
1317     check.names=FALSE))
1318 # GI cancer in past 3 months
1319 P.y <- table(subset(subject.data, Case_status == "PD")$GI_cancer_past_3_months)['Y']
1320 P.n <- table(subset(subject.data, Case_status == "PD")$GI_cancer_past_3_months)['N']
1321 C.y <- table(subset(subject.data, Case_status == "Control")$GI_cancer_past_3_months)['Y']
1322 C.n <- table(subset(subject.data, Case_status == "Control")$GI_cancer_past_3_months)['N']
1323 if (is.na(P.n)){P.n <- 0}
1324 if (is.na(P.y)){P.y <- 0}
1325 if (is.na(C.n)){C.n <- 0}
1326 if (is.na(C.y)){C.y <- 0}
1327 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1328 if (any(c(P.y, P.n, C.y, C.n)==0)){
1329   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
1330   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1331                          CImethod='Wolf')$conf.int[1],1),
1332              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1333                          CImethod='Wolf')$conf.int[2],1), sep='-')
1334 }else{
1335   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1336   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1337                                     nrow=2))$conf.int[1],1),
1338              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1339                                     nrow=2))$conf.int[2],1), sep='-')
1340 }
1341 results <- rbind(results, data.frame(Category='',
1342                                     Metadata="GI cancer in the last 3 months",
1343                                     `PD N`=P.n+P.y,
1344                                     `PD summary stats`=paste(P.y,
1345                                                               " ", "(",
1346                                                               round(P.y/(P.n+P.y)*100, 1),
1347                                                               "%", ")\"",
1348                                                               sep=""),
1349                                     `NHC N`=C.n+C.y,
1350                                     `NHC summary stats`=paste(C.y,
1351                                                               " ", "(",
1352                                                               round(C.y/(C.n+C.y)*100, 0),
1353                                                               "%", ")\"",
1354                                                               sep=""),
1355                                     `Total N`=P.n+P.y+C.n+C.y,
1356                                     P=formatC(p, format="e", digits=1),
1357                                     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
1358                                     check.names=FALSE))
1359 # intestinal disease
1360 P.y <- table(subset(subject.data, Case_status == "PD")$Intestinal_disease)['Y']
1361 P.n <- table(subset(subject.data, Case_status == "PD")$Intestinal_disease)['N']

```

```

1362 C.y <- table(subset(subject.data, Case_status == "Control")$Intestinal_disease)['Y']
1363 C.n <- table(subset(subject.data, Case_status == "Control")$Intestinal_disease)['N']
1364 if (is.na(P.n)){P.n <- 0}
1365 if (is.na(P.y)){P.y <- 0}
1366 if (is.na(C.n)){C.n <- 0}
1367 if (is.na(C.y)){C.y <- 0}
1368 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
1369 if (any(c(P.y, P.n, C.y, C.n)==0)){
1370   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
1371   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1372                       CImethod='Wolf')$conf.int[1],1),
1373              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
1374                       CImethod='Wolf')$conf.int[2],1), sep='-')
1375 }else{
1376   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
1377   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1378                                     nrow=2))$conf.int[1],1),
1379              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
1380                                     nrow=2))$conf.int[2],1), sep='-')
1381 }
1382 results <- rbind(results, data.frame(Category='',
1383                                     Metadata="GI disease (yes to any of the eight items)",
1384                                     `PD N`=P.n+P.y,
1385                                     `PD summary stats`=paste(P.y,
1386                                                                " ", "(",
1387                                                                round(P.y/(P.n+P.y)*100, 0),
1388                                                                "%", ")"),
1389                                                                sep=""),
1390                                     `NHC N`=C.n+C.y,
1391                                     `NHC summary stats`=paste(C.y,
1392                                                                " ", "(",
1393                                                                round(C.y/(C.n+C.y)*100, 0),
1394                                                                "%", ")"),
1395                                                                sep=""),
1396                                     `Total N`=P.n+P.y+C.n+C.y,
1397                                     P=formatC(p, format="e", digits=1),
1398                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
1399                                     check.names=FALSE))

1 # indigestion drugs
2 P.y <- table(subset(subject.data, Case_status == "PD")$Indigestion_drugs)['Y']
3 P.n <- table(subset(subject.data, Case_status == "PD")$Indigestion_drugs)['N']
4 C.y <- table(subset(subject.data, Case_status == "Control")$Indigestion_drugs)['Y']
5 C.n <- table(subset(subject.data, Case_status == "Control")$Indigestion_drugs)['N']
6 if (is.na(P.n)){P.n <- 0}
7 if (is.na(P.y)){P.y <- 0}
8 if (is.na(C.n)){C.n <- 0}
9 if (is.na(C.y)){C.y <- 0}
10 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
11 if (any(c(P.y, P.n, C.y, C.n)==0)){
12   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
13   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
14                       CImethod='Wolf')$conf.int[1],1),
15              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),

```

```

16         CImethod='Woolf')$conf.int[2],1), sep='-')
17 }else{
18   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
19   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
20                                     nrow=2))$conf.int[1],1),
21              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
22                                     nrow=2))$conf.int[2],1), sep='-')
23 }
24 results <- rbind(results,
25   data.frame(Category='Medications (currently taking at time of stool collection, unless noted)',
26             Metadata ="Indigestion drugs",
27             `PD N`=P.n+P.y,
28             `PD summary stats`=paste(P.y,
29                                     " ", "(",
30                                     round(P.y/(P.n+P.y)*100, 0),
31                                     "%", ")",
32                                     sep=""),
33             `NHC N`=C.n+C.y,
34             `NHC summary stats`=paste(C.y,
35                                     " ", "(",
36                                     round(C.y/(C.n+C.y)*100, 0),
37                                     "%", ")",
38                                     sep=""),
39             `Total N`=P.n+P.y+C.n+C.y,
40             P=formatC(p, format="e", digits=1),
41             `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
42             check.names=FALSE))
43 # antibiotics
44 P.y <- table(subset(subject.data, Case_status == "PD")$Antibiotics_current)['Y']
45 P.n <- table(subset(subject.data, Case_status == "PD")$Antibiotics_current)['N']
46 C.y <- table(subset(subject.data, Case_status == "Control")$Antibiotics_current)['Y']
47 C.n <- table(subset(subject.data, Case_status == "Control")$Antibiotics_current)['N']
48 if (is.na(P.n)){P.n <- 0}
49 if (is.na(P.y)){P.y <- 0}
50 if (is.na(C.n)){C.n <- 0}
51 if (is.na(C.y)){C.y <- 0}
52 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
53 if (any(c(P.y, P.n, C.y, C.n)==0)){
54   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
55   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
56                             CImethod='Woolf')$conf.int[1],1),
57              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
58                             CImethod='Woolf')$conf.int[2],1), sep='-')
59 }else{
60   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
61   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
62                                     nrow=2))$conf.int[1],1),
63              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
64                                     nrow=2))$conf.int[2],1), sep='-')
65 }
66 results <- rbind(results, data.frame(Category='',
67                                   Metadata ="Antibiotics",
68                                   `PD N`=P.n+P.y,
69                                   `PD summary stats`=paste(P.y,

```



```

70         " ", "(" ,
71         round(P.y/(P.n+P.y)*100, 0),
72         "%", ")" ,
73         sep=""),
74     `NHC N`=C.n+C.y,
75     `NHC summary stats`=paste(C.y,
76         " ", "(" ,
77         round(C.y/(C.n+C.y)*100, 0),
78         "%", ")" ,
79         sep=""),
80     `Total N`=P.n+P.y+C.n+C.y,
81     P=formatC(p, format="e", digits=1),
82     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']' ,sep=''),
83     check.names=FALSE))
84 # antibiotics in past 3 months
85 P.y <- table(subset(subject.data, Case_status == "PD")$Antibiotics_past_3_months)[ 'Y' ]
86 P.n <- table(subset(subject.data, Case_status == "PD")$Antibiotics_past_3_months)[ 'N' ]
87 C.y <- table(subset(subject.data, Case_status == "Control")$Antibiotics_past_3_months)[ 'Y' ]
88 C.n <- table(subset(subject.data, Case_status == "Control")$Antibiotics_past_3_months)[ 'N' ]
89 if (is.na(P.n)){P.n <- 0}
90 if (is.na(P.y)){P.y <- 0}
91 if (is.na(C.n)){C.n <- 0}
92 if (is.na(C.y)){C.y <- 0}
93 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
94 if (any(c(P.y, P.n, C.y, C.n)==0)){
95     or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
96     ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
97         CImethod='Wolf')$conf.int[1],1),
98         round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
99         CImethod='Wolf')$conf.int[2],1), sep='-')
100 }else{
101     or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
102     ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
103         nrow=2))$conf.int[1],1),
104         round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
105         nrow=2))$conf.int[2],1), sep='-')
106 }
107 results <- rbind(results, data.frame(Category='',
108     Metadata ="Antibiotics in past 3 months",
109     `PD N`=P.n+P.y,
110     `PD summary stats`=paste(P.y,
111         " ", "(" ,
112         round(P.y/(P.n+P.y)*100, 0),
113         "%", ")" ,
114         sep=""),
115     `NHC N`=C.n+C.y,
116     `NHC summary stats`=paste(C.y,
117         " ", "(" ,
118         round(C.y/(C.n+C.y)*100, 0),
119         "%", ")" ,
120         sep=""),
121     `Total N`=P.n+P.y+C.n+C.y,
122     P=formatC(p, format="e", digits=1),
123     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']' ,sep=''),

```

```

124                                     check.names=FALSE))
125 # laxatives
126 P.y <- table(subset(subject.data, Case_status == "PD")$Laxatives)['Y']
127 P.n <- table(subset(subject.data, Case_status == "PD")$Laxatives)['N']
128 C.y <- table(subset(subject.data, Case_status == "Control")$Laxatives)['Y']
129 C.n <- table(subset(subject.data, Case_status == "Control")$Laxatives)['N']
130 if (is.na(P.n)){P.n <- 0}
131 if (is.na(P.y)){P.y <- 0}
132 if (is.na(C.n)){C.n <- 0}
133 if (is.na(C.y)){C.y <- 0}
134 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
135 if (any(c(P.y, P.n, C.y, C.n)==0)){
136   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
137   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
138                       CImethod='Wolf')$conf.int[1],1),
139              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
140                       CImethod='Wolf')$conf.int[2],1), sep='-')
141 }else{
142   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
143   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
144                                   nrow=2))$conf.int[1],1),
145              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
146                                   nrow=2))$conf.int[2],1), sep='-')
147 }
148 results <- rbind(results, data.frame(Category='',
149                                     Metadata ="Laxatives",
150                                     `PD N`=P.n+P.y,
151                                     `PD summary stats`=paste(P.y,
152                                                                " ", "(",
153                                                                round(P.y/(P.n+P.y)*100, 0),
154                                                                "%", ")\"",
155                                                                sep=""),
156                                     `NHC N`=C.n+C.y,
157                                     `NHC summary stats`=paste(C.y,
158                                                                " ", "(",
159                                                                round(C.y/(C.n+C.y)*100, 0),
160                                                                "%", ")\"",
161                                                                sep=""),
162                                     `Total N`=P.n+P.y+C.n+C.y,
163                                     P=formatC(p, format="e", digits=1),
164                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
165                                     check.names=FALSE))
166 # anti-inflammatory drugs
167 P.y <- table(subset(subject.data, Case_status == "PD")$Anti_inflammatory_drugs)['Y']
168 P.n <- table(subset(subject.data, Case_status == "PD")$Anti_inflammatory_drugs)['N']
169 C.y <- table(subset(subject.data, Case_status == "Control")$Anti_inflammatory_drugs)['Y']
170 C.n <- table(subset(subject.data, Case_status == "Control")$Anti_inflammatory_drugs)['N']
171 if (is.na(P.n)){P.n <- 0}
172 if (is.na(P.y)){P.y <- 0}
173 if (is.na(C.n)){C.n <- 0}
174 if (is.na(C.y)){C.y <- 0}
175 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
176 if (any(c(P.y, P.n, C.y, C.n)==0)){
177   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate

```

```

178   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
179                       CImethod='Wolf')$conf.int[1],1),
180              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
181                       CImethod='Wolf')$conf.int[2],1), sep='-')
182 }else{
183   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
184   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
185                                   nrow=2))$conf.int[1],1),
186              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
187                                   nrow=2))$conf.int[2],1), sep='-')
188 }
189 results <- rbind(results, data.frame(Category='',
190                                   Metadata ="Anti-inflammatory drugs",
191                                   `PD N`=P.n+P.y,
192                                   `PD summary stats`=paste(P.y,
193                                                           " ", "(" ,
194                                                           round(P.y/(P.n+P.y)*100, 0),
195                                                           "%", ") ",
196                                                           sep=""),
197                                   `NHC N`=C.n+C.y,
198                                   `NHC summary stats`=paste(C.y,
199                                                           " ", "(" ,
200                                                           round(C.y/(C.n+C.y)*100, 0),
201                                                           "%", ") ",
202                                                           sep=""),
203                                   `Total N`=P.n+P.y+C.n+C.y,
204                                   P=formatC(p, format="e", digits=1),
205                                   `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
206                                   check.names=FALSE))
207 # probiotics
208 P.y <- table(subset(subject.data, Case_status == "PD")$Probiotic)['Y']
209 P.n <- table(subset(subject.data, Case_status == "PD")$Probiotic)['N']
210 C.y <- table(subset(subject.data, Case_status == "Control")$Probiotic)['Y']
211 C.n <- table(subset(subject.data, Case_status == "Control")$Probiotic)['N']
212 if (is.na(P.n)){P.n <- 0}
213 if (is.na(P.y)){P.y <- 0}
214 if (is.na(C.n)){C.n <- 0}
215 if (is.na(C.y)){C.y <- 0}
216 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
217 if (any(c(P.y, P.n, C.y, C.n)==0)){
218   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
219   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
220                           CImethod='Wolf')$conf.int[1],1),
221              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
222                           CImethod='Wolf')$conf.int[2],1), sep='-')
223 }else{
224   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
225   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
226                                   nrow=2))$conf.int[1],1),
227              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
228                                   nrow=2))$conf.int[2],1), sep='-')
229 }
230 results <- rbind(results, data.frame(Category='',
231                                   Metadata ="Probiotics",

```

```

232     `PD N`=P.n+P.y,
233     `PD summary stats`=paste(P.y,
234                               " ", "(",
235                               round(P.y/(P.n+P.y)*100, 0),
236                               "%", ") ",
237                               sep=""),
238     `NHC N`=C.n+C.y,
239     `NHC summary stats`=paste(C.y,
240                               " ", "(",
241                               round(C.y/(C.n+C.y)*100, 0),
242                               "%", ") ",
243                               sep=""),
244     `Total N`=P.n+P.y+C.n+C.y,
245     P=formatC(p, format="e", digits=1),
246     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
247     check.names=FALSE))
248 # radiation or chemotherapy
249 P.y <- table(subset(subject.data, Case_status == "PD")$Radiation_Chemo)['Y']
250 P.n <- table(subset(subject.data, Case_status == "PD")$Radiation_Chemo)['N']
251 C.y <- table(subset(subject.data, Case_status == "Control")$Radiation_Chemo)['Y']
252 C.n <- table(subset(subject.data, Case_status == "Control")$Radiation_Chemo)['N']
253 if (is.na(P.n)){P.n <- 0}
254 if (is.na(P.y)){P.y <- 0}
255 if (is.na(C.n)){C.n <- 0}
256 if (is.na(C.y)){C.y <- 0}
257 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
258 if (any(c(P.y, P.n, C.y, C.n)==0)){
259   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
260   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
261                             CImethod='Wolf')$conf.int[1],1),
262               round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
263                             CImethod='Wolf')$conf.int[2],1), sep='-')
264 }else{
265   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
266   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
267                                       nrow=2))$conf.int[1],1),
268               round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
269                                       nrow=2))$conf.int[2],1), sep='-')
270 }
271 results <- rbind(results, data.frame(Category='',
272                                     Metadata ="Radiation or chemotherapy",
273                                     `PD N`=P.n+P.y,
274                                     `PD summary stats`=paste(P.y,
275                                                               " ", "(",
276                                                               round(P.y/(P.n+P.y)*100, 0),
277                                                               "%", ") ",
278                                                               sep=""),
279                                     `NHC N`=C.n+C.y,
280                                     `NHC summary stats`=paste(C.y,
281                                                               " ", "(",
282                                                               round(C.y/(C.n+C.y)*100, 0),
283                                                               "%", ") ",
284                                                               sep=""),
285                                     `Total N`=P.n+P.y+C.n+C.y,

```

```

286         P=formatC(p, format="e", digits=1),
287         `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
288         check.names=FALSE))
289 # blood thinners
290 P.y <- table(subset(subject.data, Case_status == "PD")$Blood_thinners)['Y']
291 P.n <- table(subset(subject.data, Case_status == "PD")$Blood_thinners)['N']
292 C.y <- table(subset(subject.data, Case_status == "Control")$Blood_thinners)['Y']
293 C.n <- table(subset(subject.data, Case_status == "Control")$Blood_thinners)['N']
294 if (is.na(P.n)){P.n <- 0}
295 if (is.na(P.y)){P.y <- 0}
296 if (is.na(C.n)){C.n <- 0}
297 if (is.na(C.y)){C.y <- 0}
298 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
299 if (any(c(P.y, P.n, C.y, C.n)==0)){
300     or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
301     ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
302                         CImethod='Wolf')$conf.int[1],1),
303                round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
304                         CImethod='Wolf')$conf.int[2],1), sep='-')
305 }else{
306     or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
307     ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
308                                     nrow=2))$conf.int[1],1),
309                round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
310                                     nrow=2))$conf.int[2],1), sep='-')
311 }
312 results <- rbind(results, data.frame(Category='',
313                                     Metadata ="Blood thinners",
314                                     `PD N`=P.n+P.y,
315                                     `PD summary stats`=paste(P.y,
316                                                                " ", "(",
317                                                                round(P.y/(P.n+P.y)*100, 0),
318                                                                "%", ")\"",
319                                                                sep=""),
320                                     `NHC N`=C.n+C.y,
321                                     `NHC summary stats`=paste(C.y,
322                                                                " ", "(",
323                                                                round(C.y/(C.n+C.y)*100, 0),
324                                                                "%", ")\"",
325                                                                sep=""),
326                                     `Total N`=P.n+P.y+C.n+C.y,
327                                     P=formatC(p, format="e", digits=1),
328                                     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
329                                     check.names=FALSE))
330 # cholesterol medication
331 P.y <- table(subset(subject.data, Case_status == "PD")$Cholesterol_med)['Y']
332 P.n <- table(subset(subject.data, Case_status == "PD")$Cholesterol_med)['N']
333 C.y <- table(subset(subject.data, Case_status == "Control")$Cholesterol_med)['Y']
334 C.n <- table(subset(subject.data, Case_status == "Control")$Cholesterol_med)['N']
335 if (is.na(P.n)){P.n <- 0}
336 if (is.na(P.y)){P.y <- 0}
337 if (is.na(C.n)){C.n <- 0}
338 if (is.na(C.y)){C.y <- 0}
339 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value

```

```

340 if (any(c(P.y, P.n, C.y, C.n)==0)){
341   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
342   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
343                     CImethod='Wolf')$conf.int[1],1),
344              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
345                     CImethod='Wolf')$conf.int[2],1), sep='-')
346 }else{
347   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
348   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
349                                     nrow=2))$conf.int[1],1),
350              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
351                                     nrow=2))$conf.int[2],1), sep='-')
352 }
353 results <- rbind(results, data.frame(Category='',
354                                     Metadata ="Cholesterol medication",
355                                     `PD N`=P.n+P.y,
356                                     `PD summary stats`=paste(P.y,
357                                                                " ", "(",
358                                                                round(P.y/(P.n+P.y)*100, 0),
359                                                                "%", ")"),
360                                                                sep=""),
361                                     `NHC N`=C.n+C.y,
362                                     `NHC summary stats`=paste(C.y,
363                                                                " ", "(",
364                                                                round(C.y/(C.n+C.y)*100, 0),
365                                                                "%", ")"),
366                                                                sep=""),
367                                     `Total N`=P.n+P.y+C.n+C.y,
368                                     P=formatC(p, format="e", digits=1),
369                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
370                                     check.names=FALSE))
371 # blood pressure medication
372 P.y <- table(subset(subject.data, Case_status == "PD")$Blood_pressure_med)['Y']
373 P.n <- table(subset(subject.data, Case_status == "PD")$Blood_pressure_med)['N']
374 C.y <- table(subset(subject.data, Case_status == "Control")$Blood_pressure_med)['Y']
375 C.n <- table(subset(subject.data, Case_status == "Control")$Blood_pressure_med)['N']
376 if (is.na(P.n)){P.n <- 0}
377 if (is.na(P.y)){P.y <- 0}
378 if (is.na(C.n)){C.n <- 0}
379 if (is.na(C.y)){C.y <- 0}
380 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
381 if (any(c(P.y, P.n, C.y, C.n)==0)){
382   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
383   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
384                     CImethod='Wolf')$conf.int[1],1),
385              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
386                     CImethod='Wolf')$conf.int[2],1), sep='-')
387 }else{
388   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
389   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
390                                     nrow=2))$conf.int[1],1),
391              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
392                                     nrow=2))$conf.int[2],1), sep='-')
393 }

```

```

394 results <- rbind(results, data.frame(Category='',
395                                     Metadata ="Blood pressure medication",
396                                     `PD N`=P.n+P.y,
397                                     `PD summary stats`=paste(P.y,
398                                                             " ", "(",
399                                                             round(P.y/(P.n+P.y)*100, 0),
400                                                             "%", ")"),
401                                     sep=""),
402                                     `NHC N`=C.n+C.y,
403                                     `NHC summary stats`=paste(C.y,
404                                                             " ", "(",
405                                                             round(C.y/(C.n+C.y)*100, 0),
406                                                             "%", ")"),
407                                     sep=""),
408                                     `Total N`=P.n+P.y+C.n+C.y,
409                                     P=formatC(p, format="e", digits=1),
410                                     `OR [95%CI]`=paste(round(or, 1), ' [' ,ci, ']', sep=''),
411                                     check.names=FALSE))
412 # thyroid medication
413 P.y <- table(subset(subject.data, Case_status == "PD")$Thyroid_med)['Y']
414 P.n <- table(subset(subject.data, Case_status == "PD")$Thyroid_med)['N']
415 C.y <- table(subset(subject.data, Case_status == "Control")$Thyroid_med)['Y']
416 C.n <- table(subset(subject.data, Case_status == "Control")$Thyroid_med)['N']
417 if (is.na(P.n)){P.n <- 0}
418 if (is.na(P.y)){P.y <- 0}
419 if (is.na(C.n)){C.n <- 0}
420 if (is.na(C.y)){C.y <- 0}
421 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
422 if (any(c(P.y, P.n, C.y, C.n)==0)){
423   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
424   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
425                           CImethod='Wolf')$conf.int[1],1),
426              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
427                           CImethod='Wolf')$conf.int[2],1), sep='-')
428 }else{
429   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
430   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
431                                       nrow=2))$conf.int[1],1),
432              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
433                                       nrow=2))$conf.int[2],1), sep='-')
434 }
435 results <- rbind(results, data.frame(Category='',
436                                     Metadata ="Thyroid medication",
437                                     `PD N`=P.n+P.y,
438                                     `PD summary stats`=paste(P.y,
439                                                             " ", "(",
440                                                             round(P.y/(P.n+P.y)*100, 0),
441                                                             "%", ")"),
442                                     sep=""),
443                                     `NHC N`=C.n+C.y,
444                                     `NHC summary stats`=paste(C.y,
445                                                             " ", "(",
446                                                             round(C.y/(C.n+C.y)*100, 0),
447                                                             "%", ")"),

```

```

448                                     sep=""),
449     `Total N`=P.n+P.y+C.n+C.y,
450     P=formatC(p, format="e", digits=1),
451     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
452     check.names=FALSE))
453 # asthma or COPD medication
454 P.y <- table(subset(subject.data, Case_status == "PD")$Asthma_or_COPD_med) ['Y']
455 P.n <- table(subset(subject.data, Case_status == "PD")$Asthma_or_COPD_med) ['N']
456 C.y <- table(subset(subject.data, Case_status == "Control")$Asthma_or_COPD_med) ['Y']
457 C.n <- table(subset(subject.data, Case_status == "Control")$Asthma_or_COPD_med) ['N']
458 if (is.na(P.n)){P.n <- 0}
459 if (is.na(P.y)){P.y <- 0}
460 if (is.na(C.n)){C.n <- 0}
461 if (is.na(C.y)){C.y <- 0}
462 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
463 if (any(c(P.y, P.n, C.y, C.n)==0)){
464   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
465   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
466                         CImethod='Wolf')$conf.int[1],1),
467              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
468                         CImethod='Wolf')$conf.int[2],1), sep='-')
469 }else{
470   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
471   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
472                                     nrow=2))$conf.int[1],1),
473              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
474                                     nrow=2))$conf.int[2],1), sep='-')
475 }
476 results <- rbind(results, data.frame(Category='',
477                                     Metadata ="Asthma or COPD medication",
478                                     `PD N`=P.n+P.y,
479                                     `PD summary stats`=paste(P.y,
480                                                                " ", "(",
481                                                                round(P.y/(P.n+P.y)*100, 0),
482                                                                "%", ")"),
483                                                                sep=""),
484                                     `NHC N`=C.n+C.y,
485                                     `NHC summary stats`=paste(C.y,
486                                                                " ", "(",
487                                                                round(C.y/(C.n+C.y)*100, 0),
488                                                                "%", ")"),
489                                                                sep=""),
490                                     `Total N`=P.n+P.y+C.n+C.y,
491                                     P=formatC(p, format="e", digits=1),
492                                     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']', sep=''),
493                                     check.names=FALSE))
494 # diabetes medication
495 P.y <- table(subset(subject.data, Case_status == "PD")$Diabetes_med) ['Y']
496 P.n <- table(subset(subject.data, Case_status == "PD")$Diabetes_med) ['N']
497 C.y <- table(subset(subject.data, Case_status == "Control")$Diabetes_med) ['Y']
498 C.n <- table(subset(subject.data, Case_status == "Control")$Diabetes_med) ['N']
499 if (is.na(P.n)){P.n <- 0}
500 if (is.na(P.y)){P.y <- 0}
501 if (is.na(C.n)){C.n <- 0}

```



```

502 if (is.na(C.y)){C.y <- 0}
503 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
504 if (any(c(P.y, P.n, C.y, C.n)==0)){
505   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
506   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
507                     CImethod='Woolf')$conf.int[1],1),
508              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
509                     CImethod='Woolf')$conf.int[2],1), sep='-')
510 }else{
511   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
512   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
513                                     nrow=2))$conf.int[1],1),
514              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
515                                     nrow=2))$conf.int[2],1), sep='-')
516 }
517 results <- rbind(results, data.frame(Category='',
518                                     Metadata ="Diabetes medication",
519                                     `PD N`=P.n+P.y,
520                                     `PD summary stats`=paste(P.y,
521                                                                " ", "(",
522                                                                round(P.y/(P.n+P.y)*100, 0),
523                                                                "%", ")\"",
524                                                                sep=""),
525                                     `NHC N`=C.n+C.y,
526                                     `NHC summary stats`=paste(C.y,
527                                                                " ", "(",
528                                                                round(C.y/(C.n+C.y)*100, 0),
529                                                                "%", ")\"",
530                                                                sep=""),
531                                     `Total N`=P.n+P.y+C.n+C.y,
532                                     P=formatC(p, format="e", digits=1),
533                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
534                                     check.names=FALSE))
535 # pain medication
536 P.y <- table(subset(subject.data, Case_status == "PD")$Pain_med)['Y']
537 P.n <- table(subset(subject.data, Case_status == "PD")$Pain_med)['N']
538 C.y <- table(subset(subject.data, Case_status == "Control")$Pain_med)['Y']
539 C.n <- table(subset(subject.data, Case_status == "Control")$Pain_med)['N']
540 if (is.na(P.n)){P.n <- 0}
541 if (is.na(P.y)){P.y <- 0}
542 if (is.na(C.n)){C.n <- 0}
543 if (is.na(C.y)){C.y <- 0}
544 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
545 if (any(c(P.y, P.n, C.y, C.n)==0)){
546   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
547   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
548                     CImethod='Woolf')$conf.int[1],1),
549              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
550                     CImethod='Woolf')$conf.int[2],1), sep='-')
551 }else{
552   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
553   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
554                                     nrow=2))$conf.int[1],1),
555              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),

```

```

556         nrow=2))$conf.int[2],1), sep='-')
557     }
558     results <- rbind(results, data.frame(Category='',
559         Metadata ="Pain medication",
560         `PD N`=P.n+P.y,
561         `PD summary stats`=paste(P.y,
562             " ", "(",
563             round(P.y/(P.n+P.y)*100, 0),
564             "%", ") ",
565             sep=""),
566         `NHC N`=C.n+C.y,
567         `NHC summary stats`=paste(C.y,
568             " ", "(",
569             round(C.y/(C.n+C.y)*100, 0),
570             "%", ") ",
571             sep=""),
572         `Total N`=P.n+P.y+C.n+C.y,
573         P=formatC(p, format="e", digits=1),
574         `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
575         check.names=FALSE))
576     # depression, anxiety, mood medication
577     P.y <- table(subset(subject.data, Case_status == "PD")$Depression_anxiety_mood_med)['Y']
578     P.n <- table(subset(subject.data, Case_status == "PD")$Depression_anxiety_mood_med)['N']
579     C.y <- table(subset(subject.data, Case_status == "Control")$Depression_anxiety_mood_med)['Y']
580     C.n <- table(subset(subject.data, Case_status == "Control")$Depression_anxiety_mood_med)['N']
581     if (is.na(P.n)){P.n <- 0}
582     if (is.na(P.y)){P.y <- 0}
583     if (is.na(C.n)){C.n <- 0}
584     if (is.na(C.y)){C.y <- 0}
585     p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
586     if (any(c(P.y, P.n, C.y, C.n)==0)){
587         or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
588         ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
589             CImethod='Woolf')$conf.int[1],1),
590             round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
591             CImethod='Woolf')$conf.int[2],1), sep='-')
592     }else{
593         or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
594         ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
595             nrow=2))$conf.int[1],1),
596             round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
597             nrow=2))$conf.int[2],1), sep='-')
598     }
599     results <- rbind(results, data.frame(Category='',
600         Metadata ="Depression, anxiety, mood medication",
601         `PD N`=P.n+P.y,
602         `PD summary stats`=paste(P.y,
603             " ", "(",
604             round(P.y/(P.n+P.y)*100, 0),
605             "%", ") ",
606             sep=""),
607         `NHC N`=C.n+C.y,
608         `NHC summary stats`=paste(C.y,
609             " ", "(",

```

```

610                                     round(C.y/(C.n+C.y)*100, 0),
611                                     "%",")",
612                                     sep=""),
613                                     `Total N`=P.n+P.y+C.n+C.y,
614                                     P=formatC(p, format="e", digits=1),
615                                     `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
616                                     check.names=FALSE))
617 # birth control or estrogen (females only)
618 P.y <- table(subset(subject.data,
619   Case_status == "PD" & Sex == "F")$Birth_control_or_estrogen)['Y']
620 P.n <- table(subset(subject.data,
621   Case_status == "PD" & Sex == "F")$Birth_control_or_estrogen)['N']
622 C.y <- table(subset(subject.data,
623   Case_status == "Control" & Sex == "F")$Birth_control_or_estrogen)['Y']
624 C.n <- table(subset(subject.data,
625   Case_status == "Control" & Sex == "F")$Birth_control_or_estrogen)['N']
626 if (is.na(P.n)){P.n <- 0}
627 if (is.na(P.y)){P.y <- 0}
628 if (is.na(C.n)){C.n <- 0}
629 if (is.na(C.y)){C.y <- 0}
630 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
631 if (any(c(P.y, P.n, C.y, C.n)==0)){
632   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
633   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
634     CImethod='Wolf')$conf.int[1],1),
635     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
636     CImethod='Wolf')$conf.int[2],1), sep='-')
637 }else{
638   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
639   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
640     nrow=2))$conf.int[1],1),
641     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
642     nrow=2))$conf.int[2],1), sep='-')
643 }
644 results <- rbind(results, data.frame(Category='',
645   Metadata ="Birth control or estrogen (females)",
646   `PD N`=P.n+P.y,
647   `PD summary stats`=paste(P.y,
648     " ", "(" ,
649     round(P.y/(P.n+P.y)*100, 0),
650     "%",")",
651     sep=""),
652   `NHC N`=C.n+C.y,
653   `NHC summary stats`=paste(C.y,
654     " ", "(" ,
655     round(C.y/(C.n+C.y)*100, 0),
656     "%",")",
657     sep=""),
658   `Total N`=P.n+P.y+C.n+C.y,
659   P=formatC(p, format="e", digits=1),
660   `OR [95%CI]`=paste(round(or, 1), ' [',ci,']',sep=''),
661   check.names=FALSE))
662 # antihistamines
663 P.y <- table(subset(subject.data, Case_status == "PD")$Antihistamines)['Y']

```

```

664 P.n <- table(subset(subject.data, Case_status == "PD")$Antihistamines)['N']
665 C.y <- table(subset(subject.data, Case_status == "Control")$Antihistamines)['Y']
666 C.n <- table(subset(subject.data, Case_status == "Control")$Antihistamines)['N']
667 if (is.na(P.n)){P.n <- 0}
668 if (is.na(P.y)){P.y <- 0}
669 if (is.na(C.n)){C.n <- 0}
670 if (is.na(C.y)){C.y <- 0}
671 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
672 if (any(c(P.y, P.n, C.y, C.n)==0)){
673   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
674   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
675                       CImethod='Wolf')$conf.int[1],1),
676              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
677                       CImethod='Wolf')$conf.int[2],1), sep='-')
678 }else{
679   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
680   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
681                                   nrow=2))$conf.int[1],1),
682              round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
683                                   nrow=2))$conf.int[2],1), sep='-')
684 }
685 results <- rbind(results, data.frame(Category='',
686                                     Metadata ="Antihistamines",
687                                     `PD N`=P.n+P.y,
688                                     `PD summary stats`=paste(P.y,
689                                                                " ", "(",
690                                                                round(P.y/(P.n+P.y)*100, 0),
691                                                                "%", ")\"",
692                                                                sep=""),
693                                     `NHC N`=C.n+C.y,
694                                     `NHC summary stats`=paste(C.y,
695                                                                " ", "(",
696                                                                round(C.y/(C.n+C.y)*100, 0),
697                                                                "%", ")\"",
698                                                                sep=""),
699                                     `Total N`=P.n+P.y+C.n+C.y,
700                                     P=formatC(p, format="e", digits=1),
701                                     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
702                                     check.names=FALSE))
703 # Co Q 10
704 P.y <- table(subset(subject.data, Case_status == "PD")$Co_Q_10)['Y']
705 P.n <- table(subset(subject.data, Case_status == "PD")$Co_Q_10)['N']
706 C.y <- table(subset(subject.data, Case_status == "Control")$Co_Q_10)['Y']
707 C.n <- table(subset(subject.data, Case_status == "Control")$Co_Q_10)['N']
708 if (is.na(P.n)){P.n <- 0}
709 if (is.na(P.y)){P.y <- 0}
710 if (is.na(C.n)){C.n <- 0}
711 if (is.na(C.y)){C.y <- 0}
712 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
713 if (any(c(P.y, P.n, C.y, C.n)==0)){
714   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Wolf')$estimate
715   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
716                       CImethod='Wolf')$conf.int[1],1),
717              round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),

```

```

718         CImethod='Woolf')$conf.int[2],1), sep='-')
719 }else{
720   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
721   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
722         nrow=2))$conf.int[1],1),
723     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
724         nrow=2))$conf.int[2],1), sep='-')
725 }
726 results <- rbind(results, data.frame(Category='',
727     Metadata ="Co-Q 10",
728     `PD N`=P.n+P.y,
729     `PD summary stats`=paste(P.y,
730         " ", "(",
731         round(P.y/(P.n+P.y)*100, 0),
732         "%", ")\"",
733         sep=""),
734     `NHC N`=C.n+C.y,
735     `NHC summary stats`=paste(C.y,
736         " ", "(",
737         round(C.y/(C.n+C.y)*100, 0),
738         "%", ")\"",
739         sep=""),
740     `Total N`=P.n+P.y+C.n+C.y,
741     P=formatC(p, format="e", digits=1),
742     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
743     check.names=FALSE))
744 # sleep aid
745 P.y <- table(subset(subject.data, Case_status == "PD")$Sleep_aid)['Y']
746 P.n <- table(subset(subject.data, Case_status == "PD")$Sleep_aid)['N']
747 C.y <- table(subset(subject.data, Case_status == "Control")$Sleep_aid)['Y']
748 C.n <- table(subset(subject.data, Case_status == "Control")$Sleep_aid)['N']
749 if (is.na(P.n)){P.n <- 0}
750 if (is.na(P.y)){P.y <- 0}
751 if (is.na(C.n)){C.n <- 0}
752 if (is.na(C.y)){C.y <- 0}
753 p <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$p.value
754 if (any(c(P.y, P.n, C.y, C.n)==0)){
755   or <- Prop.or(x=c(P.y,P.n), y=c(C.y,C.n), CImethod='Woolf')$estimate
756   ci <- paste(round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
757     CImethod='Woolf')$conf.int[1],1),
758     round(Prop.or(x=c(P.y,P.n), y=c(C.y,C.n),
759     CImethod='Woolf')$conf.int[2],1), sep='-')
760 }else{
761   or <- fisher.test(matrix(c(P.y,P.n,C.y,C.n), nrow=2))$estimate
762   ci <- paste(round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
763     nrow=2))$conf.int[1],1),
764     round(fisher.test(matrix(c(P.y,P.n,C.y,C.n),
765     nrow=2))$conf.int[2],1), sep='-')
766 }
767 results <- rbind(results, data.frame(Category='',
768     Metadata ="Sleep aid",
769     `PD N`=P.n+P.y,
770     `PD summary stats`=paste(P.y,
771         " ", "(",

```

```

772         round(P.y/(P.n+P.y)*100, 0),
773         "%", ")",
774         sep=""),
775     `NHC N`=C.n+C.y,
776     `NHC summary stats`=paste(C.y,
777         " ", "(",
778         round(C.y/(C.n+C.y)*100, 0),
779         "%", ")",
780         sep=""),
781     `Total N`=P.n+P.y+C.n+C.y,
782     P=formatC(p, format="e", digits=1),
783     `OR [95%CI]`=paste(round(or, 1), ' [', ci, ']', sep=''),
784     check.names=FALSE))
785 # add variable numbers
786 results <- data.frame(Index=c(' ', 1:(nrow(results)-1)), results, check.names=FALSE)
787
788 # write out results
789 # create workbook
790 wb <- createWorkbook()
791 # add worksheet, write data, and format output
792 addWorksheet(wb, 'Subject characteristics')
793 writeData(wb, 'Subject characteristics', results, keepNA=TRUE)
794 setColWidths(wb, 'Subject characteristics', cols=seq_len(ncol(results)),
795             widths=c(10,20,55,10,15,10,15,10,10,12)) ### format cells
796 addStyle(wb, 'Subject characteristics', cols=seq_len(ncol(results)),
797         rows=1:(nrow(results)+1), gridExpand=TRUE, style=center, stack=TRUE)
798 mergeCells(wb, 'Subject characteristics', cols=2, rows=2:4)
799 mergeCells(wb, 'Subject characteristics', cols=2, rows=5:7)
800 mergeCells(wb, 'Subject characteristics', cols=2, rows=8:10)
801 mergeCells(wb, 'Subject characteristics', cols=2, rows=11:18)
802 mergeCells(wb, 'Subject characteristics', cols=2, rows=19:25)
803 mergeCells(wb, 'Subject characteristics', cols=2, rows=26:27)
804 mergeCells(wb, 'Subject characteristics', cols=2, rows=28:36)
805 mergeCells(wb, 'Subject characteristics', cols=2, rows=37:55)
806 addStyle(wb, 'Subject characteristics', cols=seq_len(ncol(results)),
807         rows=1, style=bold, stack=TRUE) ### font
808 addStyle(wb, 'Subject characteristics', cols=seq_len(ncol(results)),
809         rows=c(1,2,(nrow(results)+2)), ### borders
810         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
811 addStyle(wb, 'Subject characteristics', cols=seq_len(ncol(results)),
812         rows=c(5,8,11,19,26,28,37), gridExpand=TRUE,
813         style=horizontal_border_thin, stack=TRUE)
814 # convert numbers from strings back to numbers
815 convertNum(results, wb, 'Subject characteristics', TRUE)
816 # save workbook
817 saveWorkbook(wb, 'PDSHOTGUNAnalysis_out/1.Metadata/Subject_characteristics_PDvsNHC.xlsx',
818             overwrite=TRUE)

```

# Analyses of species and genera

## Preparing relative abundance and count data for downstream analyses

```
1  ##### PREPARE RELATIVE ABUNDANCE AND COUNT DATA #####
2
3  # read in metadata
4  metadata <- data.frame(read_xlsx('Source_Data.xlsx', sheet='subject_metadata'))
5  rownames(metadata) <- metadata$sample_name
6
7  # read in tables that were previously generated by taxonomic profiling
8  abund <- data.frame(read_xlsx('Source_Data.xlsx', sheet='metaphlan_counts'))
9
10 ra <- data.frame(read_xlsx('Source_Data.xlsx', sheet='metaphlan_rel_ab'))
11
12 # order same as metadata
13 abund <- abund[,c('clade_name',metadata$sample_name)]
14
15 ra <- ra[,c('clade_name',metadata$sample_name)]
16
17 # make table sample x feature
18 rownames(abund) <- abund$clade_name
19 abund <- data.frame(t(abund[,-1]), check.names=FALSE)
20
21 rownames(ra) <- ra$clade_name
22 ra <- data.frame(t(ra[,-1]), check.names=FALSE)
23
24 # compile count data into phyloseq objects for species and genus taxonomic levels
25 abund.sub <- abund[,grep("s_|UNKNOWN", colnames(abund))]
26 abund.ps.s <- phyloseq(otu_table(as.matrix(abund.sub), taxa_are_rows=FALSE),
27                               sample_data(metadata),
28                               tax_table(as.matrix(
29                                   data.frame(Kingdom=sapply(strsplit(colnames(abund.sub), "\\|"),
30                                                           function(x){x[1]}),
31                                   Phylum=sapply(strsplit(colnames(abund.sub), "\\|"),
32                                                           function(x){x[2]}),
33                                   Class=sapply(strsplit(colnames(abund.sub), "\\|"),
34                                                           function(x){x[3]}),
35                                   Order=sapply(strsplit(colnames(abund.sub), "\\|"),
36                                                           function(x){x[4]}),
37                                   Family=sapply(strsplit(colnames(abund.sub), "\\|"),
38                                                           function(x){x[5]}),
39                                   Genus=sapply(strsplit(colnames(abund.sub), "\\|"),
40                                                           function(x){x[6]}),
41                                   Species=sapply(strsplit(colnames(abund.sub), "\\|"),
42                                                           function(x){x[7]}),
43                                   check.names=FALSE, row.names=colnames(abund.sub))))))
44 abund.sub <- abund[,intersect(grep("s_|UNKNOWN", colnames(abund), invert=TRUE),
45                               grep("g_|UNKNOWN", colnames(abund)))]
46 abund.ps.g <- phyloseq(otu_table(as.matrix(abund.sub), taxa_are_rows=FALSE),
47                               sample_data(metadata),
48                               tax_table(as.matrix(
49                                   data.frame(Kingdom=sapply(strsplit(colnames(abund.sub), "\\|"),
```

```

50         function(x){x[1]}),
51     Phylum=sapply(strsplit(colnames(abun.sub), "\\|"),
52         function(x){x[2]}),
53     Class=sapply(strsplit(colnames(abun.sub), "\\|"),
54         function(x){x[3]}),
55     Order=sapply(strsplit(colnames(abun.sub), "\\|"),
56         function(x){x[4]}),
57     Family=sapply(strsplit(colnames(abun.sub), "\\|"),
58         function(x){x[5]}),
59     Genus=sapply(strsplit(colnames(abun.sub), "\\|"),
60         function(x){x[6]}),
61     Species=sapply(strsplit(colnames(abun.sub), "\\|"),
62         function(x){x[7]}),
63     check.names=FALSE, row.names=colnames(abun.sub))))
64
65 # compile relative abundance data into phyloseq objects for species and
66 # genus taxonomic levels
67 ra.sub <- ra[,grep("s_|UNKNOWN", colnames(ra))]
68 ra.ps.s <- phyloseq(otu_table(as.matrix(ra.sub), taxa_are_rows=FALSE),
69     sample_data(metadata),
70     tax_table(as.matrix(
71     data.frame(Kingdom=sapply(strsplit(colnames(ra.sub), "\\|"),
72         function(x){x[1]}),
73     Phylum=sapply(strsplit(colnames(ra.sub), "\\|"),
74         function(x){x[2]}),
75     Class=sapply(strsplit(colnames(ra.sub), "\\|"),
76         function(x){x[3]}),
77     Order=sapply(strsplit(colnames(ra.sub), "\\|"),
78         function(x){x[4]}),
79     Family=sapply(strsplit(colnames(ra.sub), "\\|"),
80         function(x){x[5]}),
81     Genus=sapply(strsplit(colnames(ra.sub), "\\|"),
82         function(x){x[6]}),
83     Species=sapply(strsplit(colnames(ra.sub), "\\|"),
84         function(x){x[7]}),
85     check.names=FALSE, row.names=colnames(ra.sub))))))
86 ra.sub <- ra[,intersect(grep("s_", colnames(ra), invert=TRUE),
87     grep("g_|UNKNOWN", colnames(ra)))]
88 ra.ps.g <- phyloseq(otu_table(as.matrix(ra.sub), taxa_are_rows=FALSE),
89     sample_data(metadata),
90     tax_table(as.matrix(
91     data.frame(Kingdom=sapply(strsplit(colnames(ra.sub), "\\|"),
92         function(x){x[1]}),
93     Phylum=sapply(strsplit(colnames(ra.sub), "\\|"),
94         function(x){x[2]}),
95     Class=sapply(strsplit(colnames(ra.sub), "\\|"),
96         function(x){x[3]}),
97     Order=sapply(strsplit(colnames(ra.sub), "\\|"),
98         function(x){x[4]}),
99     Family=sapply(strsplit(colnames(ra.sub), "\\|"),
100         function(x){x[5]}),
101     Genus=sapply(strsplit(colnames(ra.sub), "\\|"),
102         function(x){x[6]}),
103     Species=sapply(strsplit(colnames(ra.sub), "\\|"),

```



```

104         function(x){x[7]}),
105         check.names=FALSE, row.names=colnames(ra.sub))))))

```

## Principal Component Analysis

To observe inter-sample differences in gut microbiome compositions (beta-diversity), principal component analysis (PCA) was performed to visually inspect differences in microbiome compositions between samples.

- To perform the PCA, species counts from MetaPhlAn were transformed using the clr transformation (formula:  $\log(x+1) - \text{mean}(\log(x+1))$  where  $x$  is a vector of all the species counts for a sample). PCA was then performed using the `prcomp` function with default parameters. PC1 and PC2 were then plotted with convex hull areas for each group using the `autoplot` function from `ggfortify`.

```

1  ##### PERFORM PCA #####
2
3  # make sure taxa that are all 0 are removed, then transform abundances to clr
4  abund.clr <- transform_sample_counts(filter_taxa(abund.ps.s, function(x){sum(x>0)>0}, TRUE),
5                                     function(x){log(x+1)-mean(log(x+1))})
6  abund.clr <- prune_taxa(taxa_names(abund.clr)[taxa_names(abund.clr) != 'UNKNOWN'], abund.clr)
7
8  # perform PCA
9  pca <- prcomp(otu_table(abund.clr))
10
11 # plot PC1 and PC2 coloring by case status
12 suppress(
13 g <- autoplot(pca, data=data.frame(sample_data(abund.clr)), colour='Case_status',
14             shape='Case_status', scale=FALSE, frame=TRUE) +
15   theme_bw() +
16   scale_colour_manual(labels=c('NHC', 'PD'), values=c("#E69F00", "#00BFC4")) +
17   scale_fill_manual(labels=c('NHC', 'PD'), values=c("#E69F00", "#00BFC4")) +
18   scale_shape_manual(labels=c('NHC', 'PD'), values=c(17,16)) +
19   labs(fill="Case status", color="Case status", shape="Case status")
20 )
21 ggsave('PDSHOTGUNAnalysis_out/2.Gut_microbiome_composition/PCA_case_status.pdf',
22        g, device='pdf', width=5, height=5)

```

- To observe the influence of rarer species on the PCA, PCA was performed and PC 1 and 2 plotted again after excluding species that were detected in <5% of samples.

```

1  ##### PERFORM PCA EXCLUDING RARE SPECIES #####
2
3  # remove taxa that are found in <5% of samples, then transform abundances to clr
4  abund.clr.filt <- transform_sample_counts(
5     filter_taxa(abund.ps.s,
6                 function(x){sum(x>0)>(0.05*nsamples(abund.ps.s))}, TRUE),
7     function(x){log(x+1)-mean(log(x+1))})
8  abund.clr.filt <- prune_taxa(
9     taxa_names(abund.clr.filt)[taxa_names(abund.clr.filt) != 'UNKNOWN'],
10    abund.clr.filt)
11
12 # perform PCA
13 pca <- prcomp(otu_table(abund.clr.filt))
14
15 # plot PC1 and PC2 coloring by case status
16 suppress(

```

```

17 g <- autoplot(pca, data=data.frame(sample_data(abun.clr.filt)), colour='Case_status',
18             shape='Case_status', scale=FALSE, frame=TRUE) +
19   theme_bw() +
20   scale_colour_manual(labels=c('NHC', 'PD'), values=c("#E69F00", "#00BFC4")) +
21   scale_fill_manual(labels=c('NHC', 'PD'), values=c("#E69F00", "#00BFC4")) +
22   scale_shape_manual(labels=c('NHC', 'PD'), values=c(17,16)) +
23   labs(fill="Case status", color="Case status", shape="Case status")
24 )
25 ggsave('PDSHOTGUNAnalysis_out/2.Gut_microbiome_composition/PCA_case_status_filtered.pdf',
26       g, device='pdf', width=5, height=5)

```

## PERMANOVA and PERMDISP

To test if case status significantly associates with inter-sample variation in microbiome compositions (beta-diversity), permutational multivariate analysis of variance (PERMANOVA) was performed.

- PERMANOVA was performed using the function `adonis2` from `vegan` adjusting for stool sampling method, and total sequence count per sample (standardized using the `scale` function). All variables were adjusted for one another in a marginal model by setting `by='margin'`.
- To test if significant results of PERMANOVA were due to differences in heterogeneity of dispersions between groups, a permutation-based test of multivariate homogeneity of group dispersions (PERMDISP) was performed using the `betadisper` (setting `type='median'`) and `permutest` functions from `vegan` to perform the test.
- Aitchison distance (Euclidean distance of clr transformed data) was used as the distance matrix outcome for both PERMANOVA and PERMDISP (calculated using `vegdist` from `vegan` specifying `method='euclidean'`).
- Significance of permutational tests were determined using 9999 permutations.
- PERMANOVA and PERMDISP were performed once with all species and again for species found in >5% of samples.

```

1  ##### PERFORM PERMANOVA & PERMDISP #####
2
3  # standardize total sequence count
4  sample_data(abun.clr)$seqs_scaled <- scale(sample_data(abun.clr)$total_sequences)
5
6  # calculate euclidean distances on clr transformed data (Aitchison distances)
7  aitch.dist <- vegdist(otu_table(abun.clr), method='euclidean')
8
9  # run adonis2 (PERMANOVA) marginal model and 9,999 permutations
10 set.seed(1234)
11 fit <- adonis2(aitch.dist ~ Case_status + seqs_scaled + collection_method,
12              data=data.frame(sample_data(abun.clr)), by='margin', perm=9999)
13
14 # run betadisper and permutest (PERMDISP) with euclidean distance
15 set.seed(1234)
16 disp <- permutest(betadisper(aitch.dist,
17                            sample_data(abun.clr)$Case_status,
18                            type='median'),
19                  permutations=9999)
20 disp.r2 <- disp$tab['Groups', 'Sum Sq']/(disp$tab['Groups', 'Sum Sq'] +
21                                       disp$tab['Residuals', 'Sum Sq'])
22
23 ##### PERFORM PERMANOVA & PERMDISP EXCLUDING RARE SPECIES #####
24

```

```

25 # standardize total sequence count
26 sample_data(abun.clr.filt)$seqs_scaled <- scale(sample_data(abun.clr.filt)$total_sequences)
27
28 # calculate euclidean distances on clr transformed data (Aitchison distances)
29 aitch.dist <- vegdist(otu_table(abun.clr.filt), method='euclidean')
30
31 # run adonis2 (PERMANOVA) marginal model and 9,999 permutations
32 set.seed(1234)
33 fit.filt <- adonis2(aitch.dist ~ Case_status + seqs_scaled + collection_method,
34                   data=data.frame(sample_data(abun.clr.filt)), by='margin', perm=9999)
35
36 # run betadisper and permutest (PERMDISP) with euclidean distance
37 set.seed(1234)
38 disp.filt <- permutest(betadisper(aitch.dist,
39                               sample_data(abun.clr.filt)$Case_status,
40                               type='median'),
41                       permutations=9999)
42 disp.r2.filt <- disp.filt$tab['Groups','Sum Sq']/(disp.filt$tab['Groups','Sum Sq']+
43                                                  disp.filt$tab['Residuals','Sum Sq'])
44
45 # coalesce the results
46 results <- data.frame(
47   `Included species`=c('all detected species',' ',' '),
48   Variable=c('case status','sequence depth (standardized)','collection method'),
49   `PERMANOVA Results`=c(paste('R2=',round(fit$R2[1],3),', P<',
50                               formatC(fit$`Pr(>F)`[1],
51                                       format='e',digits=0),sep=' '),
52                         paste('R2=',round(fit$R2[2],3),', P<',
53                               formatC(fit$`Pr(>F)`[2],
54                                       format='e',digits=0),sep=' '),
55                         paste('R2=',round(fit$R2[3],3),', P=',
56                               formatC(fit$`Pr(>F)`[3],
57                                       format='e',digits=0),sep=' ')),
58   `PERMDISP Results`=c(paste('R2=',round(disp.r2,3),', P<',
59                               formatC(disp$tab$`Pr(>F)`[1],
60                                       format='e',digits=0),sep=' '),
61                         '-',' '),
62   check.names=FALSE)
63
64 results <- rbind(results, data.frame(
65   `Included species`=c('species in >5% samples',' ',' '),
66   Variable=c('case status','sequence depth (standardized)','collection method'),
67   `PERMANOVA Results`=c(paste('R2=',round(fit.filt$R2[1],3),', P<',
68                               formatC(fit.filt$`Pr(>F)`[1],
69                                       format='e',digits=0),sep=' '),
70                         paste('R2=',round(fit.filt$R2[2],3),', P<',
71                               formatC(fit.filt$`Pr(>F)`[2],
72                                       format='e',digits=0),sep=' '),
73                         paste('R2=',round(fit.filt$R2[3],3),', P=',
74                               round(fit.filt$`Pr(>F)`[3],2),sep=' ')),
75   `PERMDISP Results`=c(paste('R2=',round(disp.r2.filt,3),', P<',
76                               formatC(disp.filt$tab$`Pr(>F)`[1],
77                                       format='e',digits=0),sep=' '),
78                         '-',' '),

```

```

79     check.names=FALSE))
80
81 # write results
82 # create workbook
83 wb <- createWorkbook()
84 # add worksheet, write data, and format output
85 addWorksheet(wb, 'PERMANOVA PERMDISP')
86 writeData(wb, 'PERMANOVA PERMDISP', results, keepNA=TRUE)
87 setColWidths(wb, 'PERMANOVA PERMDISP', cols=seq_len(ncol(results)),
88             widths=rep(20,ncol(results))) ### format cells
89 addStyle(wb, 'PERMANOVA PERMDISP', cols=seq_len(ncol(results)),
90         rows=1:(nrow(results)+1), gridExpand=TRUE, style=center, stack=TRUE)
91 addStyle(wb, 'PERMANOVA PERMDISP', cols=seq_len(ncol(results)),
92         rows=1, style=bold, stack=TRUE) ### font
93 addStyle(wb, 'PERMANOVA PERMDISP', cols=seq_len(ncol(results)),
94         rows=c(1,2,(nrow(results)+2)), ### borders
95         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
96 # save workbook
97 saveWorkbook(wb,
98             'PDSshotgunAnalysis_out/2.Gut_microbiome_composition/PERMANOVA_PERMDISP_PDvsNHC.xlsx',
99             overwrite=TRUE)

```

## Enterotype analysis: PD vs NHC

To determine if PD patients had a different distribution of enterotype frequencies than NHC, enterotype profiling was performed using the web-based [EMBL enterotype classifier](#), then differences in overall enterotype frequencies between PD and NHC were tested. The enterotype classifier uses the original enterotype definitions, classifying each sample as the enterotype *Bacteroides*, *Firmicutes*, or *Prevotella*.

- To perform enterotype profiling, the MetaPhlan relative abundance file was subsetted for only genus level entries, then uploaded to the web-based EMBL enterotype classifier. The raw results were downloaded (see Source Data file for enterotype designations) and used to perform analyses, and generate a mosaic plot. Only subjects that were detected by the EMBL classifier as compositionally similar to the training data used to build the classifier (`Within_ET_space` is `TRUE`) were used for analyses (N = 450; 284 PD, 166 NHC).
- Differences in enterotype frequencies between PD and NHC were tested using the `chisq.test` function to perform Pearson's Chi-squared test.
- Relative predispositional effect (RPE) of enterotypes was then investigated to determine what enterotype(s) were driving the difference between PD vs NHC.
- Odds ratios and corresponding significance for the effect driving enterotype(s) were calculated using Fisher's exact test via the `fisher.test` function.

```

1  ##### ENTEROTYPE ANALYSIS #####
2
3  # read in enterotype profiles
4  et <- data.frame(read_xlsx('Source_Data.xlsx', sheet='enterotypes', skip=2))
5
6  # make a quick column for case status
7  et$case_status <- NA
8  et$case_status[grepl('P', et$sample_name)] <- "PD"
9  et$case_status[grepl('C', et$sample_name)] <- "NHC"
10
11 # subset for samples with 'Within_ET_space' equal to TRUE
12 et <- et[et$Within_ET_space == TRUE,]

```

```

13
14 # perform chi-squared test for overall difference in enterotype distribution
15 x2.test.res <- chisq.test(table(et$case_status, et$ET))
16
17 ##### RPE #####
18
19 ### round 1 ###
20 # calculate chi-squared statistics for each PD enterotype using
21 # NHC frequencies to calculate expected values
22 O <- table(et$case_status, et$ET)['PD','ET_B']
23 E <- sum(table(et$case_status, et$ET)['PD',])*
24     (table(et$case_status, et$ET)['NHC','ET_B']/
25     sum(table(et$case_status, et$ET)['NHC',]))
26 x2.ET_B <- (O - E)^2/E
27
28 O <- table(et$case_status, et$ET)['PD','ET_F']
29 E <- sum(table(et$case_status, et$ET)['PD',])*
30     (table(et$case_status, et$ET)['NHC','ET_F']/
31     sum(table(et$case_status, et$ET)['NHC',]))
32 x2.ET_F <- (O - E)^2/E
33
34 O <- table(et$case_status, et$ET)['PD','ET_P']
35 E <- sum(table(et$case_status, et$ET)['PD',])*
36     (table(et$case_status, et$ET)['NHC','ET_P']/
37     sum(table(et$case_status, et$ET)['NHC',]))
38 x2.ET_P <- (O - E)^2/E
39
40 x2.list <- c(x2.ET_B, x2.ET_F, x2.ET_P)
41
42 # calculate total chi-squared statistic and p-value
43 x2.1 <- sum(x2.list)
44 x2.p1 <- pchisq(q=x2.1, df=length(x2.list)-1, lower.tail=FALSE)
45
46 ### round 2 ###
47 # calculate chi-squared statistics for each PD enterotype using
48 # NHC frequencies to calculate expected values (after removing Firmicutes
49 # enterotype that had max chi-squared statistic from last round)
50 O <- table(et$case_status, et$ET)['PD','ET_B']
51 E <- sum(table(et$case_status, et$ET)['PD',c('ET_B','ET_P')])*
52     (table(et$case_status, et$ET)['NHC','ET_B']/
53     sum(table(et$case_status, et$ET)['NHC',c('ET_B','ET_P')]))
54 x2.ET_B <- (O - E)^2/E
55
56 O <- table(et$case_status, et$ET)['PD','ET_P']
57 E <- sum(table(et$case_status, et$ET)['PD',c('ET_B','ET_P')])*
58     (table(et$case_status, et$ET)['NHC','ET_P']/
59     sum(table(et$case_status, et$ET)['NHC',c('ET_B','ET_P')]))
60 x2.ET_P <- (O - E)^2/E
61
62 x2.list <- c(x2.ET_B, x2.ET_P)
63
64 # calculate total chi-squared statistic and p-value
65 x2.2 <- sum(x2.list, na.rm=TRUE)
66 x2.p2 <- pchisq(q=x2.2, df=length(x2.list)-1, lower.tail=FALSE)

```

```

67
68 ##### RPE END #####
69
70 # calculate odds ratio and significance for individual enterotypes
71 fisher.ET_F <- fisher.test(table(et$case_status,
72                               dplyr::recode(et$ET, ET_F='1', ET_B='0', ET_P='0')))
73 fisher.ET_P <- fisher.test(table(et$case_status,
74                               dplyr::recode(et$ET, ET_F='0', ET_P='1', ET_B='0')))
75 fisher.ET_P2 <- fisher.test(table(et$case_status[et$ET != 'ET_F'],
76                               dplyr::recode(et$ET[et$ET != 'ET_F'], ET_P='1', ET_B='0')))
77
78 # coalesce results
79 results <- data.frame(`Case status`=names(rev(table(et$case_status))),
80                       `N Total`=as.vector(rev(table(et$case_status))),
81                       `N Bacteroides`=c(paste(table(et$case_status, et$ET)[2,1], ' (',
82                                               round(table(et$case_status, et$ET)[2,1]/
83                                                       sum(table(et$case_status, et$ET)[2,])*100,0),
84                                               '%)', sep=' '),
85                                           paste(table(et$case_status, et$ET)[1,1], ' (',
86                                               round(table(et$case_status, et$ET)[1,1]/
87                                                       sum(table(et$case_status, et$ET)[1,])*100,0),
88                                               '%)', sep=' ')),
89                       `N Firmicutes`=c(paste(table(et$case_status, et$ET)[2,2], ' (',
90                                               round(table(et$case_status, et$ET)[2,2]/
91                                                       sum(table(et$case_status, et$ET)[2,])*100,0),
92                                               '%)', sep=' '),
93                                           paste(table(et$case_status, et$ET)[1,2], ' (',
94                                               round(table(et$case_status, et$ET)[1,2]/
95                                                       sum(table(et$case_status, et$ET)[1,])*100,0),
96                                               '%)', sep=' ')),
97                       `N Prevotella`=c(paste(table(et$case_status, et$ET)[2,3], ' (',
98                                               round(table(et$case_status, et$ET)[2,3]/
99                                                       sum(table(et$case_status, et$ET)[2,])*100,0),
100                                              '%)', sep=' '),
101                                           paste(table(et$case_status, et$ET)[1,3], ' (',
102                                               round(table(et$case_status, et$ET)[1,3]/
103                                                       sum(table(et$case_status, et$ET)[1,])*100,0),
104                                              '%)', sep=' ')),
105                       `PD vs NHC`=c('',
106                                       paste('X2=', round(x2.test.res$statistic, 1),
107                                             ', P=', formatC(x2.test.res$p.value, format='e', digits=0),
108                                             sep='')),
109                       `PD observed vs expected`=c('',
110                                                   paste('X2=',
111                                                       round(x2.1, 1),
112                                                       ', P=',
113                                                       formatC(x2.p1, format='e', digits=0),
114                                                       sep='')),
115                       `PD observed vs expected (no Firmicutes)`=c('',
116                                                                     paste('X2=',
117                                                       round(x2.2, 1),
118                                                       ', P=',
119                                                       round(x2.p2, 2),
120                                                       sep='')),

```

```

121         `Odds ratio for Firmicutes`=c('',
122                                     paste('OR[95%CI]=' ,
123                                             round(fisher.ET_F$estimate,1),
124                                             '[' ,
125                                             round(fisher.ET_F$conf.int[1],1),
126                                             '-' ,
127                                             round(fisher.ET_F$conf.int[2],1),
128                                             ']' ,
129                                             ', P=' ,
130                                             formatC(fisher.ET_F$p.value,
131                                                       format='e',digits=0),
132                                             sep=' ')),
133         `Odds ratio for Prevotella`=c('',
134                                     paste('OR[95%CI]=' ,
135                                             round(fisher.ET_P$estimate,1),
136                                             '[' ,
137                                             round(fisher.ET_P$conf.int[1],1),
138                                             '-' ,
139                                             round(fisher.ET_P$conf.int[2],1),
140                                             ']' ,
141                                             ', P=' ,
142                                             round(fisher.ET_P$p.value,2),
143                                             sep=' ')),
144         `Odds ratio for Prevotella (no Firmicutes)`=c('',
145                                     paste('OR[95%CI]=' ,
146                                             round(fisher.ET_P2$estimate,1),
147                                             '[' ,
148                                             round(fisher.ET_P2$conf.int[1],1),
149                                             '-' ,
150                                             round(fisher.ET_P2$conf.int[2],1),']' ,
151                                             ', P=' ,
152                                             round(fisher.ET_P2$p.value,2),
153                                             sep=' ')),
154         check.names=FALSE)
155
156 # write results
157 # create workbook
158 wb <- createWorkbook()
159 # add worksheet, write data, and format output
160 addWorksheet(wb, 'Enterotype results')
161 writeData(wb, 'Enterotype results', results, keepNA=TRUE)
162 setColWidths(wb, 'Enterotype results', cols=seq_len(ncol(results)),
163              widths=c(rep(15,8),rep(26,3))) ### format cells
164 addStyle(wb, 'Enterotype results', cols=seq_len(ncol(results)),
165          rows=1:(nrow(results)+1), gridExpand=TRUE, style=center, stack=TRUE)
166 addStyle(wb, 'Enterotype results', cols=seq_len(ncol(results)),
167          rows=1, style=bold, stack=TRUE) ### font
168 addStyle(wb, 'Enterotype results', cols=seq_len(ncol(results)),
169          rows=c(1,2,(nrow(results)+2)), ### borders
170          gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
171 # save workbook
172 saveWorkbook(wb,
173              'PDSHOTGUNAnalysis_out/2.Gut_microbiome_composition/Enterotype_results.xlsx',
174              overwrite=TRUE)

```

```

175
176 # prep data for mosaic plot
177 counts <- table(dplyr::recode(et$case_status, PD=1L, NHC=2L), et$ET)
178 rownames(counts) <- c("PD", "NHC")
179 colnames(counts) <- c("Bacteroides", "Firmicutes", "Prevotella")
180 dimnames(counts) <- list(Case_status=c(paste("PD (N=",results$`N Total`[1],")",sep=""),
181                                     paste("NHC (N=",results$`N Total`[2],")",sep="")),
182                          Enterotype=c("Bacteroides", "Firmicutes", "Prevotella"))
183 percents <- rbind(paste(round(table(et$case_status, et$ET)[2,]/
184                             sum(table(et$case_status, et$ET)[2,])*100,0),'%',sep=''),
185                  paste(round(table(et$case_status, et$ET)[1,]/
186                             sum(table(et$case_status, et$ET)[1,])*100,0),'%',sep=''))
187 rownames(percents) <- c("PD", "NHC")
188 colnames(percents) <- c("Bacteroides", "Firmicutes", "Prevotella")
189 dimnames(percents) <- list(Case_status=c(paste("PD (N=",results$`N Total`[1],")",sep=""),
190                                     paste("NHC (N=",results$`N Total`[2],")",sep="")),
191                          Enterotype=c("Bacteroides", "Firmicutes", "Prevotella"))
192
193 # create mosaic plot
194 pdf('PDSHOTGUNAnalysis_out/2.Gut_microbiome_composition/Enterotype_mosaic_plot.pdf',
195     height=3, width=10)
196 mosaic(counts, main=NULL, highlighting='Enterotype',
197        highlighting_fill=c('grey50','dodgerblue3','firebrick'),
198        spacing=spacing_equal(sp=0.5), margins=c(2,2,2,7),
199        labeling=labeling_border(varname=FALSE, rot_labels=0,
200        just_labels=c('center','center','center','right')),
201        keep_aspect_ratio=FALSE, pop=FALSE)
202 labeling_cells(text=percents, gp_text=gpar(col='white'), margin=0)(counts)
203 trash <- dev.off()

```

## Differential abundance of species and genera

### MWAS

To determine what species and genera are differentially abundant between PD and NHC samples, differential abundance analysis was performed using two methods: 1) ANCOM-BC with count data [relative abundance with unknown estimation x total reads] and 2) linear regression with log<sub>2</sub> transformed relative abundances (without unknown estimation) as implemented in MaAsLin2.

- To perform differential abundance analysis using ANCOM-BC with counts, counts were used as input for the `ancombc` function of the ANCOMBC R package. The ANCOM-BC formula included case status (PD vs NHC), collection method (swab vs OMNIgene GUT kit), and total sequence count (taken from `#nread` line of bowtie2 intermediate files produced by MetaPhlAn) standardized using the `scale` function in R with default parameters. All parameters were left as default except for the FDR adjustment which was made to be the Benjamini-Hochberg (BH) method, and the `zero_cut` which was made 0.95 to make the effective sample size for analysis 37 samples.
- To perform differential abundance analysis using linear regression with log<sub>2</sub> transformed relative abundances, relative abundances from MetaPhlAn were divided by 100 to convert to proportions and used as input to the `Maaslin2` function. All parameters were left as default except for `min_prevalence` which was set to 0.05 to make the effective sample size 37, `normalization` which was set to NONE as we are already inputting relative abundances, `standardize` which was set to FALSE as this is being done prior to MaAsLin2, and `max_significance` which was set to 0.05. The MaAsLin2 `fixed_effects` model included case status (PD vs NHC), collection method (swab vs OMNIgene GUT kit), and total sequence



count (taken from #nread line of bowtie2 intermediate files produced by MetaPhlAn) standardized using the scale function.

```

1  ##### SPECIES AND GENUS MWAS #####
2
3  # recode categorical variables to get correct effect direction and scale numeric data
4  sample_data(abun.ps.s)$Case_status <- dplyr::recode(sample_data(abun.ps.s)$Case_status,
5                                                    PD=1, Control=0)
6  sample_data(abun.ps.g)$Case_status <- dplyr::recode(sample_data(abun.ps.g)$Case_status,
7                                                    PD=1, Control=0)
8  sample_data(abun.ps.s)$collection_method <- dplyr::recode(sample_data(abun.ps.s)$collection_method,
9                                                            swab=1, `OMNIGene GUT`=0)
10 sample_data(abun.ps.g)$collection_method <- dplyr::recode(sample_data(abun.ps.g)$collection_method,
11                                                            swab=1, `OMNIGene GUT`=0)
12 sample_data(abun.ps.s)$seqs_scaled <- scale(sample_data(abun.ps.s)$total_sequences)
13 sample_data(abun.ps.g)$seqs_scaled <- scale(sample_data(abun.ps.g)$total_sequences)
14
15 sample_data(ra.ps.s)$Case_status <- dplyr::recode(sample_data(ra.ps.s)$Case_status,
16                                                    PD=1, Control=0)
17 sample_data(ra.ps.g)$Case_status <- dplyr::recode(sample_data(ra.ps.g)$Case_status,
18                                                    PD=1, Control=0)
19 sample_data(ra.ps.s)$collection_method <- dplyr::recode(sample_data(ra.ps.s)$collection_method,
20                                                            swab=1, `OMNIGene GUT`=0)
21 sample_data(ra.ps.g)$collection_method <- dplyr::recode(sample_data(ra.ps.g)$collection_method,
22                                                            swab=1, `OMNIGene GUT`=0)
23 sample_data(ra.ps.s)$seqs_scaled <- scale(sample_data(ra.ps.s)$total_sequences)
24 sample_data(ra.ps.g)$seqs_scaled <- scale(sample_data(ra.ps.g)$total_sequences)
25
26 # perform differential abundance analysis using ANCOM-BC with count data
27 ancom.s <- ANCOMBC.plus(ps=abun.ps.s,
28                        formula="Case_status + collection_method + seqs_scaled",
29                        p_adj_method="BH",
30                        zero_cut=0.95)
31 ancom.g <- ANCOMBC.plus(ps=abun.ps.g,
32                        formula="Case_status + collection_method + seqs_scaled",
33                        p_adj_method="BH",
34                        zero_cut=0.95)
35
36 # prep temporary directory for MaAsLin2 output
37 system('
38 if [ ! -d "temp_directory" ]
39 then
40   mkdir temp_directory
41 fi
42 ')
43
44 # perform differential abundance analysis using linear regression with
45 # log2 transformed relative abundances
46 suppress(
47 lm.s <- MaAsLin2.plus(ps=phyloseq(otu_table(ra.ps.s)/100,
48 sample_data(ra.ps.s)),
49                       output='temp_directory',
50                       metadata=c('Case_status', 'collection_method', 'seqs_scaled'),
51                       min_prevalence=0.05,

```

```

52         normalization='NONE',
53         max_significance=0.05,
54         standardize=FALSE,
55         plot_heatmap=FALSE,
56         plot_scatter=FALSE)
57     )
58     suppress(
59     lm.g <- MaAsLin2.plus(ps=phyloseq(otu_table(ra.ps.g)/100,
60         sample_data(ra.ps.g)),
61         output='temp_directory',
62         metadata=c('Case_status','collection_method','seqs_scaled'),
63         min_prevalence=0.05,
64         normalization='NONE',
65         max_significance=0.05,
66         standardize=FALSE,
67         plot_heatmap=FALSE,
68         plot_scatter=FALSE)
69     )
70
71     # remove temporary output directory
72     system('rm -r temp_directory')
73
74     # initialize workbook
75     wb <- createWorkbook()
76
77     # coalesce results for species
78     res.summ <- merge(
79         data.frame(Variable=lm.s$result.summary$Variable,
80             Kingdom=gsub('_', ' ',
81                 gsub('k__', ' ',
82                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
83                         function(x){x[1]}))),
84             Phylum=gsub('_', ' ',
85                 gsub('p__', ' ',
86                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
87                         function(x){x[2]}))),
88             Class=gsub('_', ' ',
89                 gsub('c__', ' ',
90                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
91                         function(x){x[3]}))),
92             Order=gsub('_', ' ',
93                 gsub('o__', ' ',
94                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
95                         function(x){x[4]}))),
96             Family=gsub('_', ' ',
97                 gsub('f__', ' ',
98                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
99                         function(x){x[5]}))),
100             Genus=gsub('_', ' ',
101                 gsub('g__', ' ',
102                     sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
103                         function(x){x[6]}))),
104             Species=gsub('_', ' ',
105                 gsub('s__', ' ',

```

```

106         sapply(strsplit(lm.s$result.summary$Feature, "\\|"),
107               function(x){x[7]})),
108 `N PD`=lm.s$result.summary$N1,
109 `N NHC`=lm.s$result.summary$N2,
110 space_1='',
111 `RA in PD`=lm.s$result.summary$Mean1,
112 `RA in NHC`=lm.s$result.summary$Mean2,
113 lm.s$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
114 `FC lower`=lm.s$result.summary$FC_lower,
115 `FC upper`=lm.s$result.summary$FC_upper,
116 space_2='', check.names=FALSE),
117 data.frame(Variable=ancom.s$result.summary$Variable,
118           Kingdom=gsub('_', ' ',
119                       gsub('k__', ' ',
120                             sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
121                                   function(x){x[1]}))),
122           Phylum=gsub('_', ' ',
123                        gsub('p__', ' ',
124                              sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
125                                    function(x){x[2]}))),
126           Class=gsub('_', ' ',
127                     gsub('c__', ' ',
128                           sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
129                                   function(x){x[3]}))),
130           Order=gsub('_', ' ',
131                     gsub('o__', ' ',
132                           sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
133                                   function(x){x[4]}))),
134           Family=gsub('_', ' ',
135                      gsub('f__', ' ',
136                            sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
137                                    function(x){x[5]}))),
138           Genus=gsub('_', ' ',
139                    gsub('g__', ' ',
140                          sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
141                                  function(x){x[6]}))),
142           Species=gsub('_', ' ',
143                       gsub('s__', ' ',
144                             sapply(strsplit(ancom.s$result.summary$Feature, "\\|"),
145                                   function(x){x[7]}))),
146 `N PD`=ancom.s$result.summary$N1,
147 `N NHC`=ancom.s$result.summary$N2,
148 `BC-OA in PD`=ancom.s$result.summary$Mean1,
149 `BC-OA in NHC`=ancom.s$result.summary$Mean2,
150 ancom.s$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
151 `FC lower`=ancom.s$result.summary$FC_lower,
152 `FC upper`=ancom.s$result.summary$FC_upper,
153 check.names=FALSE),
154 by=c('Variable', 'Kingdom', 'Phylum', 'Class', 'Order',
155       'Family', 'Genus', 'Species', 'N PD', 'N NHC'),
156 suffix=c('_m', '_a'), all=TRUE, sort=FALSE)
157 res.summ <- res.summ[res.summ$Variable=='Case_status',-1]
158 res.summ <- rbind(data.frame(Kingdom='', Phylum='', Class='', Order='',
159                             Family='', Genus='', Species=''),

```

```

160         `N PD`='', `N NHC`='', space_1='',
161         `RA in PD`='MaAsLin2 results', `RA in NHC`='',
162         Beta_m='', SE_m='', P_m='', FDR_m='',
163         FC_m='', `FC lower_m`='', `FC upper_m`='', space_2='',
164         `BC-OA in PD`='ANCOM-BC results', `BC-OA in NHC`='',
165         Beta_a='', SE_a='', P_a='', FDR_a='',
166         FC_a='', `FC lower_a`='', `FC upper_a`='',
167         check.names=FALSE),
168     data.frame(Kingdom='Kingdom', Phylum='Phylum', Class='Class', Order='Order',
169               Family='Famiy', Genus='Genus', Species='Species',
170               `N PD`='N PD', `N NHC`='N NHC', space_1='',
171               `RA in PD`='RA in PD', `RA in NHC`='RA in NHC',
172               Beta_m='Beta', SE_m='SE', P_m='P', FDR_m='FDR', FC_m='FC',
173               `FC lower_m`='FC lower', `FC upper_m`='FC upper', space_2='',
174               `BC-OA in PD`='BC-OA in PD', `BC-OA in NHC`='BC-OA in NHC',
175               Beta_a='Beta', SE_a='SE', P_a='P', FDR_a='FDR', FC_a='FC',
176               `FC lower_a`='FC lower', `FC upper_a`='FC upper',
177               check.names=FALSE),
178     res.summ)
179 res.summ[3:(nrow(res.summ)-1),
180         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
181         grep('ANCOM', res.summ[1,]):ncol(res.summ))][
182     is.na(res.summ[3:(nrow(res.summ)-1),
183         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
184         grep('ANCOM', res.summ[1,]):ncol(res.summ)))]] <- 'NT'
185
186 # add species results to workbook and format
187 addWorksheet(wb, 'Species results')
188 writeData(wb, 'Species results', res.summ, keepNA=FALSE, colNames=FALSE)
189 setColWidths(wb, 'Species results', cols=seq_len(ncol(res.summ)),
190             widths=c(10, rep(22,6), rep(11,2), 2, rep(11,9), 2, rep(11,9))) ### format cells
191 mergeCells(wb, 'Species results',
192           cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=1)
193 mergeCells(wb, 'Species results',
194           cols=grep('ANCOM', res.summ[1,]):ncol(res.summ), rows=1)
195 addStyle(wb, 'Species results',
196         cols=seq_len(ncol(res.summ)), rows=1:2, style=bold, stack=TRUE, gridExpand=TRUE) ### font
197 addStyle(wb, 'Species results',
198         cols=seq_len(ncol(res.summ)), rows=c(1,3,(nrow(res.summ)+1)), ### borders
199         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
200 addStyle(wb, 'Species results',
201         cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=2,
202         style=horizontal_border_thin, stack=TRUE)
203 addStyle(wb, 'Species results', cols=grep('ANCOM', res.summ[1,]):ncol(res.summ),
204         rows=2, style=horizontal_border_thin, stack=TRUE)
205 # convert numbers from strings back to numbers
206 convertNum(res.summ, wb, 'Species results', FALSE)
207
208 # coalesce results for genera
209 res.summ <- merge(
210     data.frame(Variable=lm.g$result.summary$Variable,
211               Kingdom=gsub('_', ' '),
212               gsub('k_', '',
213                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),

```

```

214         function(x){x[1]})),
215 Phylum=gsub('_', ' ',
216             gsub('p__', ' ',
217                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),
218                     function(x){x[2]}))),
219 Class=gsub('_', ' ',
220             gsub('c__', ' ',
221                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),
222                     function(x){x[3]}))),
223 Order=gsub('_', ' ',
224             gsub('o__', ' ',
225                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),
226                     function(x){x[4]}))),
227 Family=gsub('_', ' ',
228             gsub('f__', ' ',
229                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),
230                     function(x){x[5]}))),
231 Genus=gsub('_', ' ',
232             gsub('g__', ' ',
233                 sapply(strsplit(lm.g$result.summary$Feature, "\\|"),
234                     function(x){x[6]}))),
235 `N PD`=lm.g$result.summary$N1,
236 `N NHC`=lm.g$result.summary$N2,
237 space_1='',
238 `RA in PD`=lm.g$result.summary$Mean1,
239 `RA in NHC`=lm.g$result.summary$Mean2,
240 lm.g$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
241 `FC lower`=lm.g$result.summary$FC_lower,
242 `FC upper`=lm.g$result.summary$FC_upper,
243 space_2='', check.names=FALSE),
244 data.frame(Variable=ancom.g$result.summary$Variable,
245 Kingdom=gsub('_', ' ',
246             gsub('k__', ' ',
247                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),
248                     function(x){x[1]}))),
249 Phylum=gsub('_', ' ',
250             gsub('p__', ' ',
251                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),
252                     function(x){x[2]}))),
253 Class=gsub('_', ' ',
254             gsub('c__', ' ',
255                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),
256                     function(x){x[3]}))),
257 Order=gsub('_', ' ',
258             gsub('o__', ' ',
259                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),
260                     function(x){x[4]}))),
261 Family=gsub('_', ' ',
262             gsub('f__', ' ',
263                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),
264                     function(x){x[5]}))),
265 Genus=gsub('_', ' ',
266             gsub('g__', ' ',
267                 sapply(strsplit(ancom.g$result.summary$Feature, "\\|"),

```

```

268         function(x){x[6]})),
269     `N PD`=ancom.g$result.summary$N1,
270     `N NHC`=ancom.g$result.summary$N2,
271     `BC-OA in PD`=ancom.g$result.summary$Mean1,
272     `BC-OA in NHC`=ancom.g$result.summary$Mean2,
273     ancom.g$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
274     `FC lower`=ancom.g$result.summary$FC_lower,
275     `FC upper`=ancom.g$result.summary$FC_upper,
276     check.names=FALSE),
277     by=c('Variable', 'Kingdom', 'Phylum', 'Class', 'Order',
278         'Family', 'Genus', 'N PD', 'N NHC'),
279     suffix=c('_m', '_a'), all=TRUE, sort=FALSE)
280 res.summ <- res.summ[res.summ$Variable=='Case_status',-1]
281 res.summ <- rbind(data.frame(Kingdom='', Phylum='', Class='', Order='', Family='', Genus='',
282     `N PD`='', `N NHC`='', space_1='',
283     `RA in PD`='MaAsLin2 results', `RA in NHC`='',
284     Beta_m='', SE_m='', P_m='', FDR_m='',
285     FC_m='', `FC lower_m`='', `FC upper_m`='', space_2='',
286     `BC-OA in PD`='ANCOM-BC results', `BC-OA in NHC`='',
287     Beta_a='', SE_a='', P_a='', FDR_a='',
288     FC_a='', `FC lower_a`='', `FC upper_a`='',
289     check.names=FALSE),
290     data.frame(Kingdom='Kingdom', Phylum='Phylum', Class='Class',
291     Order='Order', Family='Famiy', Genus='Genus',
292     `N PD`='N PD', `N NHC`='N NHC', space_1='',
293     `RA in PD`='RA in PD', `RA in NHC`='RA in NHC',
294     Beta_m='Beta', SE_m='SE', P_m='P', FDR_m='FDR', FC_m='FC',
295     `FC lower_m`='FC lower', `FC upper_m`='FC upper', space_2='',
296     `BC-OA in PD`='BC-OA in PD', `BC-OA in NHC`='BC-OA in NHC',
297     Beta_a='Beta', SE_a='SE', P_a='P', FDR_a='FDR', FC_a='FC',
298     `FC lower_a`='FC lower', `FC upper_a`='FC upper',
299     check.names=FALSE),
300     res.summ)
301 res.summ[3:(nrow(res.summ)-1),
302     c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
303     grep('ANCOM', res.summ[1,]):ncol(res.summ))][
304     is.na(res.summ[3:(nrow(res.summ)-1),
305     c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
306     grep('ANCOM', res.summ[1,]):ncol(res.summ)))] <- 'NT'
307
308 # add genus results to workbook and format
309 addWorksheet(wb, 'Genus results')
310 writeData(wb, 'Genus results', res.summ, keepNA=FALSE, colNames=FALSE)
311 setColWidths(wb, 'Genus results', cols=seq_len(ncol(res.summ)),
312     widths=c(10, rep(22,5), rep(11,2), 2, rep(11,9), 2, rep(11,9))) ### format cells
313 mergeCells(wb, 'Genus results',
314     cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=1)
315 mergeCells(wb, 'Genus results',
316     cols=grep('ANCOM', res.summ[1,]):ncol(res.summ), rows=1)
317 addStyle(wb, 'Genus results',
318     cols=seq_len(ncol(res.summ)), rows=1:2, style=bold, stack=TRUE, gridExpand=TRUE) ### font
319 addStyle(wb, 'Genus results',
320     cols=seq_len(ncol(res.summ)), rows=c(1,3,(nrow(res.summ)+1)), ### borders
321     gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)

```

```

322 addStyle(wb, 'Genus results',
323           cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=2,
324           style=horizontal_border_thin, stack=TRUE)
325 addStyle(wb, 'Genus results',
326           cols=grep('ANCOM', res.summ[1,]):ncol(res.summ),
327           rows=2, style=horizontal_border_thin, stack=TRUE)
328 # convert numbers from strings back to numbers
329 convertNum(res.summ, wb, 'Genus results', FALSE)
330
331 # save workbook
332 saveWorkbook(wb,
333              'PDSshotgunAnalysis_out/3.Taxonomic_associations/MaAsLin2_ANCOMBC_MWAS_PDvsNHC.xlsx',
334              overwrite=TRUE)

```

## MaAsLin2 and ANCOM-BC MWAS concordance

To visualize the concordance of results between MaAsLin2 and ANCOM-BC, the FDR q-values resulting from each method were plotted together. Species tagged as significantly enriched (colored blue) or depleted (colored red) in PD were also highlighted. Venn diagrams showing the overlap of detected signals at FDR < 0.05 and < 0.1 between MaAsLin2 and ANCOM-BC were also generated.

```

1  ##### MAASLIN2 AND ANCOMBC MWAS CONCORDANCE #####
2
3  # get FDR q-values ready for plotting
4  plot.data <- merge(lm.s$result.summary[lm.s$result.summary$Variable == 'Case_status',
5                    c('Feature', 'FDR', 'FC')],
6                    ancom.s$result.summary[ancom.s$result.summary$Variable == 'Case_status',
7                    c('Feature', 'FDR', 'FC')],
8                    by='Feature', suffix=c('_maaslin', '_ancombc'))
9  plot.data <- plot.data[rowSums(is.na(plot.data)) == 0,]
10
11 # tag PD-associated enriched and depleted species
12 plot.data$`PD association` <- ifelse(plot.data[,2] < 0.05 & round(plot.data[,4],1) <= 0.1 |
13   round(plot.data[,2],1) <= 0.1 & plot.data[,4] < 0.05,
14   ifelse(plot.data[,3] > 1 & plot.data[,5] > 1, 'enriched',
15   ifelse(plot.data[,3] < 1 & plot.data[,5] < 1,
16   'depleted', 'opposite directions')),
17   'not associated')
18
19 # create column to label features reaching FDR 1E-4 in either method
20 labels <- gsub('_', ' ', sapply(plot.data[,1], function(x){strsplit(x, 's_')[[1]][2]}))
21 plot.data$labels <- ''
22 plot.data$labels[plot.data[,2] < 1E-4 | plot.data[,4] < 1E-4] <- labels[plot.data[,2] < 1E-4 |
23   plot.data[,4] < 1E-4]
24
25 # tag what species were detected at FDR q-value thresholds of 0.1 and 0.05
26 plot.data$`MaAsLin2 FDR<0.1`[plot.data$FDR_maaslin < 0.1] <- TRUE
27 plot.data$`MaAsLin2 FDR<0.1`[plot.data$FDR_maaslin > 0.1] <- FALSE
28 plot.data$`MaAsLin2 FDR<0.05`[plot.data$FDR_maaslin < 0.05] <- TRUE
29 plot.data$`MaAsLin2 FDR<0.05`[plot.data$FDR_maaslin > 0.05] <- FALSE
30
31 plot.data$`ANCOM-BC FDR<0.1`[plot.data$FDR_ancombc < 0.1] <- TRUE
32 plot.data$`ANCOM-BC FDR<0.1`[plot.data$FDR_ancombc > 0.1] <- FALSE
33 plot.data$`ANCOM-BC FDR<0.05`[plot.data$FDR_ancombc < 0.05] <- TRUE

```

```

34 plot.data$`ANCOM-BC FDR<0.05`[plot.data$FDR_ancombc > 0.05] <- FALSE
35
36 plot.data$`MaAsLin2 FDR<0.1`[plot.data$`MaAsLin2 FDR<0.1` +
37     plot.data$`ANCOM-BC FDR<0.1` == 0] <- NA
38 plot.data$`ANCOM-BC FDR<0.1`[plot.data$`MaAsLin2 FDR<0.1` +
39     plot.data$`ANCOM-BC FDR<0.1` == 0] <- NA
40 plot.data$`MaAsLin2 FDR<0.05`[plot.data$`MaAsLin2 FDR<0.05` +
41     plot.data$`ANCOM-BC FDR<0.05` == 0] <- NA
42 plot.data$`ANCOM-BC FDR<0.05`[plot.data$`MaAsLin2 FDR<0.05` +
43     plot.data$`ANCOM-BC FDR<0.05` == 0] <- NA
44
45 # create venn diagrams of overlapping signals
46 g1 <- ggplot(data=plot.data) +
47     geom_venn(aes(A=`MaAsLin2 FDR<0.1`, B=`ANCOM-BC FDR<0.1`), fill_color='white',
48         stroke_size=0.5, stroke_color='black', stroke_linetype=c('dashed','solid'),
49         set_name_size=4, text_size=7, auto_scale=TRUE, position=position_dodge(2),
50         show_percentage=FALSE) +
51     theme_void()
52 ggsave(
53     'PDSHotgunAnalysis_out/3.Taxonomic_associations/MaAsLin2_vs_ANCOMBC_species_venn_diag_FDR_0.1.pdf',
54     g1, device='pdf', width=5, height=5)
55
56 g2 <- ggplot(data=plot.data) +
57     geom_venn(aes(A=`MaAsLin2 FDR<0.05`, B=`ANCOM-BC FDR<0.05`), fill_color='white',
58         stroke_size=0.5, stroke_color='black', stroke_linetype=c('dashed','solid'),
59         set_name_size=4, text_size=7, auto_scale=TRUE, position=position_dodge(2),
60         show_percentage=FALSE) +
61     theme_void()
62 ggsave(
63     'PDSHotgunAnalysis_out/3.Taxonomic_associations/MaAsLin2_vs_ANCOMBC_species_venn_diag_FDR_0.05.pdf',
64     g2, device='pdf', width=5, height=5)
65
66 # create scatter plot of FDR q-values
67 set.seed(1234)
68 g3 <- ggplot(data=plot.data, aes(y=-log10(plot.data[,2]), x=-log10(plot.data[,4]),
69     fill=`PD association`, label=labels)) +
70     geom_point(size=4, shape=21) +
71     geom_text_repel(min.segment.length=0, box.padding=0.5, size=6, color='grey25') +
72     geom_vline(xintercept=-log10(0.05), color='grey50', linetype='dashed') +
73     geom_hline(yintercept=-log10(0.05), color='grey50', linetype='dashed') +
74     labs(y='-log10(MaAsLin2 FDR)', x='-log10(ANCOM-BC FDR)') +
75     scale_y_continuous(breaks=c(0,-log10(0.05),-log10(0.01),-log10(1E-4),-log10(1E-6),-log10(1E-8)),
76         labels=c('0',
77             paste(round(-log10(0.05),1),' \n','(0.05)',sep=''),
78             paste(-log10(0.01),' \n','(0.01)',sep=''),
79             paste(-log10(1E-4),' \n','(1E-4)',sep=''),
80             paste(-log10(1E-6),' \n','(1E-6)',sep=''),
81             paste(-log10(1E-8),' \n','(1E-8)',sep='')),
82         limits=c(0,-log10(min(plot.data[,c('FDR_maaslin','FDR_ancombc')]))) ,
83         minor_breaks=NULL) +
84     scale_x_continuous(breaks=c(0,-log10(0.05),-log10(0.01),-log10(1E-4),-log10(1E-6),-log10(1E-8)),
85         labels=c('0',
86             paste(round(-log10(0.05),1),' \n','(0.05)',sep=''),
87             paste(-log10(0.01),' \n','(0.01)',sep=''),

```



```

88         paste(-log10(1E-4), '\n', '(1E-4)', sep=''),
89         paste(-log10(1E-6), '\n', '(1E-6)', sep=''),
90         paste(-log10(1E-8), '\n', '(1E-8)', sep='')),
91         limits=c(0, -log10(min(plot.data[,c('FDR_maaslin', 'FDR_ancombc')]))) ,
92         minor_breaks=NULL) +
93     scale_fill_manual(values=c('red', 'blue', 'grey')) +
94     guides(fill=guide_legend(override.aes=list(shape=21))) +
95     theme_bw() +
96     theme(legend.title=element_text(size=20), legend.text=element_text(size=20), legend.text.align=0,
97           legend.position=c(0.87, 0.3), legend.background=element_rect(fill='white', color='grey50'),
98           axis.text.x=element_text(size=18), axis.text.y=element_text(size=18, vjust=0.8),
99           axis.title=element_text(size=20),
100          axis.title.x=element_text(vjust=-0.75), axis.title.y=element_text(vjust=3),
101          plot.margin=margin(t=10, r=30, b=10, l=10, unit = "pt"))
102 ggsave(
103 'PDSShotgunAnalysis_out/3.Taxonomic_associations/MaAsLin2_vs_ANCOMBC_species_MWAS_qvalues.pdf',
104 g3, device='pdf', width=12, height=10)

```

## Genus heterogeneity

Species counts for species that were tested, found significant, and found elevated or reduced for each significant genus from differential abundance analysis were calculated to observe heterogeneity of genera and their association with PD. This was also done for genera not found significant in differential abundance analysis, but had a significant species in the species differential abundance analysis.

```

1  ##### GENUS HETEROGENEITY #####
2
3  # get names of significant taxa and tested taxa
4  sub.data <- merge(lm.s$result.summary[lm.s$result.summary$Variable == 'Case_status',
5                    c('Feature', 'FDR')],
6                  ancom.s$result.summary[ancom.s$result.summary$Variable == 'Case_status',
7                    c('Feature', 'FDR')],
8                  by='Feature')
9  tested.species <- sub.data$Feature[rowSums(is.na(sub.data)) == 0]
10 sig.species <- ifelse(sub.data[,2] < 0.05 & round(sub.data[,3],1) <= 0.1 |
11                      round(sub.data[,2],1) <= 0.1 & sub.data[,3] < 0.05,
12                      sub.data$Feature, NA)
13 sig.species <- sig.species[!is.na(sig.species)]
14
15 sub.data <- merge(lm.g$result.summary[lm.g$result.summary$Variable == 'Case_status',
16                    c('Feature', 'FDR')],
17                  ancom.g$result.summary[ancom.g$result.summary$Variable == 'Case_status',
18                    c('Feature', 'FDR')],
19                  by='Feature')
20 tested.genera <- sub.data$Feature[rowSums(is.na(sub.data)) == 0]
21 sig.genera <- ifelse(sub.data[,2] < 0.05 & round(sub.data[,3],1) <= 0.1 |
22                    round(sub.data[,2],1) <= 0.1 & sub.data[,3] < 0.05,
23                    sub.data$Feature, NA)
24 sig.genera <- sig.genera[!is.na(sig.genera)]
25
26 # create table giving species count for each significant genus
27 species.counts <- data.frame()
28 for (genus in seq_along(tested.genera)){
29   genus.name <- strsplit(tested.genera[genus], '\\|')[[1]][6]

```

```

30 species.n <- length(tested.species[grep(genus.name, tested.species)])
31 elev.species <- length(
32     lm.s$result.summary$Feature[lm.s$result.summary$Variable == 'Case_status' &
33     lm.s$result.summary$Feature %in%
34     sig.species[grep(genus.name, sig.species)] &
35     lm.s$result.summary$FC > 1])
36 red.species <- length(
37     lm.s$result.summary$Feature[lm.s$result.summary$Variable == 'Case_status' &
38     lm.s$result.summary$Feature %in%
39     sig.species[grep(genus.name, sig.species)] &
40     lm.s$result.summary$FC < 1])
41 if (tested.genera[genus] %in% sig.genera){
42     genus.fc <- ifelse(lm.g$result.summary$FC[lm.g$result.summary$Variable == 'Case_status' &
43     lm.g$result.summary$Feature == tested.genera[genus]] < 1,
44     'Reduced', 'Elevated')
45 }else{
46     genus.fc <- 'Missed'
47 }
48 species.counts <- rbind(species.counts,
49     data.frame(`PD-associated genera`=gsub('_', ' ',
50     gsub('g__', ' ', genus.name)),
51     `N species tested`=species.n,
52     `N PD assoc species`=elev.species+red.species,
53     `N species elevated`=elev.species,
54     `N species reduced`=red.species,
55     `Genus level MWAS`=genus.fc,
56     check.names=FALSE))
57 }
58
59 # remove genera who were not significant and did not have any significant species
60 species.counts <- species.counts[species.counts$`N PD assoc species` > 0 |
61     species.counts$`Genus level MWAS` != 'Missed',]
62
63 # sort by genus name and put missed genera at bottom
64 species.counts <- species.counts[order(species.counts$`PD-associated genera`),]
65 species.counts <- rbind(species.counts[species.counts$`Genus level MWAS` != 'Missed',],
66     data.frame(`PD-associated genera`='Association at species level, missed at genus level',
67     `N species tested`='',
68     `N PD assoc species`='',
69     `N species elevated`='',
70     `N species reduced`='',
71     `Genus level MWAS`='',
72     check.names=FALSE),
73     species.counts[species.counts$`Genus level MWAS` == 'Missed',])
74
75 # write results
76 # create workbook
77 wb <- createWorkbook()
78 # add worksheet, write data, and format output
79 addWorksheet(wb, 'Genus hetero')
80 writeData(wb, 'Genus hetero', species.counts, keepNA=TRUE, colNames=TRUE)
81 setColWidths(wb, 'Genus hetero', cols=seq_len(ncol(species.counts)),
82     widths=c(29, rep(18, (ncol(species.counts)-1)))) ### format cells
83 mergeCells(wb, 'Genus hetero', cols=seq_len(ncol(species.counts)), rows=36)

```

```

84 addStyle(wb, 'Genus hetero', cols=seq_len(ncol(species.counts)),
85          rows=c(1,36), style=bold, gridExpand=TRUE, stack=TRUE) ### font
86 addStyle(wb, 'Genus hetero', cols=seq_len(ncol(species.counts)),
87          rows=c(1,2,36,37,(nrow(species.counts)+2)), ### borders
88          style=horizontal_border_med, gridExpand=TRUE, stack=TRUE)
89 # convert numbers from strings back to numbers
90 convertNum(species.counts, wb, 'Genus hetero', TRUE)
91 # save workbook
92 saveWorkbook(wb,
93             'PDSHOTGUNAnalysis_out/3.Taxonomic_associations/Genus_heterogeneity.xlsx',
94             overwrite=TRUE)

```

## Species distributions and fold changes from MWAS

Log2 transformed relative abundances and natural log transformed bias-corrected abundances (estimated from ANCOM-BC) were plotted as boxplots for the 84 PD-associated species to see distribution of the data, along with fold changes from MaAsLin2 and ANCOM-BC. A smaller plot was also created focusing on PD-associated species that had a 75% change in relative abundance (absolute fold change of 1.75 or higher).

```

1  ##### PD-ASSOCIATED SPECIES DISTRIBUTION & FOLD CHANGES #####
2
3  # grab relative abundances and bias-corrected abundances of species
4  ra.spp <- data.frame(otu_table(ra.ps.s)/100, check.names=FALSE)
5  ba.spp <- data.frame(otu_table(ancom.s$bias.corrected.ps), check.names=FALSE)
6
7  # pull out plotting data for PD-associated species
8  spp.ra.data <- ra.spp[,colnames(ra.spp) %in% sig.species, FALSE]
9  spp.ra.fc.data <- lm.s$result.summary[lm.s$result.summary$Feature %in% sig.species &
10                                     lm.s$result.summary$Variable == 'Case_status',
11                                     c('FC', 'FC_lower', 'FC_upper')]
12  spp.ra.fc.data <- spp.ra.fc.data[order(sapply(spp.ra.fc.data$FC,
13                                               function(x){ifelse(x<1,1/x,x)}),
14                                     decreasing=FALSE),]
15  spp.ra.data <- spp.ra.data[,rownames(spp.ra.fc.data), FALSE]
16  spp.ra.fc.data <- data.frame(plot='Absolute fold change with 95%CI', line='MaAsLin2',
17                              variable=gsub('_', ' ',
18                                             sapply(strsplit(rownames(spp.ra.fc.data),
19                                                             "\\|s_"),
20                                                    function(x){x[2]})),
21                              spp.ra.fc.data)
22  spp.ra.data <- data.frame(plot='log2(Relative abundances)',
23                            Case_status=sample_data(ra.ps.s)$Case_status,
24                            spp.ra.data, check.names=FALSE)
25
26  spp.ba.data <- ba.spp[,colnames(ba.spp) %in% sig.species, FALSE]
27  spp.ba.fc.data <- ancom.s$result.summary[ancom.s$result.summary$Feature %in% sig.species &
28                                           ancom.s$result.summary$Variable == 'Case_status',
29                                           c('FC', 'FC_lower', 'FC_upper')]
30  spp.ba.fc.data <- spp.ba.fc.data[rownames(spp.ra.fc.data),, FALSE]
31  spp.ba.data <- spp.ba.data[,rownames(spp.ra.fc.data), FALSE]
32  spp.ba.fc.data <- data.frame(plot='Absolute fold change with 95%CI', line='ANCOM-BC',
33                              variable=gsub('_', ' ',
34                                             sapply(strsplit(rownames(spp.ba.fc.data),
35                                                             "\\|s_"),
36                                                    function(x){x[2]})),
37                              spp.ba.fc.data)
38

```

```

36                                     function(x){x[2]})),
37                                     spp.ba.fc.data)
38 spp.ba.data <- data.frame(plot='log(Bias-corrected abundances)',
39                           Case_status=sample_data(ra.ps.s)$Case_status,
40                           spp.ba.data, check.names=FALSE)
41
42 # combine MaAsLin2 and ANCOM-BC data
43 fc.plot.data <- data.frame(rbind(spp.ra.fc.data, spp.ba.fc.data))
44 ab.plot.data <- merge(spp.ra.data, spp.ba.data, all=TRUE, sort=FALSE)
45 colnames(ab.plot.data)[3:ncol(ab.plot.data)] <- gsub('_', ' ',
46                                     sapply(strsplit(colnames(ab.plot.data)[3:ncol(ab.plot.data)],
47                                               "\\|s_"),
48                                               function(x){x[2]}))
49
50 # prep fold change data for plotting
51 fc.plot.data$line <- factor(fc.plot.data$line, levels=rev(unique(fc.plot.data$line)))
52 fc.plot.data$variable <- factor(fc.plot.data$variable, levels=unique(fc.plot.data$variable))
53 fc.plot.data$color[fc.plot.data$FC < 1] <- 'elevated'
54 fc.plot.data$color[fc.plot.data$FC > 1] <- 'depleted'
55 fc.plot.data$FC_mod[fc.plot.data$FC > 1] <- fc.plot.data$FC[fc.plot.data$FC > 1]-1
56 fc.plot.data$FC_mod[fc.plot.data$FC < 1] <- -((1/fc.plot.data$FC[fc.plot.data$FC < 1])-1)
57 fc.plot.data$FC_lower_mod[fc.plot.data$FC_lower > 1] <-
58     fc.plot.data$FC_lower[fc.plot.data$FC_lower > 1]-1
59 fc.plot.data$FC_lower_mod[fc.plot.data$FC_lower < 1] <-
60     -((1/fc.plot.data$FC_lower[fc.plot.data$FC_lower < 1])-1)
61 fc.plot.data$FC_upper_mod[fc.plot.data$FC_upper > 1] <-
62     fc.plot.data$FC_upper[fc.plot.data$FC_upper > 1]-1
63 fc.plot.data$FC_upper_mod[fc.plot.data$FC_upper < 1] <-
64     -((1/fc.plot.data$FC_upper[fc.plot.data$FC_upper < 1])-1)
65
66 # prep abundance data for plotting
67 ab.plot.data$Case_status <- dplyr::recode(ab.plot.data$Case_status, '1'='PD', '0'='NHC')
68 ab.plot.data$Case_status <- factor(ab.plot.data$Case_status,
69                                   levels=rev(unique(ab.plot.data$Case_status)))
70 ab.plot.data$plot <- factor(ab.plot.data$plot, levels=unique(ab.plot.data$plot))
71 ab.plot.data.melt <- reshape2::melt(ab.plot.data)
72 ab.plot.data.melt <- ab.plot.data.melt[!is.na(ab.plot.data.melt$value),]
73 ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log2(Relative abundances)'] <-
74     log2.trans(ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log2(Relative abundances)'])-20
75 ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log(Bias-corrected abundances)'] <-
76     ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log(Bias-corrected abundances)']+20
77
78 ##### FULL PLOT #####
79
80 # merge data
81 plot.data <- merge(ab.plot.data.melt, fc.plot.data, all=TRUE, sort=FALSE)
82 plot.data$plot <- factor(plot.data$plot, levels=unique(plot.data$plot))
83 plot.data$variable <- factor(gsub('Candidatus Methanomassiliicoccus intestinalis',
84                                 'Candidatus Methanomassiliicoccus\nintestinalis',
85                                 plot.data$variable),
86                             levels=gsub('Candidatus Methanomassiliicoccus intestinalis',
87                                         'Candidatus Methanomassiliicoccus\nintestinalis',
88                                         unique(plot.data$variable)))
89

```

```

90 # create breaks and break labels for plot
91 breaks <- c(-40,-35,-30,-25,-20,-15,-10,-5,-2,0,2,5,10,20,25,30,35)
92 break_labels <- c(paste(breaks[1:5]+20, '\n(', gsub('e\\+00', '', gsub('e-0', 'e-',
93     formatC(2^(breaks[1:5]+20), format='e', digits=0))), ')', sep=''),
94     gsub('1x', '0x', paste(abs(breaks[6:13])+1, 'x', sep='')),
95     paste(breaks[14:17]-20, '\n(', round(exp(breaks[14:17]-20), 1), ')', sep=''))
96
97 # create plot
98 g1 <- ggplot(data=plot.data[grep('log', plot.data$plot)],
99     aes(x=variable, y=value, fill=as.character(Case_status))) +
100     geom_boxplot(notch=FALSE, outlier.size=0.5) +
101     geom_errorbar(inherit.aes=FALSE,
102     data=plot.data[plot.data$plot=='Absolute fold change with 95%CI'],
103     aes(x=variable, ymin=FC_lower_mod, ymax=FC_upper_mod,
104     color=color, linetype=line),
105     width=0, position=position_dodge(0.75), size=0.75) +
106     geom_point(inherit.aes=FALSE,
107     data=plot.data[plot.data$plot=='Absolute fold change with 95%CI'],
108     aes(x=variable, y=FC_mod, color=color, pch=line),
109     position=position_dodge(0.75), size=1.75) +
110     geom_hline(data=plot.data[plot.data$plot=='Absolute fold change with 95%CI'],
111     aes(yintercept=0),
112     size=0.5, linetype='dashed', alpha=0.5) +
113     facet_nested(. ~ plot, scales='free', space='free_y', switch='y',
114     strip=strip_nested(text_y=list(element_text(angle=0))),
115     labeller=labeller(group=label_wrap_gen(width=10),
116     sub_group=label_wrap_gen(width=10))) +
117     scale_x_discrete(position='bottom') +
118     scale_y_continuous(position='right', breaks=breaks, labels=break_labels) +
119     coord_flip() +
120     scale_fill_manual(values=c("#E69F00", "#00BFC4")) +
121     scale_color_manual(values=c("blue", "red"), labels=c("elevated", "depleted")) +
122     scale_linetype_manual(values=c("11", "solid")) +
123     scale_shape_manual(values=c(16, 15)) +
124     guides(fill=guide_legend(order=1, title="Subject group", title.position="top"),
125     color=guide_legend(order=2, title="Fold change direction", title.position="top"),
126     linetype=guide_legend(title="Fold change source", title.position="top", reverse=TRUE),
127     pch=guide_legend(title="Fold change source", title.position="top", reverse=TRUE)) +
128     theme(legend.position="top", legend.key=element_blank(),
129     legend.title=element_text(size=12), legend.text=element_text(size=12),
130     axis.title.x=element_blank(), axis.text.x=element_text(size=10),
131     axis.title.y=element_blank(), axis.text.y=element_text(size=10),
132     strip.text=element_text(size=12),
133     strip.background=element_rect(fill='gray90', color='gray'),
134     strip.placement="outside", panel.spacing.y=unit(0.5, "lines"))
135 ggsave(
136 'PDShotgunAnalysis_out/3.Taxonomic_associations/PD_associated_species_distributions_foldchanges_1.pdf',
137 g1, device='pdf', width=12, height=30)
138
139 ##### REDUCED PLOT #####
140
141 # merge data
142 targets <- fc.plot.data$variable[fc.plot.data$line == 'MaAsLin2' &
143     (round(fc.plot.data$FC, 2) >= 1.75 |

```

```

144         round(fc.plot.data$FC,2) <= 0.57])
145 plot.data <- merge(ab.plot.data.melt[ab.plot.data.melt$variable %in% targets,],
146                 fc.plot.data[fc.plot.data$variable %in% targets,],
147                 all=TRUE, sort=FALSE)
148 plot.data$plot <- factor(plot.data$plot, levels=unique(plot.data$plot))
149
150 # truncate upper limits of fold changes to < 10x
151 plot.data$FC_lower_mod[plot.data$plot == 'Absolute fold change with 95%CI' &
152 plot.data$FC_lower < 0.1] <- -9
153 plot.data$FC_upper_mod[plot.data$plot == 'Absolute fold change with 95%CI' &
154 plot.data$FC_upper > 10] <- 9
155
156 # create breaks and break labels for plot
157 breaks <- c(-40,-35,-30,-25,-20,-9,-5,-2,0,2,5,9,20,25,30,35)
158 break_labels <- c(paste(breaks[1:5]+20, '\n(', gsub('e\\+00', '', gsub('e-0', 'e-',
159 formatC(2^(breaks[1:5]+20), format='e', digits=0))), ')', sep=''),
160 gsub('1x', '0x', paste(abs(breaks[6:12])+1, 'x', sep='')),
161 paste(breaks[13:16]-20, '\n(', round(exp(breaks[13:16]-20), 1), ')', sep=''))
162
163 # create plot
164 g2 <- ggplot(data=plot.data[grep('log', plot.data$plot),],
165             aes(x=variable, y=value, fill=as.character(Case_status))) +
166   geom_boxplot(notch=FALSE, outlier.size=0.5) +
167   geom_errorbar(inherit.aes=FALSE,
168               data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
169               aes(x=variable, ymin=FC_lower_mod, ymax=FC_upper_mod,
170                 color=color, linetype=line),
171               width=0, position=position_dodge(0.75), size=0.75) +
172   geom_point(inherit.aes=FALSE,
173             data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
174             aes(x=variable, y=FC_mod, color=color, pch=line),
175             position=position_dodge(0.75), size=1.75) +
176   geom_hline(data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
177            aes(yintercept=0),
178            size=0.5, linetype='dashed', alpha=0.5) +
179   facet_nested(. ~ plot, scales='free', space='free_y', switch='y',
180              strip=strip_nested(text_y=list(element_text(angle=0))),
181              labeller=labeller(group=label_wrap_gen(width=10),
182                               sub_group=label_wrap_gen(width=10))) +
183   scale_x_discrete(position='bottom') +
184   scale_y_continuous(position='right', breaks=breaks, labels=break_labels) +
185   coord_flip() +
186   scale_fill_manual(values=c("#E69F00", "#00BFC4")) +
187   scale_color_manual(values=c("blue", "red"), labels=c("elevated", "depleted")) +
188   scale_linetype_manual(values=c("11", "solid")) +
189   scale_shape_manual(values=c(16, 15)) +
190   guides(fill=guide_legend(order=1, title="Subject group", title.position="top"),
191          color=guide_legend(order=2, title="Fold change direction", title.position="top"),
192          linetype=guide_legend(title="Fold change source", title.position="top", reverse=TRUE),
193          pch=guide_legend(title="Fold change source", title.position="top", reverse=TRUE)) +
194   theme(legend.position="top", legend.key=element_blank(),
195         legend.title=element_text(size=12), legend.text=element_text(size=12),
196         axis.title.x=element_blank(), axis.text.x=element_text(size=10),
197         axis.title.y=element_blank(), axis.text.y=element_text(size=10),

```

```

198     strip.text=element_text(size=12),
199     strip.background=element_rect(fill='gray90', color='gray'),
200     strip.placement="outside", panel.spacing.y=unit(0.5, "lines"))
201 ggsave(
202 'PDShotgunAnalysis_out/3.Taxonomic_associations/PD_associated_species_distributions_foldchanges_2.pdf',
203 g2, device='pdf', width=12, height=18)

```

## Sex, age, and confounder analysis

To see how PD-species associations are affected when adjusting for age and sex and extrinsic PD-associated subject data (variables associated with PD from earlier subject metadata analysis that are exposure variables not intrinsically or biologically related to the disease), re-ran MaAsLin2 for PD-associated species adjusting for these variables.

- Note: data for pain meds and sleep aid were missing for 5 subjects who had their stool samples collected with sterile swabs, therefore, these subjects were excluded from these analyses to control for collection method instead of adjusting for it in the model.

MaAsLin2 was ran once adjusting for age and sex, and then again adjusting for the 7 potential confounding variables. In both analyses total sequence count per sample (standardized) was also adjusted for. After running the confounding analysis, a table was created tagging which model variables associated with each taxon.

```

1  ##### SEX, AGE, & CONFOUNDER ANALYSES #####
2
3  # recode categorical variable to get correct effect direction and scale numeric variables
4  sample_data(ra.ps.s)$Sex <- dplyr::recode(sample_data(ra.ps.s)$Sex, M=1, F=0)
5  sample_data(ra.ps.s)$Do_you_drink_alcohol <-
6      dplyr::recode(sample_data(ra.ps.s)$Do_you_drink_alcohol, Y=1, N=0)
7  sample_data(ra.ps.s)$Laxatives <- dplyr::recode(sample_data(ra.ps.s)$Laxatives, Y=1, N=0)
8  sample_data(ra.ps.s)$Probiotic <- dplyr::recode(sample_data(ra.ps.s)$Probiotic, Y=1, N=0)
9  sample_data(ra.ps.s)$Pain_med <- dplyr::recode(sample_data(ra.ps.s)$Pain_med, Y=1, N=0)
10 sample_data(ra.ps.s)$Depression_anxiety_mood_med <-
11     dplyr::recode(sample_data(ra.ps.s)$Depression_anxiety_mood_med, Y=1, N=0)
12 sample_data(ra.ps.s)$Antihistamines <- dplyr::recode(sample_data(ra.ps.s)$Antihistamines, Y=1, N=0)
13 sample_data(ra.ps.s)$Sleep_aid <- dplyr::recode(sample_data(ra.ps.s)$Sleep_aid, Y=1, N=0)
14 sample_data(ra.ps.s)$age_scaled <- scale(sample_data(ra.ps.s)$Age_at_collection)
15
16 ##### SEX AND AGE #####
17
18 # prep temporary directory for MaAsLin2 output
19 system('
20 if [ ! -d "temp_directory" ]
21 then
22     mkdir temp_directory
23 fi
24 ')
25
26 # perform differential abundance analysis for PD-associated species using
27 # linear regression with log2 transformed relative abundances
28 variables <- c('Case_status', 'seqs_scaled', 'collection_method', 'Sex', 'age_scaled')
29
30 ps <- phyloseq(otu_table(prune_taxa(sig.species, ra.ps.s))/100, sample_data(ra.ps.s))
31 suppress(
32 lm.s.adj <- MaAsLin2.plus(ps=ps,

```

```

33         metadata=variables,
34         output='temp_directory',
35         min_prevalence=0.05,
36         normalization='NONE',
37         max_significance=0.05,
38         standardize=FALSE,
39         plot_heatmap=FALSE,
40         plot_scatter=FALSE)
41 )
42
43 # coalesce results for sex and age analysis
44 res.summ <- data.frame(Variable=lm.s.adj$result.summary$Variable,
45                       Species=gsub('_', ' ',
46                                   gsub('s__', ' ',
47                                       sapply(strsplit(lm.s.adj$result.summary$Feature, "\\|"),
48                                             function(x){x[7]}))),
49                       lm.s.adj$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
50                       `FC lower`=lm.s.adj$result.summary$FC_lower,
51                       `FC upper`=lm.s.adj$result.summary$FC_upper,
52                       check.names=FALSE)
53 res.summ <- res.summ[order(res.summ$Species),]
54 res.summ$Variable <- gsub('Case_status', 'Case status',
55                          res.summ$Variable)
56 res.summ$Variable <- gsub('seqs_scaled', 'Total sequence count (standardized)',
57                          res.summ$Variable)
58 res.summ$Variable <- gsub('collection_method', 'Stool collection method',
59                          res.summ$Variable)
60 res.summ$Variable <- gsub('age_scaled', 'Age (standardized)',
61                          res.summ$Variable)
62
63 # initialize workbook
64 wb <- createWorkbook()
65
66 # add results for sex and age analysis and format
67 addWorksheet(wb, 'Sex age results')
68 writeData(wb, 'Sex age results', res.summ, keepNA=TRUE, colNames=TRUE)
69 setColWidths(wb, 'Sex age results', cols=seq_len(ncol(res.summ)),
70             widths=c(22, 34, rep(10,7))) ### format cells
71 addStyle(wb, 'Sex age results', cols=seq_len(ncol(res.summ)),
72         rows=1, style=bold, stack=TRUE) ### font
73 addStyle(wb, 'Sex age results', cols=seq_len(ncol(res.summ)),
74         rows=c(1,2,(nrow(res.summ)+2)), ### borders
75         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
76
77 ##### PD EXTRINSIC CONFOUNDERS #####
78
79 # perform differential abundance analysis for PD-associated species using
80 # linear regression with log2 transformed relative abundances
81 # (Note: subjects who had sample collected with sterile swab are removed prior to testing)
82 variables <- c('Case_status', 'seqs_scaled', 'Do_you_drink_alcohol',
83              'Laxatives', 'Probiotic', 'Pain_med',
84              'Depression_anxiety_mood_med', 'Antihistamines', 'Sleep_aid')
85
86 ps <- phyloseq(otu_table(prune_taxa(sig.species,

```



```

87         subset_samples(ra.ps.s,
88                        collection_method==0))/100,
89     sample_data(ra.ps.s))
90 suppress(
91 lm.s.adj <- MaAsLin2.plus(ps=ps,
92                          metadata=variables,
93                          output='temp_directory',
94                          min_prevalence=0.05,
95                          normalization='NONE',
96                          max_significance=0.05,
97                          standardize=FALSE,
98                          plot_heatmap=FALSE,
99                          plot_scatter=FALSE)
100 )
101
102 # remove temporary output directory
103 system('rm -r temp_directory')
104
105 # coalesce results for confounder analysis
106 res.summ <- data.frame(Variable=lm.s.adj$result.summary$Variable,
107                        Species=gsub('_', ' ',
108                                     gsub('s__', ' ',
109                                           sapply(strsplit(lm.s.adj$result.summary$Feature, "\\|"),
110                                                     function(x){x[7]}))),
111                        lm.s.adj$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
112                        `FC lower`=lm.s.adj$result.summary$FC_lower,
113                        `FC upper`=lm.s.adj$result.summary$FC_upper,
114                        check.names=FALSE)
115 res.summ <- res.summ[order(res.summ$Species),]
116 res.summ$Variable <- gsub('Case_status', 'Case status',
117                          res.summ$Variable)
118 res.summ$Variable <- gsub('seqs_scaled', 'Total sequence count (standardized)',
119                          res.summ$Variable)
120 res.summ$Variable <- gsub('Do_you_drink_alcohol', 'Alcohol',
121                          res.summ$Variable)
122 res.summ$Variable <- gsub('Pain_med', 'Pain medication',
123                          res.summ$Variable)
124 res.summ$Variable <- gsub('Depression_anxiety_mood_med',
125                          'Depression, anxiety, mood medication',
126                          res.summ$Variable)
127 res.summ$Variable <- gsub('Sleep_aid', 'Sleep aid',
128                          res.summ$Variable)
129
130 # add results for confounder analysis and format
131 addWorksheet(wb, 'Confounder var results')
132 writeData(wb, 'Confounder var results', res.summ, keepNA=TRUE, colNames=TRUE)
133 setColWidths(wb, 'Confounder var results', cols=seq_len(ncol(res.summ)),
134              widths=c(22, 34, rep(10,7))) ### format cells
135 addStyle(wb, 'Confounder var results', cols=seq_len(ncol(res.summ)),
136          rows=1, style=bold, stack=TRUE) ### font
137 addStyle(wb, 'Confounder var results', cols=seq_len(ncol(res.summ)),
138          rows=c(1,2,(nrow(res.summ)+2)), ### borders
139          gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
140

```

```

141 # make table of what species associated with what variable
142 var.breakdown <- data.frame(Feature=unique(lm.s.adj$result.summary$Feature),
143                               `Associated variable`=NA, check.names=FALSE)
144 for (taxa in seq_len(nrow(var.breakdown))){
145   taxa.name <- var.breakdown$Feature[taxa]
146   sig.var <- lm.s.adj$result.summary[lm.s.adj$result.summary$Feature == taxa.name &
147                                     round(lm.s.adj$result.summary$FDR,1) <= 0.1,]
148   if (nrow(sig.var) > 0){
149     sig.var.lab <- c()
150     for (var in seq_len(nrow(sig.var))){
151       var.res <- sig.var[var,]
152       if (!(is.na(var.res$FC))){
153         if (var.res$FC > 1 && var.res$FDR < 0.05){sig.var.lab <-
154           c(sig.var.lab, paste(var.res$Variable, '++', sep=''))}
155         if (var.res$FC > 1 && var.res$FDR >= 0.05){sig.var.lab <-
156           c(sig.var.lab, paste(var.res$Variable, '+', sep=''))}
157         if (var.res$FC < 1 && var.res$FDR < 0.05){sig.var.lab <-
158           c(sig.var.lab, paste(var.res$Variable, '--', sep=''))}
159         if (var.res$FC < 1 && var.res$FDR >= 0.05){sig.var.lab <-
160           c(sig.var.lab, paste(var.res$Variable, '-', sep=''))}
161       }
162     }
163     var.breakdown$`Associated variable`[taxa] <- paste(sig.var.lab, collapse=',')
164   }
165 }
166 var.breakdown$`Associated variable`[is.na(var.breakdown$`Associated variable`)] <- ''
167
168 # replace variable names with smaller names
169 var.breakdown$`Associated variable` <- gsub('Case_status', 'PD',
170                                             var.breakdown$`Associated variable`)
171 var.breakdown$`Associated variable` <- gsub('Depression_anxiety_mood_med', 'Mood med',
172                                             var.breakdown$`Associated variable`)
173 var.breakdown$`Associated variable` <- gsub('Do_you_drink_alcohol', 'Alcohol',
174                                             var.breakdown$`Associated variable`)
175 var.breakdown$`Associated variable` <- gsub('_', ' ',
176                                             var.breakdown$`Associated variable`)
177
178 # coalesce results for associating variables
179 res.summ <- data.frame(Species=gsub('_', ' ',
180                                     gsub('s__', ' ',
181                                           sapply(strsplit(var.breakdown$Feature, "\\|"),
182                                                     function(x){x[7]}))),
183                       `Associated variable`=var.breakdown$`Associated variable`,
184                       check.names=FALSE)
185 res.summ <- res.summ[order(res.summ$Species),]
186
187 # add results for associating variables and format
188 addWorksheet(wb, 'Assoc confound var')
189 writeData(wb, 'Assoc confound var', res.summ, keepNA=TRUE, colNames=TRUE)
190 setColWidths(wb, 'Assoc confound var', cols=seq_len(ncol(res.summ)),
191              widths=c(22, 30)) ### format cells
192 addStyle(wb, 'Assoc confound var', cols=seq_len(ncol(res.summ)),
193          rows=1, style=bold, stack=TRUE) ### font
194 addStyle(wb, 'Assoc confound var', cols=seq_len(ncol(res.summ)),

```

```

195     rows=c(1,2,(nrow(res.summ)+2)), ### borders
196     gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
197
198 # save workbook
199 saveWorkbook(wb,
200             'PDSHOTGUNAnalysis_out/3.Taxonomic_associations/PD_associated_species_adj_covariates.xlsx',
201             overwrite=TRUE)

```

## Correlation networks

In order to get an inferred ecological picture of the PD and NHC gut microbiome, co-occurrence networks were constructed using SparCC. To construct the correlation networks, pairwise correlations and corresponding P-values were calculated using SparCC (FastSpar C++ implementation) on species count data. These correlations were visualized as a network using the GUI program Gephi.

## SparCC

SparCC was used to calculate pairwise correlations between species in PD and NHC samples separately.

- SparCC (FastSpar) was performed using 100 iterations (double the default) to get inter-random seed stable correlation calculations with the default `--threshold` parameter of 0.1.
- To calculate p-values for SparCC correlations, the input PD and NHC data were randomly permuted 1,000 times to make 1,000 random datasets. SparCC correlations were then calculated on these random datasets. P-values were calculated by comparing SparCC correlations computed on random datasets to those computed on the real data to see how many instances the real data correlations were better than those derived from random.

```

1  ##### SPARCC CORRELATIONS #####
2
3  # separate to PD and healthy NHC data making sure to remove
4  # UNKNOWN group and any taxa with all 0 after subsetting subjects
5  ps.pd.s <- filter_taxa(prune_taxa(taxa_names(abun.ps.s)[grep('UNKNOWN',
6                                     taxa_names(abun.ps.s),
7                                     invert=TRUE)]),
8                          subset_samples(abun.ps.s, Case_status == 1)),
9                          function(x){sum(x > 0) > 0}, TRUE)
10 hc.s <- filter_taxa(prune_taxa(taxa_names(abun.ps.s)[grep('UNKNOWN',
11                                                         taxa_names(abun.ps.s),
12                                                         invert=TRUE)]),
13                     subset_samples(abun.ps.s, Case_status == 0)),
14                     function(x){sum(x > 0) > 0}, TRUE)
15
16 # format for input to SparCC
17 pd.s <- data.frame(OTU_id=sapply(strsplit(as.character(taxa_names(ps.pd.s)), "s__"),
18                               function(x){x[2]}),
19                  t(otu_table(ps.pd.s)), check.names=FALSE)
20
21 hc.s <- data.frame(OTU_id=sapply(strsplit(as.character(taxa_names(ps.hc.s)), "s__"),
22                               function(x){x[2]}),
23                  t(otu_table(ps.hc.s)), check.names=FALSE)
24
25 write.table(pd.s, 'PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_table.txt',
26             row.names=FALSE, quote=FALSE, sep='\t')

```

```

27
28 write.table(hc.s, 'PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_table.txt',
29             row.names=FALSE, quote=FALSE, sep='\t')
30
31 # calculate SparCC correlations in HPC environment
32 system('
33 fastspar --iterations 100 \\  

34 --threads 10 \\  

35 --threshold 0.1 \\  

36 --seed 1234 \\  

37 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_table.txt \\  

38 --correlation PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Cor_Matrix.txt \\  

39 --covariance PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Cov_Matrix.txt
40 ')
41
42 system('
43 fastspar --iterations 100 \\  

44 --threads 10 \\  

45 --threshold 0.1 \\  

46 --seed 1234 \\  

47 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_table.txt \\  

48 --correlation PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Cor_Matrix.txt \\  

49 --covariance PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Cov_Matrix.txt
50 ')
51
52 # create directory for temporary permuted data
53 system('
54 if [ ! -d "temp_data" ]
55 then
56     mkdir temp_data
57     mkdir temp_data/ErrorOut
58     mkdir temp_data/Output
59 fi
60 ')
61
62 # create randomly permuted datasets
63 system('
64 fastspar_bootstrap \\  

65 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_table.txt \\  

66 --threads 20 \\  

67 --number 1000 \\  

68 --seed 1234 \\  

69 --prefix temp_data/PD_temp_data
70 ')
71
72 system('
73 fastspar_bootstrap \\  

74 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_table.txt \\  

75 --threads 20 \\  

76 --number 1000 \\  

77 --seed 1234 \\  

78 --prefix temp_data/HC_temp_data
79 ')
80

```

```

81 # calculated correlations for each permuted dataset
82 # (Note: this section is set up to be run on a HPC with a SLURM scheduler)
83 system('
84 echo "#!/bin/bash" > bash_script.sh
85 echo "#SBATCH --partition=amd-hdr100" >> bash_script.sh
86 echo "#SBATCH --job-name=SparCC" >> bash_script.sh
87 echo "#SBATCH --error=temp_data/ErrorOut/SparCC_%A_%a.err" >> bash_script.sh
88 echo "#SBATCH --output=temp_data/Output/SparCC_%A_%a.out" >> bash_script.sh
89 echo "#SBATCH --time=2:00:00" >> bash_script.sh
90 echo "#SBATCH --ntasks=1" >> bash_script.sh
91 echo "#SBATCH --cpus-per-task=1" >> bash_script.sh
92 echo "#SBATCH --mem-per-cpu=4000" >> bash_script.sh
93 echo "#SBATCH --mail-type=FAIL" >> bash_script.sh
94 echo "#SBATCH --mail-user=wallenz@uab.edu" >> bash_script.sh
95 echo "#SBATCH --array=0-999" >> bash_script.sh
96 echo "#SBATCH --wait" >> bash_script.sh
97 echo " " >> bash_script.sh
98 echo "source ~/miniconda3/etc/profile.d/conda.sh" >> bash_script.sh
99 echo "conda activate fastspar" >> bash_script.sh
100 echo " " >> bash_script.sh
101 echo "fastspar --iterations 100 \\\\" >> bash_script.sh
102 echo "--threads 10 \\\\" >> bash_script.sh
103 echo "--threshold 0.1 \\\\" >> bash_script.sh
104 echo "--seed 1234 \\\\" >> bash_script.sh
105 echo "--otu_table temp_data/PD_temp_data_\\${SLURM_ARRAY_TASK_ID}.tsv \\\\" >> bash_script.sh
106 echo "--correlation temp_data/PD_Cor_Mat_\\${SLURM_ARRAY_TASK_ID}.tsv \\\\" >> bash_script.sh
107 echo "--covariance temp_data/PD_Cov_Mat_\\${SLURM_ARRAY_TASK_ID}.tsv" >> bash_script.sh
108 echo " " >> bash_script.sh
109 echo "fastspar --iterations 100 \\\\" >> bash_script.sh
110 echo "--threads 10 \\\\" >> bash_script.sh
111 echo "--threshold 0.1 \\\\" >> bash_script.sh
112 echo "--seed 1234 \\\\" >> bash_script.sh
113 echo "--otu_table temp_data/HC_temp_data_\\${SLURM_ARRAY_TASK_ID}.tsv \\\\" >> bash_script.sh
114 echo "--correlation temp_data/HC_Cor_Mat_\\${SLURM_ARRAY_TASK_ID}.tsv \\\\" >> bash_script.sh
115 echo "--covariance temp_data/HC_Cov_Mat_\\${SLURM_ARRAY_TASK_ID}.tsv" >> bash_script.sh
116 ')
117
118 system('sbatch bash_script.sh')
119
120 # calculate permuted p-values for each correlation
121 system('
122 fastspar_pvalues --threads 20 \\
123 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_table.txt \\
124 --correlation PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Cor_Matrix.txt \\
125 --prefix temp_data/PD_Cor_Mat_ \\
126 --permutations 1000 \\
127 --outfile PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Pval_Matrix.txt
128 ')
129
130 system('
131 fastspar_pvalues --threads 20 \\
132 --otu_table PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_table.txt \\
133 --correlation PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Cor_Matrix.txt \\
134 --prefix temp_data/HC_Cor_Mat_ \\

```

```

135 --permutations 1000 \\
136 --outfile PDSHotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Pval_Matrix.txt
137 ')
138
139 # clean up
140 system('
141 rm -r temp_data
142 rm bash_script.sh
143 ')

```

## Community detection and creating node and edge files for network visualization

Once SparCC correlations and p-values were computed, they were formatted into node and edge data.frames and files that could be imported into **igraph** and the visualization program **Gephi**.

- Before importing into **igraph**, SparCC correlations were filtered for species correlations that resulted in a permuted p-value of  $< 0.05$ .
- Once imported into **igraph**, edges were further filtered for those with  $r > 0.2$ , and nodes were then filtered for those who had edges remaining.
- Communities of nodes (species) were then detected using the Louvain algorithm via `cluster_louvain` function in **igraph**.
- Degree for each species was calculated using the `degree` function in **igraph**.
- Node and edge CSV files were outputted from R, and PD and NHC networks were plotted in **Gephi** using the force directed **Force Atlas 2** algorithm to position nodes, then coloring first by species cluster memberships (detected with Louvain algorithm) then by PD-associated species differentiating between elevated (blue) vs reduced (red) species.

```

1  ##### COMMUNITY DETECTION & PREPARING NODE/EDGE FILES #####
2
3  # read in SparCC correlations and pvalues
4  pd.cor.s <- read.table('PDSHotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Cor_Matrix.txt',
5                        row.names="#OTU ID", header=TRUE, stringsAsFactors=FALSE,
6                        check.names=FALSE, comment.char='', sep='\t')
7  pd.pval.s <- read.table('PDSHotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_Pval_Matrix.txt',
8                        row.names="#OTU ID", header=TRUE, stringsAsFactors=FALSE,
9                        check.names=FALSE, comment.char='', sep='\t')
10
11 hc.cor.s <- read.table('PDSHotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Cor_Matrix.txt',
12                      row.names="#OTU ID", header=TRUE, stringsAsFactors=FALSE,
13                      check.names=FALSE, comment.char='', sep='\t')
14 hc.pval.s <- read.table('PDSHotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_Pval_Matrix.txt',
15                      row.names="#OTU ID", header=TRUE, stringsAsFactors=FALSE,
16                      check.names=FALSE, comment.char='', sep='\t')
17
18 # subset and merge result data for case status results
19 res.s <- merge(ancom.s$result.summary[ancom.s$result.summary$Variable == 'Case_status' &
20                                     ancom.s$result.summary$Feature != 'UNKNOWN',],
21              lm.s$result.summary[lm.s$result.summary$Variable == 'Case_status',],
22              by=c('Variable', 'Feature', 'N1', 'N2'), suffix=c('_ancombc', '_maaslin'))
23 res.s <- res.s[order(res.s$P_ancom, decreasing=TRUE),]
24
25 # create node file
26 nodes.s <- data.frame(Id=apply(strsplit(as.character(res.s$Feature), "s__"),
27                               function(x){x[2]}),

```

```

28         Label=sapply(strsplit(as.character(res.s$Feature), "s_"),
29                       function(x){x[2]}),
30         ANCOMBC_Beta=res.s$Beta_ancombc,
31         ANCOMBC_FDR=res.s$FDR_ancombc,
32         MaAsLin2_Beta=res.s$Beta_maaslin,
33         MaAsLin2_FDR=res.s$FDR_maaslin,
34         row.names=NULL)
35 nodes.s$`PD-associated` <- 'No'
36 nodes.s$`PD-associated` [((nodes.s$ANCOMBC_FDR < 0.05 & round(nodes.s$MaAsLin2_FDR,1) <= 0.1) |
37                          (round(nodes.s$ANCOMBC_FDR,1) <= 0.1 & nodes.s$MaAsLin2_FDR < 0.05)) &
38                          (nodes.s$ANCOMBC_Beta > 0 & nodes.s$MaAsLin2_Beta > 0)] <- 'Yes_Increased'
39 nodes.s$`PD-associated` [((nodes.s$ANCOMBC_FDR < 0.05 & round(nodes.s$MaAsLin2_FDR,1) <= 0.1) |
40                          (round(nodes.s$ANCOMBC_FDR,1) <= 0.1 & nodes.s$MaAsLin2_FDR < 0.05)) &
41                          (nodes.s$ANCOMBC_Beta < 0 & nodes.s$MaAsLin2_Beta < 0)] <- 'Yes_Decreased'
42 nodes.s$`PD-associated` [is.na(nodes.s$ANCOMBC_FDR) & is.na(nodes.s$MaAsLin2_FDR)] <- 'Not_tested'
43 nodes.s <- nodes.s[,grep('Beta|FDR', colnames(nodes.s), invert=TRUE)]
44
45 # create edge files for PD and NHC
46 corr.direction <- c()
47 corr.direction[pd.cor.s[lower.tri(pd.cor.s)] > 0] <- "+"
48 corr.direction[pd.cor.s[lower.tri(pd.cor.s)] < 0] <- "-"
49 pd.edges.s <- data.frame(Source=t(combn(rownames(pd.cor.s), 2))[,1],
50                             Target=t(combn(rownames(pd.cor.s), 2))[,2],
51                             Weight=abs(pd.cor.s[lower.tri(pd.cor.s)]),
52                             `Direction of correlation`=corr.direction,
53                             `Correlation P-value`=pd.pval.s[lower.tri(pd.pval.s)],
54                             check.names=FALSE)
55
56 corr.direction <- c()
57 corr.direction[hc.cor.s[lower.tri(hc.cor.s)] > 0] <- "+"
58 corr.direction[hc.cor.s[lower.tri(hc.cor.s)] < 0] <- "-"
59 hc.edges.s <- data.frame(Source=t(combn(rownames(hc.cor.s), 2))[,1],
60                             Target=t(combn(rownames(hc.cor.s), 2))[,2],
61                             Weight=abs(hc.cor.s[lower.tri(hc.cor.s)]),
62                             `Direction of correlation`=corr.direction,
63                             `Correlation P-value`=hc.pval.s[lower.tri(hc.pval.s)],
64                             check.names=FALSE)
65
66 # tag edges that contains a significant taxon
67 pd.edges.s$`PD-associated` [pd.edges.s$Source %in%
68                             nodes.s$Id[grep('Yes', nodes.s$`PD-associated`)] |
69                             pd.edges.s$Target %in%
70                             nodes.s$Id[grep('Yes', nodes.s$`PD-associated`))] <- "Yes"
71 pd.edges.s$`PD-associated` [is.na(pd.edges.s$`PD-associated`)] <- "No"
72
73 hc.edges.s$`PD-associated` [hc.edges.s$Source %in%
74                             nodes.s$Id[grep('Yes', nodes.s$`PD-associated`)] |
75                             hc.edges.s$Target %in%
76                             nodes.s$Id[grep('Yes', nodes.s$`PD-associated`))] <- "Yes"
77 hc.edges.s$`PD-associated` [is.na(hc.edges.s$`PD-associated`)] <- "No"
78
79 # filter edges for significant correlations (permuted pvalue < 0.05)
80 pd.edges.s <- pd.edges.s[pd.edges.s$`Correlation P-value` < 0.05,]
81

```

```

82 hc.edges.s <- hc.edges.s[hc.edges.s$`Correlation P-value` < 0.05,]
83
84 # import data into igraph
85 pd.igraph.s <- graph_from_data_frame(pd.edges.s, directed=FALSE, vertices=nodes.s)
86 E(pd.igraph.s)$weight <- E(pd.igraph.s)$Weight
87
88 hc.igraph.s <- graph_from_data_frame(hc.edges.s, directed=FALSE, vertices=nodes.s)
89 E(hc.igraph.s)$weight <- E(hc.igraph.s)$Weight
90
91 # remove edges with correlations < 0.2
92 pd.igraph.s <- delete_edges(pd.igraph.s, which(E(pd.igraph.s)$weight < 0.2))
93
94 hc.igraph.s <- delete_edges(hc.igraph.s, which(E(hc.igraph.s)$weight < 0.2))
95
96 # remove nodes with degree of 0
97 V(pd.igraph.s)$Degree <- degree(pd.igraph.s, normalized=FALSE)
98 pd.igraph.s <- delete_vertices(pd.igraph.s, V(pd.igraph.s)$Degree == 0)
99
100 V(hc.igraph.s)$Degree <- degree(hc.igraph.s, normalized=FALSE)
101 hc.igraph.s <- delete_vertices(hc.igraph.s, V(hc.igraph.s)$Degree == 0)
102
103 # calculate community membership and modularity of networks
104 pd.clusters <- cluster_louvain(pd.igraph.s)
105 V(pd.igraph.s)$Cluster <- pd.clusters$membership
106
107 hc.clusters <- cluster_louvain(hc.igraph.s)
108 V(hc.igraph.s)$Cluster <- hc.clusters$membership
109
110 # add degree and community memberships to node files
111 nodes.s <- merge(nodes.s,
112                 data.frame(Id=V(pd.igraph.s)$name,
113                            `Degree in PD`=V(pd.igraph.s)$Degree,
114                            `Cluster in PD`=V(pd.igraph.s)$Cluster,
115                            check.names=FALSE),
116                 by='Id', all=TRUE)
117 nodes.s <- merge(nodes.s,
118                 data.frame(Id=V(hc.igraph.s)$name,
119                            `Degree in NHC`=V(hc.igraph.s)$Degree,
120                            `Cluster in NHC`=V(hc.igraph.s)$Cluster,
121                            check.names=FALSE),
122                 by='Id', all=TRUE)
123 nodes.s$`Degree in PD`[is.na(nodes.s$`Degree in PD`)] <- 0
124 nodes.s$`Degree in NHC`[is.na(nodes.s$`Degree in NHC`)] <- 0
125 nodes.s$`Cluster in PD`[is.na(nodes.s$`Cluster in PD`)] <- "none"
126 nodes.s$`Cluster in NHC`[is.na(nodes.s$`Cluster in NHC`)] <- "none"
127
128 # output node and edge files
129 write.csv(nodes.s,
130          "PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_node_file.csv",
131          quote=FALSE, row.names=FALSE)
132 write.csv(pd.edges.s,
133          "PDSHOTGUNAnalysis_out/4.a.Network_analysis/Species_SparCC_PD_edge_file.csv",
134          quote=FALSE, row.names=FALSE)
135 write.csv(hc.edges.s,

```



```

136         "PDShotgunAnalysis_out/4.a.Network_analysis/Species_SparCC_HC_edge_file.csv",
137         quote=FALSE, row.names=FALSE)

```

## Analyses of gene families and pathways

### Preparing count data for downstream analyses

```

1  ##### PREPARE RELATIVE ABUNDANCE AND COUNT DATA #####
2
3  # read in metadata
4  metadata <- data.frame(read_xlsx('Source_Data.xlsx', sheet='subject_metadata'))
5  rownames(metadata) <- metadata$sample_name
6
7  # read in tables that were previously generated by functional profiling
8  gene <- data.frame(read_xlsx('Source_Data.xlsx', sheet='humann_KO_group_counts'))
9
10 path <- data.frame(read_xlsx('Source_Data.xlsx', sheet='humann_pathway_counts'))
11
12 # order same as metadata
13 gene <- gene[,c('Gene.Family',metadata$sample_name)]
14
15 path <- path[,c('Pathway',metadata$sample_name)]
16
17 # make table sample x feature
18 rownames(gene) <- gene$Gene.Family
19 gene <- data.frame(t(gene[,-1]), check.names=FALSE)
20
21 rownames(path) <- path$Pathway
22 path <- data.frame(t(path[,-1]), check.names=FALSE)
23
24 # create phyloseq objects for gene families and pathways
25 gene.ps <- phyloseq(otu_table(as.matrix(gene), taxa_are_rows=FALSE),
26                          sample_data(metadata))
27 path.ps <- phyloseq(otu_table(as.matrix(path), taxa_are_rows=FALSE),
28                          sample_data(metadata))

```

### Differential abundance of gene families and pathways

#### MWAS

To perform the differential abundance analyses of KO groups and pathways, abundances of detected pathways and KO groups were provided to ANCOM-BC and MaAsLin2. Parameters and model formula used for ANCOM-BC and MaAsLin2 were kept the same as what was used in taxonomic-based analyses, except KO group and pathway abundances were total sum scaled prior to analyzing with MaAsLin2 since abundances are being used as input this time instead of relative abundances.

```

1  ##### KO GROUP AND PATHWAY MWAS #####
2
3  # recode categorical variable to get correct effect direction and scale total sequence count
4  sample_data(gene.ps)$Case_status <-
5      dplyr::recode(sample_data(gene.ps)$Case_status, PD=1, Control=0)
6  sample_data(gene.ps)$collection_method <-

```

```

7         dplyr::recode(sample_data(gene.ps)$collection_method, swab=1, `OMNIgene GUT`=0)
8 sample_data(gene.ps)$seqs_scaled <- scale(sample_data(gene.ps)$total_sequences)
9
10 sample_data(path.ps)$Case_status <-
11     dplyr::recode(sample_data(path.ps)$Case_status, PD=1, Control=0)
12 sample_data(path.ps)$collection_method <-
13     dplyr::recode(sample_data(path.ps)$collection_method, swab=1, `OMNIgene GUT`=0)
14 sample_data(path.ps)$seqs_scaled <- scale(sample_data(path.ps)$total_sequences)
15
16 # perform differential abundance analysis using ANCOM-BC with abundance data
17 ancom.gene <- ANCOMBC.plus(ps=gene.ps,
18     formula="Case_status + collection_method + seqs_scaled",
19     p_adj_method="BH",
20     zero_cut=0.95)
21
22 ancom.path <- ANCOMBC.plus(ps=path.ps,
23     formula="Case_status + collection_method + seqs_scaled",
24     p_adj_method="BH",
25     zero_cut=0.95)
26
27 # prep temporary directory for MaAsLin2 output
28 system('
29 if [ ! -d "temp_directory" ]
30 then
31     mkdir temp_directory
32 fi
33 ')
34
35 # perform differential abundance analysis using linear regression
36 # with log2 transformed relative abundances
37 suppress(
38 lm.gene <- MaAsLin2.plus(ps=transform_sample_counts(gene.ps, function(x){x/sum(x)}),
39     output='temp_directory',
40     metadata=c('Case_status', 'collection_method', 'seqs_scaled'),
41     min_prevalence=0.05,
42     normalization='NONE',
43     max_significance=0.05,
44     standardize=FALSE,
45     plot_heatmap=FALSE,
46     plot_scatter=FALSE)
47 )
48
49 suppress(
50 lm.path <- MaAsLin2.plus(ps=transform_sample_counts(path.ps, function(x){x/sum(x)}),
51     output='temp_directory',
52     metadata=c('Case_status', 'collection_method', 'seqs_scaled'),
53     min_prevalence=0.05,
54     normalization='NONE',
55     max_significance=0.05,
56     standardize=FALSE,
57     plot_heatmap=FALSE,
58     plot_scatter=FALSE)
59 )
60

```

```

61 # remove temporary output directory
62 system('rm -r temp_directory')
63
64 # initialize workbook
65 wb <- createWorkbook()
66
67 # coalesce results for KO groups
68 res.summ <- data.frame(Variable=lm.gene$result.summary$Variable,
69   `KEGG ID`=sapply(strsplit(lm.gene$result.summary$Feature, ": "),
70     function(x){x[1]}),
71   `KEGG ortholog group`=sapply(strsplit(lm.gene$result.summary$Feature, ": "),
72     function(x){x[2]}),
73   `N PD`=lm.gene$result.summary$N1,
74   `N NHC`=lm.gene$result.summary$N2,
75   space_1='',
76   `RA in PD`=lm.gene$result.summary$Mean1,
77   `RA in NHC`=lm.gene$result.summary$Mean2,
78   lm.gene$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
79   `FC lower`=lm.gene$result.summary$FC_lower,
80   `FC upper`=lm.gene$result.summary$FC_upper,
81   space_2='', check.names=FALSE)
82 res.summ <- merge(res.summ,
83   data.frame(Variable=ancom.gene$result.summary$Variable,
84     `KEGG ID`=sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
85       function(x){x[1]}),
86     `KEGG ortholog group`=sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
87       function(x){x[2]}),
88     `N PD`=ancom.gene$result.summary$N1,
89     `N NHC`=ancom.gene$result.summary$N2,
90     `BC-OA in PD`=ancom.gene$result.summary$Mean1,
91     `BC-OA in NHC`=ancom.gene$result.summary$Mean2,
92     ancom.gene$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
93     `FC lower`=ancom.gene$result.summary$FC_lower,
94     `FC upper`=ancom.gene$result.summary$FC_upper,
95     check.names=FALSE),
96     by=c('Variable', 'KEGG ID', 'KEGG ortholog group', 'N PD', 'N NHC'),
97     suffix=c('_m', '_a'), all=TRUE, sort=FALSE)
98 res.summ <- res.summ[res.summ$Variable=='Case_status',-1]
99 res.summ <- rbind(data.frame(`KEGG ID`='', `KEGG ortholog group`='',
100   `N PD`='', `N NHC`='', space_1='',
101   `RA in PD`='MaAsLin2 results', `RA in NHC`='',
102   Beta_m='', SE_m='', P_m='', FDR_m='',
103   FC_m='', `FC lower_m`='', `FC upper_m`='', space_2='',
104   `BC-OA in PD`='ANCOM-BC results', `BC-OA in NHC`='',
105   Beta_a='', SE_a='', P_a='', FDR_a='',
106   FC_a='', `FC lower_a`='', `FC upper_a`='',
107   check.names=FALSE),
108   data.frame(`KEGG ID`='KEGG ID', `KEGG ortholog group`='KEGG ortholog group',
109   `N PD`='N PD', `N NHC`='N NHC', space_1='',
110   `RA in PD`='RA in PD', `RA in NHC`='RA in NHC', Beta_m='Beta',
111   SE_m='SE', P_m='P', FDR_m='FDR', FC_m='FC',
112   `FC lower_m`='FC lower', `FC upper_m`='FC upper', space_2='',
113   `BC-OA in PD`='BC-OA in PD', `BC-OA in NHC`='BC-OA in NHC',
114   Beta_a='Beta', SE_a='SE', P_a='P', FDR_a='FDR', FC_a='FC',

```

```

115         `FC lower_a`='FC lower', `FC upper_a`='FC upper',
116         check.names=FALSE),
117         res.summ)
118 res.summ[3:(nrow(res.summ)-1),
119         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
120         grep('ANCOM', res.summ[1,]):ncol(res.summ))][
121         is.na(res.summ[3:(nrow(res.summ)-1),
122         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
123         grep('ANCOM', res.summ[1,]):ncol(res.summ)))] <- 'NT'
124
125 # add results for KO groups and format
126 addWorksheet(wb, 'KO group results')
127 writeData(wb, 'KO group results', res.summ, keepNA=FALSE, colNames=FALSE)
128 setColWidths(wb, 'KO group results', cols=seq_len(ncol(res.summ)),
129             widths=c(10, 82, rep(11,2), 2, rep(11,9), 2, rep(11,9))) ### format cells
130 mergeCells(wb, 'KO group results',
131            cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=1)
132 mergeCells(wb, 'KO group results',
133            cols=grep('ANCOM', res.summ[1,]):ncol(res.summ), rows=1)
134 addStyle(wb, 'KO group results', cols=seq_len(ncol(res.summ)),
135         rows=1:2, style=bold, stack=TRUE, gridExpand=TRUE) ### font
136 addStyle(wb, 'KO group results', cols=seq_len(ncol(res.summ)),
137         rows=c(1,3,(nrow(res.summ)+1)), ### borders
138         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
139 addStyle(wb, 'KO group results',
140         cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=2,
141         style=horizontal_border_thin, stack=TRUE)
142 addStyle(wb, 'KO group results', cols=grep('ANCOM', res.summ[1,]):ncol(res.summ),
143         rows=2, style=horizontal_border_thin, stack=TRUE)
144 # convert numbers from strings back to numbers
145 # Note: this convertNum function below may take too long to run
146 # on a standard machine, consider skipping or running over night
147 ###convertNum(res.summ, wb, 'KO group results', FALSE)
148
149 # coalesce results for pathways
150 res.summ <- data.frame(Variable=lm.path$result.summary$Variable,
151                       `MetaCyc ID`=sapply(strsplit(lm.path$result.summary$Feature, ":"),
152                                           function(x){x[1]}),
153                       `Pathway`=sapply(strsplit(lm.path$result.summary$Feature, ":"),
154                                         function(x){x[2]}),
155                       `N PD`=lm.path$result.summary$N1,
156                       `N NHC`=lm.path$result.summary$N2,
157                       space_1='',
158                       `RA in PD`=lm.path$result.summary$Mean1,
159                       `RA in NHC`=lm.path$result.summary$Mean2,
160                       lm.path$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
161                       `FC lower`=lm.path$result.summary$FC_lower,
162                       `FC upper`=lm.path$result.summary$FC_upper,
163                       space_2='', check.names=FALSE)
164 res.summ <- merge(res.summ,
165                  data.frame(Variable=ancom.path$result.summary$Variable,
166                            `MetaCyc ID`=sapply(strsplit(ancom.path$result.summary$Feature, ":"),
167                                                  function(x){x[1]}),
168                            `Pathway`=sapply(strsplit(ancom.path$result.summary$Feature, ":"),

```

```

169         function(x){x[2]}),
170         `N PD`=ancom.path$result.summary$N1,
171         `N NHC`=ancom.path$result.summary$N2,
172         `BC-OA in PD`=ancom.path$result.summary$Mean1,
173         `BC-OA in NHC`=ancom.path$result.summary$Mean2,
174         ancom.path$result.summary[,c('Beta', 'SE', 'P', 'FDR', 'FC')],
175         `FC lower`=ancom.path$result.summary$FC_lower,
176         `FC upper`=ancom.path$result.summary$FC_upper,
177         check.names=FALSE),
178         by=c('Variable', 'MetaCyc ID', 'Pathway', 'N PD', 'N NHC'),
179         suffix=c('_m', '_a'), all=TRUE, sort=FALSE)
180 res.summ <- res.summ[res.summ$Variable=='Case_status',-1]
181 res.summ <- rbind(data.frame(`MetaCyc ID`='', `Pathway`='',
182                             `N PD`='', `N NHC`='', space_1='',
183                             `RA in PD`='MaAsLin2 results', `RA in NHC`='',
184                             Beta_m='', SE_m='', P_m='', FDR_m='',
185                             FC_m='', `FC lower_m`='', `FC upper_m`='', space_2='',
186                             `BC-OA in PD`='ANCOM-BC results', `BC-OA in NHC`='',
187                             Beta_a='', SE_a='', P_a='', FDR_a='',
188                             FC_a='', `FC lower_a`='', `FC upper_a`='',
189                             check.names=FALSE),
190                 data.frame(`MetaCyc ID`='MetaCyc ID', `Pathway`='Pathway',
191                             `N PD`='N PD', `N NHC`='N NHC', space_1='',
192                             `RA in PD`='RA in PD', `RA in NHC`='RA in NHC',
193                             Beta_m='Beta', SE_m='SE', P_m='P', FDR_m='FDR', FC_m='FC',
194                             `FC lower_m`='FC lower', `FC upper_m`='FC upper', space_2='',
195                             `BC-OA in PD`='BC-OA in PD', `BC-OA in NHC`='BC-OA in NHC',
196                             Beta_a='Beta', SE_a='SE', P_a='P', FDR_a='FDR', FC_a='FC',
197                             `FC lower_a`='FC lower', `FC upper_a`='FC upper',
198                             check.names=FALSE),
199                 res.summ)
200 res.summ[3:(nrow(res.summ)-1),
201         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
202         grep('ANCOM', res.summ[1,]):ncol(res.summ))][
203         is.na(res.summ[3:(nrow(res.summ)-1),
204         c(grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1),
205         grep('ANCOM', res.summ[1,]):ncol(res.summ)))] <- 'NT'
206
207 # add results for pathways and format
208 addWorksheet(wb, 'Pathway results')
209 writeData(wb, 'Pathway results', res.summ, keepNA=FALSE, colNames=FALSE)
210 setColWidths(wb, 'Pathway results', cols=seq_len(ncol(res.summ)),
211             widths=c(10, 82, rep(11,2), 2, rep(11,9), 2, rep(11,9))) ### format cells
212 mergeCells(wb, 'Pathway results',
213           cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=1)
214 mergeCells(wb, 'Pathway results',
215           cols=grep('ANCOM', res.summ[1,]):ncol(res.summ), rows=1)
216 addStyle(wb, 'Pathway results', cols=seq_len(ncol(res.summ)),
217         rows=1:2, style=bold, stack=TRUE, gridExpand=TRUE) ### font
218 addStyle(wb, 'Pathway results', cols=seq_len(ncol(res.summ)),
219         rows=c(1,3,(nrow(res.summ)+1)), ### borders
220         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
221 addStyle(wb, 'Pathway results',
222         cols=grep('MaAsLin2', res.summ[1,]):(grep('space_2', colnames(res.summ))-1), rows=2,

```

```

223     style=horizontal_border_thin, stack=TRUE)
224 addStyle(wb, 'Pathway results', cols=grep('ANCOM', res.summ[1,]):ncol(res.summ),
225     rows=2, style=horizontal_border_thin, stack=TRUE)
226 # convert numbers from strings back to numbers
227 convertNum(res.summ, wb, 'Pathway results', FALSE)
228
229 # save workbook
230 saveWorkbook(wb,
231     'PDSHOTGUNAnalysis_out/5.Gene_pathway_associations/MaAsLin2_ANCOMBC_MWAS_PDvsNHC.xlsx',
232     overwrite=TRUE)

```

## Sporulation KO group association with constipation

Given that spore forming bacteria stimulate gut motility, and our data suggest a global depletion of the sporulation KO groups, we speculated, and tested the hypothesis that constipation, a common symptom of PD, may be related to the depletion of spore forming bacteria.

- To determine if broad reduction in sporulation KO groups is associated with constipation, tested association of PD-associated sporulation KO groups with constipation in PD and NHC groups by collapsing the relative abundances of differentially abundant sporulation KO groups (FDR < 0.05 in both MaAsLin2 and ANCOM-BC; 27 in total), and testing for association with constipation in PD and NHC subjects separately using linear regression with log2 transformed relative abundances (as done in MaAsLin2).
- To determine if constipation was driving the PD association signal observed for sporulation KO groups, tested for association of PD with the collapsed sporulation KO groups while including constipation in the model.

```

1  ##### SPORULATION KO GROUPS & CONSTIPATION #####
2
3  # recode variable to get correct effect direction
4  sample_data(gene.ps)$Constipation <- dplyr::recode(sample_data(gene.ps)$Constipation, Y=1, N=0)
5
6  # define target KO groups
7  target_ko <- intersect(ancom.gene$result.summary$Feature[ancom.gene$result.summary$FDR < 0.05 &
8     !is.na(ancom.gene$result.summary$FDR)],
9     lm.gene$result.summary$Feature[lm.gene$result.summary$FDR < 0.05 &
10    !is.na(lm.gene$result.summary$FDR)])
11 target_ko <- target_ko[grep('sporulation', target_ko)]
12
13 # collapse KO group relative abundances into one group
14 ra.mod <- data.frame(otu_table(transform_sample_counts(gene.ps, function(x){x/sum(x)})),
15     check.names=FALSE)
16 ra.mod <- data.frame(`Sporulation KOs`=rowSums(ra.mod[,colnames(ra.mod) %in% target_ko]),
17     ra.mod[,!(colnames(ra.mod) %in% target_ko)], check.names=FALSE)
18
19 # log2 transform
20 log2.ra <- data.frame(apply(ra.mod, 2, log2.trans), check.names=FALSE)
21
22 # begin result data.frame
23 mod.results <- data.frame(`Subject group`="",
24     Beta1="Results for constipation",
25     SE1="",
26     P1="",
27     FC1="")

```

```

28         Beta2="Results for case status",
29         SE2="",
30         P2="",
31         FC2="",
32         check.names=FALSE)
33 mod.results <- rbind(mod.results,
34                     data.frame(`Subject group`="Subject group",
35                                 Beta1="Beta",
36                                 SE1="SE",
37                                 P1="P",
38                                 FC1="FC",
39                                 Beta2="Beta",
40                                 SE2="SE",
41                                 P2="P",
42                                 FC2="FC",
43                                 check.names=FALSE))
44
45 ### PD ###
46
47 # subset for only PD
48 ra.sub <- ra.mod[rownames(ra.mod) %in%
49                 sample_names(subset_samples(gene.ps, Case_status == 1)),]
50 log2.sub <- log2.ra[rownames(log2.ra) %in%
51                   sample_names(subset_samples(gene.ps, Case_status == 1)),]
52 ps.sub <- subset_samples(gene.ps, Case_status == 1)
53
54 # perform linear regression
55 ra.lm <- lm(log2.sub$`Sporulation KOs` ~ Constipation + collection_method + seqs_scaled,
56            data=data.frame(sample_data(ps.sub)))
57
58 # coalesce results
59 mod.results <- rbind(mod.results,
60                     data.frame(`Subject group`="PD",
61                                 Beta1=round(summary(ra.lm)$coefficients[2,1],2),
62                                 SE1=round(summary(ra.lm)$coefficients[2,2],2),
63                                 P1=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
64                                 FC1=round(2^summary(ra.lm)$coefficients[2,1],2),
65                                 Beta2="-",
66                                 SE2="-",
67                                 P2="-",
68                                 FC2="-",
69                                 check.names=FALSE))
70
71 ### NHC ###
72
73 # subset for only NHC
74 ra.sub <- ra.mod[rownames(ra.mod) %in%
75                 sample_names(subset_samples(gene.ps, Case_status == 0)),]
76 log2.sub <- log2.ra[rownames(log2.ra) %in%
77                   sample_names(subset_samples(gene.ps, Case_status == 0)),]
78 ps.sub <- subset_samples(gene.ps, Case_status == 0)
79
80 # perform linear regression
81 ra.lm <- lm(log2.sub$`Sporulation KOs` ~ Constipation + collection_method + seqs_scaled,

```

```

82         data=data.frame(sample_data(ps.sub)))
83
84 # coalesce results
85 mod.results <- rbind(mod.results,
86     data.frame(`Subject group`="NHC",
87     Beta1=round(summary(ra.lm)$coefficients[2,1],2),
88     SE1=round(summary(ra.lm)$coefficients[2,2],2),
89     P1=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
90     FC1=round(2^summary(ra.lm)$coefficients[2,1],2),
91     Beta2="-",
92     SE2="-",
93     P2="-",
94     FC2="-",
95     check.names=FALSE))
96
97 ### PD and NHC ###
98
99 # perform linear regression
100 ra.lm <- lm(log2.ra$`Sporulation KOs` ~
101     Constipation + Case_status + collection_method + seqs_scaled,
102     data=data.frame(sample_data(gene.ps)))
103
104 # coalesce results
105 mod.results <- rbind(mod.results,
106     data.frame(`Subject group`="PD and NHC",
107     Beta1=round(summary(ra.lm)$coefficients[2,1],2),
108     SE1=round(summary(ra.lm)$coefficients[2,2],2),
109     P1=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
110     FC1=round(2^summary(ra.lm)$coefficients[2,1],2),
111     Beta2=round(summary(ra.lm)$coefficients[3,1],2),
112     SE2=round(summary(ra.lm)$coefficients[3,2],2),
113     P2=formatC(summary(ra.lm)$coefficients[3,4],format='e',digits=1),
114     FC2=round(2^summary(ra.lm)$coefficients[3,1],2),
115     check.names=FALSE))
116
117 # write results
118 # create workbook
119 wb <- createWorkbook()
120 # add worksheet, write data, and format output
121 addWorksheet(wb, 'Sporulation KO groups')
122 writeData(wb, 'Sporulation KO groups', mod.results, keepNA=TRUE, colNames=FALSE)
123 setColWidths(wb, 'Sporulation KO groups', cols=seq_len(ncol(mod.results)),
124     widths=c(20, rep(10,4), rep(10,4))) ### format cells
125 addStyle(wb, 'Sporulation KO groups', cols=seq_len(ncol(mod.results)),
126     rows=1:(nrow(mod.results)+1), gridExpand=TRUE, style=center, stack=TRUE)
127 addStyle(wb, 'Sporulation KO groups', cols=seq_len(ncol(mod.results)),
128     rows=1:2, gridExpand=TRUE, style=bold, stack=TRUE) ### font
129 addStyle(wb, 'Sporulation KO groups', cols=seq_len(ncol(mod.results)),
130     rows=c(1,3,(nrow(mod.results)+1)), ### borders
131     gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
132 # convert numbers from strings back to numbers
133 convertNum(mod.results, wb, 'Sporulation KO groups', FALSE)
134 # save workbook
135 saveWorkbook(wb,

```



```

136     'PDSHOTGUNANALYSIS_OUT/5.GENE_PATHWAY_ASSOCIATIONS/Sporulation_KOs_constipation.xlsx',
137     overwrite=TRUE)

```

## Gene family/pathway boxplots with fold changes from MWAS

To summarize differential abundance analysis results of KO groups and pathways, plotted the relative abundances and fold changes of select KO groups and pathways that were chosen based on relevance to current PD literature.

```

1  ##### PD-ASSOCIATED KO GROUPS & PATHWAY #####
2  ##### DISTRIBUTIONS & FOLD CHANGES #####
3
4  # grab relative abundances and bias-corrected abundances of KO groups and pathways
5  ra.gene <- ra.mod
6  ba.gene <- data.frame(otu_table(ancom.gene$bias.corrected.ps), check.names=FALSE)
7  ra.path <- data.frame(otu_table(transform_sample_counts(path.ps, function(x){x/sum(x)})),
8                        check.names=FALSE)
9  ba.path <- data.frame(otu_table(ancom.path$bias.corrected.ps), check.names=FALSE)
10
11 # extract only IDs
12 colnames(ra.gene) <- sapply(strsplit(colnames(ra.gene), ":"), function(x){x[1]})
13 colnames(ba.gene) <- sapply(strsplit(colnames(ba.gene), ":"), function(x){x[1]})
14 colnames(ra.path) <- sapply(strsplit(colnames(ra.path), ":"), function(x){x[1]})
15 colnames(ba.path) <- sapply(strsplit(colnames(ba.path), ":"), function(x){x[1]})
16
17 ### pull out plotting data for selected KOs and pathways
18 ### in order of common categories and proteins
19
20 ## elevated immunogenic bacterial components
21 # LPS/lipid A
22 targets <- c('K02535', 'K09949', 'K04744', 'KDO-NAGLIPASYN-PWY', 'PWY0-881', 'PWY-6285', 'ECASYN-PWY')
23
24 lps.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
25                    ra.path[,colnames(ra.path) %in% targets, FALSE])
26 lps.ra.fc.data <- rbind(
27   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
28                                   function(x){x[1]}) %in% targets &
29   lm.gene$result.summary$Variable == 'Case_status',
30   c('FC', 'FC_lower', 'FC_upper')],
31   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
32                                   function(x){x[1]}) %in% targets &
33   lm.path$result.summary$Variable == 'Case_status',
34   c('FC', 'FC_lower', 'FC_upper')])
35 rownames(lps.ra.fc.data) <- sapply(strsplit(rownames(lps.ra.fc.data), ":"), function(x){x[1]})
36 lps.ra.fc.data <- lps.ra.fc.data[order(lps.ra.fc.data$FC, decreasing=FALSE),]
37 lps.ra.data <- lps.ra.data[,rownames(lps.ra.fc.data), FALSE]
38 lps.ra.fc.data <- data.frame(sub_group='LPS/ lipid A',
39                             plot='Absolute fold change with 95%CI',
40                             line='MaAsLin2',
41                             variable=rownames(lps.ra.fc.data), lps.ra.fc.data)
42 lps.ra.data <- data.frame(sub_group='LPS/ lipid A', plot='log2(Relative abundances)',
43                          Case_status=sample_data(gene.ps)$Case_status, lps.ra.data)
44
45 lps.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],

```

```

46         ba.path[,colnames(ba.path) %in% targets, FALSE])
47 lps.ba.fc.data <- rbind(
48     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
49         function(x){x[1]}) %in% targets &
50         ancom.gene$result.summary$Variable == 'Case_status',
51         c('FC', 'FC_lower', 'FC_upper')],
52     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
53         function(x){x[1]}) %in% targets &
54         ancom.path$result.summary$Variable == 'Case_status',
55         c('FC', 'FC_lower', 'FC_upper')])
56 rownames(lps.ba.fc.data) <- sapply(strsplit(rownames(lps.ba.fc.data), ":"), function(x){x[1]})
57 lps.ba.fc.data <- lps.ba.fc.data[rownames(lps.ra.fc.data),, FALSE]
58 lps.ba.data <- lps.ba.data[,rownames(lps.ra.fc.data), FALSE]
59 lps.ba.fc.data <- data.frame(sub_group='LPS/ lipid A',
60     plot='Absolute fold change with 95%CI',
61     line='ANCOM-BC',
62     variable=rownames(lps.ba.fc.data), lps.ba.fc.data)
63 lps.ba.data <- data.frame(sub_group='LPS/ lipid A', plot='log(Bias-corrected abundances)',
64     Case_status=sample_data(gene.ps)$Case_status, lps.ba.data)
65
66 # LTA
67 targets <- c('K19005', 'K03739')
68
69 lta.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
70     ra.path[,colnames(ra.path) %in% targets, FALSE])
71 lta.ra.fc.data <- rbind(
72     lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
73         function(x){x[1]}) %in% targets &
74     lm.gene$result.summary$Variable == 'Case_status',
75     c('FC', 'FC_lower', 'FC_upper')],
76     lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
77         function(x){x[1]}) %in% targets &
78     lm.path$result.summary$Variable == 'Case_status',
79     c('FC', 'FC_lower', 'FC_upper')])
80 rownames(lta.ra.fc.data) <- sapply(strsplit(rownames(lta.ra.fc.data), ":"), function(x){x[1]})
81 lta.ra.fc.data <- lta.ra.fc.data[order(lta.ra.fc.data$FC, decreasing=FALSE),]
82 lta.ra.data <- lta.ra.data[,rownames(lta.ra.fc.data), FALSE]
83 lta.ra.fc.data <- data.frame(sub_group='LTA',
84     plot='Absolute fold change with 95%CI',
85     line='MaAsLin2',
86     variable=rownames(lta.ra.fc.data), lta.ra.fc.data)
87 lta.ra.data <- data.frame(sub_group='LTA', plot='log2(Relative abundances)',
88     Case_status=sample_data(gene.ps)$Case_status, lta.ra.data)
89
90 lta.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
91     ba.path[,colnames(ba.path) %in% targets, FALSE])
92 lta.ba.fc.data <- rbind(
93     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
94         function(x){x[1]}) %in% targets &
95     ancom.gene$result.summary$Variable == 'Case_status',
96     c('FC', 'FC_lower', 'FC_upper')],
97     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
98         function(x){x[1]}) %in% targets &
99     ancom.path$result.summary$Variable == 'Case_status',

```

```

100         c('FC', 'FC_lower', 'FC_upper']])
101 rownames(lta.ba.fc.data) <- sapply(strsplit(rownames(lta.ba.fc.data), ": "), function(x){x[1]})
102 lta.ba.fc.data <- lta.ba.fc.data[rownames(lta.ra.fc.data),, FALSE]
103 lta.ba.data <- lta.ba.data[,rownames(lta.ra.fc.data), FALSE]
104 lta.ba.fc.data <- data.frame(sub_group='LTA',
105                             plot='Absolute fold change with 95%CI',
106                             line='ANCOM-BC',
107                             variable=rownames(lta.ba.fc.data), lta.ba.fc.data)
108 lta.ba.data <- data.frame(sub_group='LTA', plot='log(Bias-corrected abundances)',
109                           Case_status=sample_data(gene.ps)$Case_status, lta.ba.data)
110
111 # BLP
112 targets <- c('K06078')
113
114 blp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
115                    ra.path[,colnames(ra.path) %in% targets, FALSE])
116 blp.ra.fc.data <- rbind(
117   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ": "),
118                                 function(x){x[1]}) %in% targets &
119                             lm.gene$result.summary$Variable == 'Case_status',
120                             c('FC', 'FC_lower', 'FC_upper')],
121   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ": "),
122                                 function(x){x[1]}) %in% targets &
123                             lm.path$result.summary$Variable == 'Case_status',
124                             c('FC', 'FC_lower', 'FC_upper')])])
125 rownames(blp.ra.fc.data) <- sapply(strsplit(rownames(blp.ra.fc.data), ": "), function(x){x[1]})
126 blp.ra.fc.data <- blp.ra.fc.data[order(blp.ra.fc.data$FC, decreasing=FALSE),]
127 blp.ra.data <- blp.ra.data[,rownames(blp.ra.fc.data), FALSE]
128 blp.ra.fc.data <- data.frame(sub_group='BLP',
129                             plot='Absolute fold change with 95%CI',
130                             line='MaAsLin2',
131                             variable=rownames(blp.ra.fc.data), blp.ra.fc.data)
132 blp.ra.data <- data.frame(sub_group='BLP', plot='log2(Relative abundances)',
133                           Case_status=sample_data(gene.ps)$Case_status, blp.ra.data)
134
135 blp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
136                    ba.path[,colnames(ba.path) %in% targets, FALSE])
137 blp.ba.fc.data <- rbind(
138   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
139                                   function(x){x[1]}) %in% targets &
140                             ancom.gene$result.summary$Variable == 'Case_status',
141                             c('FC', 'FC_lower', 'FC_upper')],
142   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ": "),
143                                   function(x){x[1]}) %in% targets &
144                             ancom.path$result.summary$Variable == 'Case_status',
145                             c('FC', 'FC_lower', 'FC_upper')])])
146 rownames(blp.ba.fc.data) <- sapply(strsplit(rownames(blp.ba.fc.data), ": "), function(x){x[1]})
147 blp.ba.fc.data <- blp.ba.fc.data[rownames(blp.ra.fc.data),, FALSE]
148 blp.ba.data <- blp.ba.data[,rownames(blp.ra.fc.data), FALSE]
149 blp.ba.fc.data <- data.frame(sub_group='BLP',
150                             plot='Absolute fold change with 95%CI',
151                             line='ANCOM-BC',
152                             variable=rownames(blp.ba.fc.data), blp.ba.fc.data)
153 blp.ba.data <- data.frame(sub_group='BLP', plot='log(Bias-corrected abundances)',

```

```

154         Case_status=sample_data(gene.ps)$Case_status, blp.ba.data)
155
156 # combine
157 fc.plot.data <- data.frame(group='Elevated immunogenic bacterial components',
158                             rbind(lps.ra.fc.data, lta.ra.fc.data, blp.ra.fc.data,
159                                   lps.ba.fc.data, lta.ba.fc.data, blp.ba.fc.data))
160 ab.plot.data <- merge(
161     data.frame(group='Elevated immunogenic bacterial components',
162               merge(lps.ra.data, merge(lta.ra.data, blp.ra.data, all=TRUE, sort=FALSE),
163                                     all=TRUE, sort=FALSE)),
164     data.frame(group='Elevated immunogenic bacterial components',
165               merge(lps.ba.data, merge(lta.ba.data, blp.ba.data, all=TRUE, sort=FALSE),
166                                     all=TRUE, sort=FALSE)),
167     all=TRUE, sort=FALSE)
168 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
169
170 ## reduced polysaccharide metabolism and loss of SCFA
171 targets <- c('K17236', 'K17234', 'K16213', 'K00702', 'PWY-7456', 'PWY-6527', 'PWY-7237',
172             'PWY-7242', 'GALACTUROCAT-PWY', 'GLUCUROCAT-PWY', 'GALACT-GLUCUROCAT-PWY')
173
174 pbm.ra.data <- cbind(ra.gene[, colnames(ra.gene) %in% targets, FALSE],
175                    ra.path[, colnames(ra.path) %in% targets, FALSE])
176 pbm.ra.fc.data <- rbind(
177     lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
178                                   function(x){x[1]}) %in% targets &
179                               lm.gene$result.summary$Variable == 'Case_status',
180                               c('FC', 'FC_lower', 'FC_upper')],
181     lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
182                                   function(x){x[1]}) %in% targets &
183                               lm.path$result.summary$Variable == 'Case_status',
184                               c('FC', 'FC_lower', 'FC_upper')])
185 rownames(pbm.ra.fc.data) <- sapply(strsplit(rownames(pbm.ra.fc.data), ":"), function(x){x[1]})
186 pbm.ra.fc.data <- pbm.ra.fc.data[order(pbm.ra.fc.data$FC, decreasing=TRUE),]
187 pbm.ra.data <- pbm.ra.data[, rownames(pbm.ra.fc.data), FALSE]
188 pbm.ra.fc.data <- data.frame(sub_group='---',
189                             plot='Absolute fold change with 95%CI',
190                             line='MaAsLin2',
191                             variable=rownames(pbm.ra.fc.data), pbm.ra.fc.data)
192 pbm.ra.data <- data.frame(sub_group='---', plot='log2(Relative abundances)',
193                          Case_status=sample_data(gene.ps)$Case_status, pbm.ra.data)
194
195 pbm.ba.data <- cbind(ba.gene[, colnames(ba.gene) %in% targets, FALSE],
196                    ba.path[, colnames(ba.path) %in% targets, FALSE])
197 pbm.ba.fc.data <- rbind(
198     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
199                                   function(x){x[1]}) %in% targets &
200                               ancom.gene$result.summary$Variable == 'Case_status',
201                               c('FC', 'FC_lower', 'FC_upper')],
202     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
203                                   function(x){x[1]}) %in% targets &
204                               ancom.path$result.summary$Variable == 'Case_status',
205                               c('FC', 'FC_lower', 'FC_upper')])
206 rownames(pbm.ba.fc.data) <- sapply(strsplit(rownames(pbm.ba.fc.data), ":"), function(x){x[1]})
207 pbm.ba.fc.data <- pbm.ba.fc.data[rownames(pbm.ra.fc.data),, FALSE]

```

```

208 pbm.ba.data <- pbm.ba.data[,rownames(pbm.ra.fc.data), FALSE]
209 pbm.ba.fc.data <- data.frame(sub_group='---',
210                             plot='Absolute fold change with 95%CI',
211                             line='ANCOM-BC',
212                             variable=rownames(pbm.ba.fc.data), pbm.ba.fc.data)
213 pbm.ba.data <- data.frame(sub_group='---', plot='log(Bias-corrected abundances)',
214                           Case_status=sample_data(gene.ps)$Case_status, pbm.ba.data)
215
216 # combine
217 fc.plot.data <- rbind(fc.plot.data,
218                      data.frame(group='Reduced plant-based polysaccharide degradation and SCFA production',
219                                rbind(pbm.ra.fc.data, pbm.ba.fc.data)))
220 ab.plot.data <- merge(ab.plot.data,
221                      merge(data.frame(group='Reduced plant-based polysaccharide degradation and SCFA production',
222                                      pbm.ra.data),
223                            data.frame(group='Reduced plant-based polysaccharide degradation and SCFA production',
224                                      pbm.ba.data),
225                                  all=TRUE, sort=FALSE),
226                          all=TRUE, sort=FALSE)
227 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
228
229 ## elevated proteolytic pathways
230 targets <- c('ORNARGDEG-PWY', 'ORNDEG-PWY', 'THREOCAT-PWY')
231
232 pdp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
233                    ra.path[,colnames(ra.path) %in% targets, FALSE])
234 pdp.ra.fc.data <- rbind(
235   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
236                                   function(x){x[1]}) %in% targets &
237                               lm.gene$result.summary$Variable == 'Case_status',
238                               c('FC', 'FC_lower', 'FC_upper')],
239   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
240                                   function(x){x[1]}) %in% targets &
241                               lm.path$result.summary$Variable == 'Case_status',
242                               c('FC', 'FC_lower', 'FC_upper')])
243 rownames(pdp.ra.fc.data) <- sapply(strsplit(rownames(pdp.ra.fc.data), ":"), function(x){x[1]})
244 pdp.ra.fc.data <- pdp.ra.fc.data[order(pdp.ra.fc.data$FC, decreasing=FALSE),]
245 pdp.ra.data <- pdp.ra.data[,rownames(pdp.ra.fc.data), FALSE]
246 pdp.ra.fc.data <- data.frame(sub_group='---',
247                             plot='Absolute fold change with 95%CI',
248                             line='MaAsLin2',
249                             variable=rownames(pdp.ra.fc.data), pdp.ra.fc.data)
250 pdp.ra.data <- data.frame(sub_group='---', plot='log2(Relative abundances)',
251                           Case_status=sample_data(gene.ps)$Case_status, pdp.ra.data)
252
253 pdp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
254                    ba.path[,colnames(ba.path) %in% targets, FALSE])
255 pdp.ba.fc.data <- rbind(
256   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
257                                   function(x){x[1]}) %in% targets &
258                               ancom.gene$result.summary$Variable == 'Case_status',
259                               c('FC', 'FC_lower', 'FC_upper')],
260   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
261                                   function(x){x[1]}) %in% targets &

```

```

262         ancom.path$result.summary$Variable == 'Case_status',
263         c('FC', 'FC_lower', 'FC_upper'))])
264 rownames(pdp.ba.fc.data) <- sapply(strsplit(rownames(pdp.ba.fc.data), ": "), function(x){x[1]})
265 pdp.ba.fc.data <- pdp.ba.fc.data[rownames(pdp.ra.fc.data),, FALSE]
266 pdp.ba.data <- pdp.ba.data[,rownames(pdp.ra.fc.data), FALSE]
267 pdp.ba.fc.data <- data.frame(sub_group='---',
268                             plot='Absolute fold change with 95%CI',
269                             line='ANCOM-BC',
270                             variable=rownames(pdp.ba.fc.data), pdp.ba.fc.data)
271 pdp.ba.data <- data.frame(sub_group='---', plot='log(Bias-corrected abundances)',
272                          Case_status=sample_data(gene.ps)$Case_status, pdp.ba.data)
273
274 # combine
275 fc.plot.data <- rbind(fc.plot.data, data.frame(group='Increased protein degradation',
276                                               rbind(pdp.ra.fc.data, pdp.ba.fc.data)))
277 ab.plot.data <- merge(ab.plot.data,
278                      merge(data.frame(group='Increased protein degradation', pdp.ra.data),
279                            data.frame(group='Increased protein degradation', pdp.ba.data),
280                            all=TRUE, sort=FALSE),
281                      all=TRUE, sort=FALSE)
282 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
283
284 ## dysregulated neuroactive signaling
285 # dopamine synthesis
286 targets <- c('COMPLETE-ARO-PWY')
287
288 dsp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
289                    ra.path[,colnames(ra.path) %in% targets, FALSE])
290 dsp.ra.fc.data <- rbind(
291   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ": "),
292                                 function(x){x[1]}) %in% targets &
293   lm.gene$result.summary$Variable == 'Case_status',
294   c('FC', 'FC_lower', 'FC_upper')],
295   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ": "),
296                                 function(x){x[1]}) %in% targets &
297   lm.path$result.summary$Variable == 'Case_status',
298   c('FC', 'FC_lower', 'FC_upper')])
299 rownames(dsp.ra.fc.data) <- sapply(strsplit(rownames(dsp.ra.fc.data), ": "), function(x){x[1]})
300 dsp.ra.fc.data <- dsp.ra.fc.data[order(dsp.ra.fc.data$FC, decreasing=TRUE),]
301 dsp.ra.data <- dsp.ra.data[,rownames(dsp.ra.fc.data), FALSE]
302 dsp.ra.fc.data <- data.frame(sub_group='dopamine synthesis',
303                             plot='Absolute fold change with 95%CI',
304                             line='MaAsLin2',
305                             variable=rownames(dsp.ra.fc.data), dsp.ra.fc.data)
306 dsp.ra.data <- data.frame(sub_group='dopamine synthesis', plot='log2(Relative abundances)',
307                          Case_status=sample_data(gene.ps)$Case_status, dsp.ra.data)
308
309 dsp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
310                   ba.path[,colnames(ba.path) %in% targets, FALSE])
311 dsp.ba.fc.data <- rbind(
312   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
313                                   function(x){x[1]}) %in% targets &
314   ancom.gene$result.summary$Variable == 'Case_status',
315   c('FC', 'FC_lower', 'FC_upper')],

```

```

316     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
317         function(x){x[1]}) %in% targets &
318         ancom.path$result.summary$Variable == 'Case_status',
319         c('FC', 'FC_lower', 'FC_upper')]]
320 rownames(dsp.ba.fc.data) <- sapply(strsplit(rownames(dsp.ba.fc.data), ":"), function(x){x[1]})
321 dsp.ba.fc.data <- dsp.ba.fc.data[rownames(dsp.ra.fc.data),, FALSE]
322 dsp.ba.data <- dsp.ba.data[,rownames(dsp.ra.fc.data), FALSE]
323 dsp.ba.fc.data <- data.frame(sub_group='dopamine synthesis',
324     plot='Absolute fold change with 95%CI',
325     line='ANCOM-BC',
326     variable=rownames(dsp.ba.fc.data), dsp.ba.fc.data)
327 dsp.ba.data <- data.frame(sub_group='dopamine synthesis', plot='log(Bias-corrected abundances)',
328     Case_status=sample_data(gene.ps)$Case_status, dsp.ba.data)
329
330 # glutamate synthesis
331 targets <- c('K00266', 'PWY-5505')
332
333 gsp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
334     ra.path[,colnames(ra.path) %in% targets, FALSE])
335 gsp.ra.fc.data <- rbind(
336     lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
337         function(x){x[1]}) %in% targets &
338         lm.gene$result.summary$Variable == 'Case_status',
339         c('FC', 'FC_lower', 'FC_upper')]],
340     lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
341         function(x){x[1]}) %in% targets &
342         lm.path$result.summary$Variable == 'Case_status',
343         c('FC', 'FC_lower', 'FC_upper')]])
344 rownames(gsp.ra.fc.data) <- sapply(strsplit(rownames(gsp.ra.fc.data), ":"), function(x){x[1]})
345 gsp.ra.fc.data <- gsp.ra.fc.data[order(gsp.ra.fc.data$FC, decreasing=TRUE),]
346 gsp.ra.data <- gsp.ra.data[,rownames(gsp.ra.fc.data), FALSE]
347 gsp.ra.fc.data <- data.frame(sub_group='glutamate synthesis',
348     plot='Absolute fold change with 95%CI',
349     line='MaAsLin2',
350     variable=rownames(gsp.ra.fc.data), gsp.ra.fc.data)
351 gsp.ra.data <- data.frame(sub_group='glutamate synthesis', plot='log2(Relative abundances)',
352     Case_status=sample_data(gene.ps)$Case_status, gsp.ra.data)
353
354 gsp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
355     ba.path[,colnames(ba.path) %in% targets, FALSE])
356 gsp.ba.fc.data <- rbind(
357     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
358         function(x){x[1]}) %in% targets &
359         ancom.gene$result.summary$Variable == 'Case_status',
360         c('FC', 'FC_lower', 'FC_upper')]],
361     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
362         function(x){x[1]}) %in% targets &
363         ancom.path$result.summary$Variable == 'Case_status',
364         c('FC', 'FC_lower', 'FC_upper')]])
365 rownames(gsp.ba.fc.data) <- sapply(strsplit(rownames(gsp.ba.fc.data), ":"), function(x){x[1]})
366 gsp.ba.fc.data <- gsp.ba.fc.data[rownames(gsp.ra.fc.data),, FALSE]
367 gsp.ba.data <- gsp.ba.data[,rownames(gsp.ra.fc.data), FALSE]
368 gsp.ba.fc.data <- data.frame(sub_group='glutamate synthesis',
369     plot='Absolute fold change with 95%CI',

```

```

370         line='ANCOM-BC',
371         variable=rownames(gsp.ba.fc.data), gsp.ba.fc.data)
372 gsp.ba.data <- data.frame(sub_group='glutamate synthesis', plot='log(Bias-corrected abundances)',
373         Case_status=sample_data(gene.ps)$Case_status, gsp.ba.data)
374
375 # serotonin synthesis
376 targets <- c('Sporulation KOs', 'TRPSYN-PWY')
377
378 ssp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
379         ra.path[,colnames(ra.path) %in% targets, FALSE])
380 ssp.ra.fc.data <- rbind(data.frame(FC=2^(coef(lm(log2.ra$`Sporulation KOs` ~
381         Case_status + collection_method + seqs_scaled,
382         data=data.frame(sample_data(gene.ps))))[2]),
383         FC_lower=2^(confint(lm(log2.ra$`Sporulation KOs` ~
384         Case_status + collection_method + seqs_scaled,
385         data=data.frame(sample_data(gene.ps))))[2,1]),
386         FC_upper=2^(confint(lm(log2.ra$`Sporulation KOs` ~
387         Case_status + collection_method + seqs_scaled,
388         data=data.frame(sample_data(gene.ps))))[2,2]),
389         row.names='Sporulation KOs'),
390         lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ": "),
391         function(x){x[1]}) %in% targets &
392         lm.path$result.summary$Variable == 'Case_status',
393         c('FC', 'FC_lower', 'FC_upper')])
394 rownames(ssp.ra.fc.data) <- sapply(strsplit(rownames(ssp.ra.fc.data), ": "), function(x){x[1]})
395 ssp.ra.fc.data <- ssp.ra.fc.data[order(ssp.ra.fc.data$FC, decreasing=TRUE),]
396 ssp.ra.data <- ssp.ra.data[,rownames(ssp.ra.fc.data), FALSE]
397 ssp.ra.fc.data <- data.frame(sub_group='serotonin synthesis',
398         plot='Absolute fold change with 95%CI',
399         line='MaAsLin2',
400         variable=rownames(ssp.ra.fc.data), ssp.ra.fc.data)
401 ssp.ra.data <- data.frame(sub_group='serotonin synthesis', plot='log2(Relative abundances)',
402         Case_status=sample_data(gene.ps)$Case_status, ssp.ra.data)
403
404 ssp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
405         ba.path[,colnames(ba.path) %in% targets, FALSE])
406 ssp.ba.fc.data <- rbind(
407         ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
408         function(x){x[1]}) %in% targets &
409         ancom.gene$result.summary$Variable == 'Case_status',
410         c('FC', 'FC_lower', 'FC_upper')],
411         ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ": "),
412         function(x){x[1]}) %in% targets &
413         ancom.path$result.summary$Variable == 'Case_status',
414         c('FC', 'FC_lower', 'FC_upper')])
415 rownames(ssp.ba.fc.data) <- sapply(strsplit(rownames(ssp.ba.fc.data), ": "), function(x){x[1]})
416 #ssp.ba.fc.data <- ssp.ba.fc.data[rownames(ssp.ra.fc.data),, FALSE]
417 #ssp.ba.data <- ssp.ba.data[,rownames(ssp.ra.fc.data), FALSE]
418 ssp.ba.fc.data <- data.frame(sub_group='serotonin synthesis',
419         plot='Absolute fold change with 95%CI',
420         line='ANCOM-BC',
421         variable=rownames(ssp.ba.fc.data), ssp.ba.fc.data)
422 ssp.ba.data <- data.frame(sub_group='serotonin synthesis', plot='log(Bias-corrected abundances)',
423         Case_status=sample_data(gene.ps)$Case_status, ssp.ba.data)

```



```

424
425 # dopamine inhibition
426 targets <- c('K18933')
427
428 dip.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
429                    ra.path[,colnames(ra.path) %in% targets, FALSE])
430 dip.ra.fc.data <- rbind(
431   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
432                                   function(x){x[1]}) %in% targets &
433                               lm.gene$result.summary$Variable == 'Case_status',
434                               c('FC', 'FC_lower', 'FC_upper')],
435   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
436                                   function(x){x[1]}) %in% targets &
437                               lm.path$result.summary$Variable == 'Case_status',
438                               c('FC', 'FC_lower', 'FC_upper')])
439 rownames(dip.ra.fc.data) <- sapply(strsplit(rownames(dip.ra.fc.data), ":"), function(x){x[1]})
440 dip.ra.fc.data <- dip.ra.fc.data[order(dip.ra.fc.data$FC, decreasing=FALSE),]
441 dip.ra.data <- dip.ra.data[,rownames(dip.ra.fc.data), FALSE]
442 dip.ra.fc.data <- data.frame(sub_group='dopamine inhibition',
443                             plot='Absolute fold change with 95%CI',
444                             line='MaAsLin2',
445                             variable=rownames(dip.ra.fc.data), dip.ra.fc.data)
446 dip.ra.data <- data.frame(sub_group='dopamine inhibition', plot='log2(Relative abundances)',
447                             Case_status=sample_data(gene.ps)$Case_status, dip.ra.data)
448
449 dip.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
450                    ba.path[,colnames(ba.path) %in% targets, FALSE])
451 dip.ba.fc.data <- rbind(
452   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
453                                   function(x){x[1]}) %in% targets &
454                               ancom.gene$result.summary$Variable == 'Case_status',
455                               c('FC', 'FC_lower', 'FC_upper')],
456   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
457                                   function(x){x[1]}) %in% targets &
458                               ancom.path$result.summary$Variable == 'Case_status',
459                               c('FC', 'FC_lower', 'FC_upper')])
460 rownames(dip.ba.fc.data) <- sapply(strsplit(rownames(dip.ba.fc.data), ":"), function(x){x[1]})
461 dip.ba.fc.data <- dip.ba.fc.data[rownames(dip.ra.fc.data),, FALSE]
462 dip.ba.data <- dip.ba.data[,rownames(dip.ra.fc.data), FALSE]
463 dip.ba.fc.data <- data.frame(sub_group='dopamine inhibition',
464                             plot='Absolute fold change with 95%CI',
465                             line='ANCOM-BC',
466                             variable=rownames(dip.ba.fc.data), dip.ba.fc.data)
467 dip.ba.data <- data.frame(sub_group='dopamine inhibition', plot='log(Bias-corrected abundances)',
468                             Case_status=sample_data(gene.ps)$Case_status, dip.ba.data)
469
470 # glutamate/GABA degradation
471 targets <- c('PWY-5088', 'ARGDEG-PWY')
472
473 ggd.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
474                    ra.path[,colnames(ra.path) %in% targets, FALSE])
475 ggd.ra.fc.data <- rbind(
476   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
477                                   function(x){x[1]}) %in% targets &

```

```

478         lm.gene$result.summary$Variable == 'Case_status',
479         c('FC', 'FC_lower', 'FC_upper')],
480     lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ": "),
481         function(x){x[1]}) %in% targets &
482         lm.path$result.summary$Variable == 'Case_status',
483         c('FC', 'FC_lower', 'FC_upper')]]
484 rownames(ggd.ra.fc.data) <- sapply(strsplit(rownames(ggd.ra.fc.data), ": "), function(x){x[1]})
485 ggd.ra.fc.data <- ggd.ra.fc.data[order(ggd.ra.fc.data$FC, decreasing=FALSE),]
486 ggd.ra.data <- ggd.ra.data[,rownames(ggd.ra.fc.data), FALSE]
487 ggd.ra.fc.data <- data.frame(sub_group='glutamate/GABA degradation',
488     plot='Absolute fold change with 95%CI',
489     line='MaAsLin2',
490     variable=rownames(ggd.ra.fc.data), ggd.ra.fc.data)
491 ggd.ra.data <- data.frame(sub_group='glutamate/GABA degradation',
492     plot='log2(Relative abundances)',
493     Case_status=sample_data(gene.ps)$Case_status, ggd.ra.data)
494
495 ggd.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
496     ba.path[,colnames(ba.path) %in% targets, FALSE])
497 ggd.ba.fc.data <- rbind(
498     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
499         function(x){x[1]}) %in% targets &
500     ancom.gene$result.summary$Variable == 'Case_status',
501     c('FC', 'FC_lower', 'FC_upper')],
502     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ": "),
503         function(x){x[1]}) %in% targets &
504     ancom.path$result.summary$Variable == 'Case_status',
505     c('FC', 'FC_lower', 'FC_upper')]]
506 rownames(ggd.ba.fc.data) <- sapply(strsplit(rownames(ggd.ba.fc.data), ": "), function(x){x[1]})
507 ggd.ba.fc.data <- ggd.ba.fc.data[rownames(ggd.ra.fc.data),, FALSE]
508 ggd.ba.data <- ggd.ba.data[,rownames(ggd.ra.fc.data), FALSE]
509 ggd.ba.fc.data <- data.frame(sub_group='glutamate/GABA degradation',
510     plot='Absolute fold change with 95%CI',
511     line='ANCOM-BC',
512     variable=rownames(ggd.ba.fc.data), ggd.ba.fc.data)
513 ggd.ba.data <- data.frame(sub_group='glutamate/GABA degradation',
514     plot='log(Bias-corrected abundances)',
515     Case_status=sample_data(gene.ps)$Case_status, ggd.ba.data)
516
517 # combine
518 fc.plot.data <- rbind(fc.plot.data,
519     data.frame(group='Dysregulated neuroactive signaling',
520         rbind(dsp.ra.fc.data, gsp.ra.fc.data, ssp.ra.fc.data,
521             dip.ra.fc.data, ggd.ra.fc.data,
522             dsp.ba.fc.data, gsp.ba.fc.data, ssp.ba.fc.data,
523             dip.ba.fc.data, ggd.ba.fc.data)))
524 ab.plot.data <- merge(ab.plot.data,
525     merge(
526         data.frame(group='Dysregulated neuroactive signaling',
527             merge(dsp.ra.data,
528                 merge(gsp.ra.data,
529                     merge(ssp.ra.data,
530                         merge(dip.ra.data, ggd.ra.data,
531                             all=TRUE, sort=FALSE),

```

```

532         all=TRUE, sort=FALSE),
533         all=TRUE, sort=FALSE),
534         all=TRUE, sort=FALSE)),
535     data.frame(group='Dysregulated neuroactive signaling',
536               merge(dsp.ba.data,
537                     merge(gsp.ba.data,
538                           merge(ssp.ba.data,
539                                 merge(dip.ba.data, ggd.ba.data,
540                                       all=TRUE, sort=FALSE),
541                                       all=TRUE, sort=FALSE),
542                                       all=TRUE, sort=FALSE),
543                                       all=TRUE, sort=FALSE)),
544               all=TRUE, sort=FALSE),
545               all=TRUE, sort=FALSE)
546 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
547 colnames(ab.plot.data) <- gsub('Sporulation-KOs', 'Sporulation KOs', colnames(ab.plot.data))
548
549 ## reduced neuroprotective molecules
550 # nicotinamide degradation
551 targets <- c('K08281')
552
553 ndp.ra.data <- cbind(ra.gene[, colnames(ra.gene) %in% targets, FALSE],
554                    ra.path[, colnames(ra.path) %in% targets, FALSE])
555 ndp.ra.fc.data <- rbind(
556   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
557                                   function(x){x[1]}) %in% targets &
558                               lm.gene$result.summary$Variable == 'Case_status',
559                               c('FC', 'FC_lower', 'FC_upper')],
560   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
561                                   function(x){x[1]}) %in% targets &
562                               lm.path$result.summary$Variable == 'Case_status',
563                               c('FC', 'FC_lower', 'FC_upper')])
564 rownames(ndp.ra.fc.data) <- sapply(strsplit(rownames(ndp.ra.fc.data), ":"), function(x){x[1]})
565 ndp.ra.fc.data <- ndp.ra.fc.data[order(ndp.ra.fc.data$FC, decreasing=FALSE),]
566 ndp.ra.data <- ndp.ra.data[, rownames(ndp.ra.fc.data), FALSE]
567 ndp.ra.fc.data <- data.frame(sub_group='nicotinamide degradation',
568                             plot='Absolute fold change with 95%CI',
569                             line='MaAsLin2',
570                             variable=rownames(ndp.ra.fc.data), ndp.ra.fc.data)
571 ndp.ra.data <- data.frame(sub_group='nicotinamide degradation',
572                             plot='log2(Relative abundances)',
573                             Case_status=sample_data(gene.ps)$Case_status, ndp.ra.data)
574
575 ndp.ba.data <- cbind(ba.gene[, colnames(ba.gene) %in% targets, FALSE],
576                    ba.path[, colnames(ba.path) %in% targets, FALSE])
577 ndp.ba.fc.data <- rbind(
578   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
579                                   function(x){x[1]}) %in% targets &
580                               ancom.gene$result.summary$Variable == 'Case_status',
581                               c('FC', 'FC_lower', 'FC_upper')],
582   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
583                                   function(x){x[1]}) %in% targets &
584                               ancom.path$result.summary$Variable == 'Case_status',
585                               c('FC', 'FC_lower', 'FC_upper')])

```

```

586 rownames(ndp.ba.fc.data) <- sapply(strsplit(rownames(ndp.ba.fc.data), ":"), function(x){x[1]})
587 ndp.ba.fc.data <- ndp.ba.fc.data[rownames(ndp.ra.fc.data),, FALSE]
588 ndp.ba.data <- ndp.ba.data[,rownames(ndp.ra.fc.data), FALSE]
589 ndp.ba.fc.data <- data.frame(sub_group='nicotinamide degradation',
590                             plot='Absolute fold change with 95%CI',
591                             line='ANCOM-BC',
592                             variable=rownames(ndp.ba.fc.data), ndp.ba.fc.data)
593 ndp.ba.data <- data.frame(sub_group='nicotinamide degradation',
594                             plot='log(Bias-corrected abundances)',
595                             Case_status=sample_data(gene.ps)$Case_status, ndp.ba.data)
596
597 # trehalose degradation
598 targets <- c('K00697','K01087','PWY-2723')
599
600 tdp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
601                     ra.path[,colnames(ra.path) %in% targets, FALSE])
602 tdp.ra.fc.data <- rbind(
603   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
604                                   function(x){x[1]}) %in% targets &
605                             lm.gene$result.summary$Variable == 'Case_status',
606                             c('FC','FC_lower','FC_upper')],
607   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
608                                   function(x){x[1]}) %in% targets &
609                             lm.path$result.summary$Variable == 'Case_status',
610                             c('FC','FC_lower','FC_upper')])
611 rownames(tdp.ra.fc.data) <- sapply(strsplit(rownames(tdp.ra.fc.data), ":"), function(x){x[1]})
612 tdp.ra.fc.data <- tdp.ra.fc.data[order(tdp.ra.fc.data$FC, decreasing=FALSE),]
613 tdp.ra.data <- tdp.ra.data[,rownames(tdp.ra.fc.data), FALSE]
614 tdp.ra.fc.data <- data.frame(sub_group='trehalose degradation and metabolism',
615                             plot='Absolute fold change with 95%CI', line='MaAsLin2',
616                             variable=rownames(tdp.ra.fc.data), tdp.ra.fc.data)
617 tdp.ra.data <- data.frame(sub_group='trehalose degradation and metabolism',
618                             plot='log2(Relative abundances)',
619                             Case_status=sample_data(gene.ps)$Case_status, tdp.ra.data)
620
621 tdp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
622                     ba.path[,colnames(ba.path) %in% targets, FALSE])
623 tdp.ba.fc.data <- rbind(
624   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
625                                   function(x){x[1]}) %in% targets &
626                             ancom.gene$result.summary$Variable == 'Case_status',
627                             c('FC','FC_lower','FC_upper')],
628   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
629                                   function(x){x[1]}) %in% targets &
630                             ancom.path$result.summary$Variable == 'Case_status',
631                             c('FC','FC_lower','FC_upper')])
632 rownames(tdp.ba.fc.data) <- sapply(strsplit(rownames(tdp.ba.fc.data), ":"), function(x){x[1]})
633 tdp.ba.fc.data <- tdp.ba.fc.data[rownames(tdp.ra.fc.data),, FALSE]
634 tdp.ba.data <- tdp.ba.data[,rownames(tdp.ra.fc.data), FALSE]
635 tdp.ba.fc.data <- data.frame(sub_group='trehalose degradation and metabolism',
636                             plot='Absolute fold change with 95%CI', line='ANCOM-BC',
637                             variable=rownames(tdp.ba.fc.data), tdp.ba.fc.data)
638 tdp.ba.data <- data.frame(sub_group='trehalose degradation and metabolism',
639                             plot='log(Bias-corrected abundances)',

```

```

640         Case_status=sample_data(gene.ps)$Case_status, tdp.ba.data)
641
642 # combine
643 fc.plot.data <- rbind(fc.plot.data,
644                       data.frame(group='Reduced neuroprotective molecules',
645                                 rbind(ndp.ra.fc.data, tdp.ra.fc.data,
646                                       ndp.ba.fc.data, tdp.ba.fc.data)))
647 ab.plot.data <- merge(ab.plot.data,
648                      merge(data.frame(group='Reduced neuroprotective molecules',
649                                      merge(ndp.ra.data, tdp.ra.data, all=TRUE, sort=FALSE)),
650                            data.frame(group='Reduced neuroprotective molecules',
651                                      merge(ndp.ba.data, tdp.ba.data, all=TRUE, sort=FALSE)),
652                                      all=TRUE, sort=FALSE),
653                      all=TRUE, sort=FALSE)
654 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
655
656 ## elevated bacterial amyloid curli
657 # curli production
658 targets <- c('K04334', 'K04336', 'K06214')
659
660 cpp.ra.data <- cbind(ra.gene[, colnames(ra.gene) %in% targets, FALSE],
661                    ra.path[, colnames(ra.path) %in% targets, FALSE])
662 cpp.ra.fc.data <- rbind(
663   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
664                                   function(x){x[1]}) %in% targets &
665                             lm.gene$result.summary$Variable == 'Case_status',
666                             c('FC', 'FC_lower', 'FC_upper')],
667   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
668                                   function(x){x[1]}) %in% targets &
669                             lm.path$result.summary$Variable == 'Case_status',
670                             c('FC', 'FC_lower', 'FC_upper')])
671 rownames(cpp.ra.fc.data) <- sapply(strsplit(rownames(cpp.ra.fc.data), ":"), function(x){x[1]})
672 cpp.ra.fc.data <- cpp.ra.fc.data[order(cpp.ra.fc.data$FC, decreasing=FALSE),]
673 cpp.ra.data <- cpp.ra.data[, rownames(cpp.ra.fc.data), FALSE]
674 cpp.ra.fc.data <- data.frame(sub_group='curli production',
675                             plot='Absolute fold change with 95%CI',
676                             line='MaAsLin2',
677                             variable=rownames(cpp.ra.fc.data), cpp.ra.fc.data)
678 cpp.ra.data <- data.frame(sub_group='curli production', plot='log2(Relative abundances)',
679                          Case_status=sample_data(gene.ps)$Case_status, cpp.ra.data)
680
681 cpp.ba.data <- cbind(ba.gene[, colnames(ba.gene) %in% targets, FALSE],
682                   ba.path[, colnames(ba.path) %in% targets, FALSE])
683 cpp.ba.fc.data <- rbind(
684   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
685                                       function(x){x[1]}) %in% targets &
686                             ancom.gene$result.summary$Variable == 'Case_status',
687                             c('FC', 'FC_lower', 'FC_upper')],
688   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
689                                       function(x){x[1]}) %in% targets &
690                             ancom.path$result.summary$Variable == 'Case_status',
691                             c('FC', 'FC_lower', 'FC_upper')])
692 rownames(cpp.ba.fc.data) <- sapply(strsplit(rownames(cpp.ba.fc.data), ":"), function(x){x[1]})
693 cpp.ba.fc.data <- cpp.ba.fc.data[rownames(cpp.ra.fc.data),, FALSE]

```

```

694 cpp.ba.data <- cpp.ba.data[,rownames(cpp.ra.fc.data), FALSE]
695 cpp.ba.fc.data <- data.frame(sub_group='curli production',
696                               plot='Absolute fold change with 95%CI',
697                               line='ANCOM-BC',
698                               variable=rownames(cpp.ba.fc.data), cpp.ba.fc.data)
699 cpp.ba.data <- data.frame(sub_group='curli production', plot='log(Bias-corrected abundances)',
700                               Case_status=sample_data(gene.ps)$Case_status, cpp.ba.data)
701
702 # curli regulation
703 targets <- c('K21963')
704
705 crp.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
706                      ra.path[,colnames(ra.path) %in% targets, FALSE])
707 crp.ra.fc.data <- rbind(
708   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ": "),
709                                   function(x){x[1]}) %in% targets &
710                               lm.gene$result.summary$Variable == 'Case_status',
711                               c('FC', 'FC_lower', 'FC_upper')],
712   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ": "),
713                                   function(x){x[1]}) %in% targets &
714                               lm.path$result.summary$Variable == 'Case_status',
715                               c('FC', 'FC_lower', 'FC_upper')])
716 rownames(crp.ra.fc.data) <- sapply(strsplit(rownames(crp.ra.fc.data), ": "), function(x){x[1]})
717 crp.ra.fc.data <- crp.ra.fc.data[order(crp.ra.fc.data$FC, decreasing=FALSE),]
718 crp.ra.data <- crp.ra.data[,rownames(crp.ra.fc.data), FALSE]
719 crp.ra.fc.data <- data.frame(sub_group='curli regulation',
720                               plot='Absolute fold change with 95%CI',
721                               line='MaAsLin2',
722                               variable=rownames(crp.ra.fc.data), crp.ra.fc.data)
723 crp.ra.data <- data.frame(sub_group='curli regulation', plot='log2(Relative abundances)',
724                               Case_status=sample_data(gene.ps)$Case_status, crp.ra.data)
725
726 crp.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
727                      ba.path[,colnames(ba.path) %in% targets, FALSE])
728 crp.ba.fc.data <- rbind(
729   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ": "),
730                                   function(x){x[1]}) %in% targets &
731                               ancom.gene$result.summary$Variable == 'Case_status',
732                               c('FC', 'FC_lower', 'FC_upper')],
733   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ": "),
734                                   function(x){x[1]}) %in% targets &
735                               ancom.path$result.summary$Variable == 'Case_status',
736                               c('FC', 'FC_lower', 'FC_upper')])
737 rownames(crp.ba.fc.data) <- sapply(strsplit(rownames(crp.ba.fc.data), ": "), function(x){x[1]})
738 crp.ba.fc.data <- crp.ba.fc.data[rownames(crp.ra.fc.data),, FALSE]
739 crp.ba.data <- crp.ba.data[,rownames(crp.ra.fc.data), FALSE]
740 crp.ba.fc.data <- data.frame(sub_group='curli regulation',
741                               plot='Absolute fold change with 95%CI',
742                               line='ANCOM-BC',
743                               variable=rownames(crp.ba.fc.data), crp.ba.fc.data)
744 crp.ba.data <- data.frame(sub_group='curli regulation', plot='log(Bias-corrected abundances)',
745                               Case_status=sample_data(gene.ps)$Case_status, crp.ba.data)
746
747 # combine

```

```

748 fc.plot.data <- rbind(fc.plot.data,
749   data.frame(group='Elevated curli, a bacterial amyloid, triggers alpha-synuclein pathology',
750     rbind(cpp.ra.fc.data, crp.ra.fc.data, cpp.ba.fc.data, crp.ba.fc.data)))
751 ab.plot.data <- merge(ab.plot.data,
752   merge(data.frame(group='Elevated curli, a bacterial amyloid, triggers alpha-synuclein pathology',
753     merge(cpp.ra.data, crp.ra.data, all=TRUE, sort=FALSE)),
754     data.frame(group='Elevated curli, a bacterial amyloid, triggers alpha-synuclein pathology',
755     merge(cpp.ba.data, crp.ba.data, all=TRUE, sort=FALSE)),
756     all=TRUE, sort=FALSE),
757   all=TRUE, sort=FALSE)
758 colnames(ab.plot.data) <- gsub('\\.', '-', colnames(ab.plot.data))
759
760 ## elevated toxic metabolite
761 # TMA from choline
762 targets <- c('K20038')
763
764 tch.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
765   ra.path[,colnames(ra.path) %in% targets, FALSE])
766 tch.ra.fc.data <- rbind(
767   lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
768     function(x){x[1]}) %in% targets &
769     lm.gene$result.summary$Variable == 'Case_status',
770     c('FC', 'FC_lower', 'FC_upper')],
771   lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
772     function(x){x[1]}) %in% targets &
773     lm.path$result.summary$Variable == 'Case_status',
774     c('FC', 'FC_lower', 'FC_upper')])
775 rownames(tch.ra.fc.data) <- sapply(strsplit(rownames(tch.ra.fc.data), ":"), function(x){x[1]})
776 tch.ra.fc.data <- tch.ra.fc.data[order(tch.ra.fc.data$FC, decreasing=FALSE),]
777 tch.ra.data <- tch.ra.data[,rownames(tch.ra.fc.data), FALSE]
778 tch.ra.fc.data <- data.frame(sub_group='TMA from choline',
779   plot='Absolute fold change with 95%CI',
780   line='MaAsLin2',
781   variable=rownames(tch.ra.fc.data), tch.ra.fc.data)
782 tch.ra.data <- data.frame(sub_group='TMA from choline', plot='log2(Relative abundances)',
783   Case_status=sample_data(gene.ps)$Case_status, tch.ra.data)
784
785 tch.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
786   ba.path[,colnames(ba.path) %in% targets, FALSE])
787 tch.ba.fc.data <- rbind(
788   ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
789     function(x){x[1]}) %in% targets &
790     ancom.gene$result.summary$Variable == 'Case_status',
791     c('FC', 'FC_lower', 'FC_upper')],
792   ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
793     function(x){x[1]}) %in% targets &
794     ancom.path$result.summary$Variable == 'Case_status',
795     c('FC', 'FC_lower', 'FC_upper')])
796 rownames(tch.ba.fc.data) <- sapply(strsplit(rownames(tch.ba.fc.data), ":"), function(x){x[1]})
797 tch.ba.fc.data <- tch.ba.fc.data[rownames(tch.ra.fc.data),, FALSE]
798 tch.ba.data <- tch.ba.data[,rownames(tch.ra.fc.data), FALSE]
799 tch.ba.fc.data <- data.frame(sub_group='TMA from choline',
800   plot='Absolute fold change with 95%CI',
801   line='ANCOM-BC',

```

```

802         variable=rownames(tch.ba.fc.data), tch.ba.fc.data)
803 tch.ba.data <- data.frame(sub_group='TMA from choline', plot='log(Bias-corrected abundances)',
804         Case_status=sample_data(gene.ps)$Case_status, tch.ba.data)
805
806 # TMA from carnitine
807 targets <- c('K05245')
808
809 tca.ra.data <- cbind(ra.gene[,colnames(ra.gene) %in% targets, FALSE],
810         ra.path[,colnames(ra.path) %in% targets, FALSE])
811 tca.ra.fc.data <- rbind(
812     lm.gene$result.summary[sapply(strsplit(lm.gene$result.summary$Feature, ":"),
813         function(x){x[1]}) %in% targets &
814         lm.gene$result.summary$Variable == 'Case_status',
815         c('FC', 'FC_lower', 'FC_upper')],
816     lm.path$result.summary[sapply(strsplit(lm.path$result.summary$Feature, ":"),
817         function(x){x[1]}) %in% targets &
818         lm.path$result.summary$Variable == 'Case_status',
819         c('FC', 'FC_lower', 'FC_upper')])])
820 rownames(tca.ra.fc.data) <- sapply(strsplit(rownames(tca.ra.fc.data), ":"), function(x){x[1]})
821 tca.ra.fc.data <- tca.ra.fc.data[order(tca.ra.fc.data$FC, decreasing=FALSE),]
822 tca.ra.data <- tca.ra.data[,rownames(tca.ra.fc.data), FALSE]
823 tca.ra.fc.data <- data.frame(sub_group='TMA from carnitine',
824         plot='Absolute fold change with 95%CI',
825         line='MaAsLin2',
826         variable=rownames(tca.ra.fc.data), tca.ra.fc.data)
827 tca.ra.data <- data.frame(sub_group='TMA from carnitine', plot='log2(Relative abundances)',
828         Case_status=sample_data(gene.ps)$Case_status, tca.ra.data)
829
830 tca.ba.data <- cbind(ba.gene[,colnames(ba.gene) %in% targets, FALSE],
831         ba.path[,colnames(ba.path) %in% targets, FALSE])
832 tca.ba.fc.data <- rbind(
833     ancom.gene$result.summary[sapply(strsplit(ancom.gene$result.summary$Feature, ":"),
834         function(x){x[1]}) %in% targets &
835         ancom.gene$result.summary$Variable == 'Case_status',
836         c('FC', 'FC_lower', 'FC_upper')],
837     ancom.path$result.summary[sapply(strsplit(ancom.path$result.summary$Feature, ":"),
838         function(x){x[1]}) %in% targets &
839         ancom.path$result.summary$Variable == 'Case_status',
840         c('FC', 'FC_lower', 'FC_upper')])])
841 rownames(tca.ba.fc.data) <- sapply(strsplit(rownames(tca.ba.fc.data), ":"), function(x){x[1]})
842 tca.ba.fc.data <- tca.ba.fc.data[rownames(tca.ra.fc.data),, FALSE]
843 tca.ba.data <- tca.ba.data[,rownames(tca.ra.fc.data), FALSE]
844 tca.ba.fc.data <- data.frame(sub_group='TMA from carnitine',
845         plot='Absolute fold change with 95%CI',
846         line='ANCOM-BC',
847         variable=rownames(tca.ba.fc.data), tca.ba.fc.data)
848 tca.ba.data <- data.frame(sub_group='TMA from carnitine', plot='log(Bias-corrected abundances)',
849         Case_status=sample_data(gene.ps)$Case_status, tca.ba.data)
850
851 # combine
852 fc.plot.data <- rbind(fc.plot.data,
853         data.frame(group='Elevated toxic metabolite',
854         rbind(tch.ra.fc.data, tca.ra.fc.data,
855         tch.ba.fc.data, tca.ba.fc.data)))

```



```

856 ab.plot.data <- merge(ab.plot.data,
857                       merge(data.frame(group='Elevated toxic metabolite',
858                                     merge(tch.ra.data, tca.ra.data, all=TRUE, sort=FALSE)),
859                                     data.frame(group='Elevated toxic metabolite',
860                                               merge(tch.ba.data, tca.ba.data, all=TRUE, sort=FALSE)),
861                                               all=TRUE, sort=FALSE),
862                       all=TRUE, sort=FALSE)
863 colnames(ab.plot.data) <- gsub('\.', '-', colnames(ab.plot.data))
864
865 # prep fold change data for plotting
866 fc.plot.data$group <- factor(fc.plot.data$group, levels=unique(fc.plot.data$group))
867 fc.plot.data$sub_group <- factor(fc.plot.data$sub_group, levels=unique(fc.plot.data$sub_group))
868 fc.plot.data$line <- factor(fc.plot.data$line, levels=rev(unique(fc.plot.data$line)))
869 fc.plot.data$variable <- factor(fc.plot.data$variable, levels=unique(fc.plot.data$variable))
870 fc.plot.data$color[fc.plot.data$FC < 1] <- 'elevated'
871 fc.plot.data$color[fc.plot.data$FC > 1] <- 'depleted'
872 fc.plot.data$FC_mod[fc.plot.data$FC > 1] <- fc.plot.data$FC[fc.plot.data$FC > 1]-1
873 fc.plot.data$FC_mod[fc.plot.data$FC < 1] <- -((1/fc.plot.data$FC[fc.plot.data$FC < 1])-1)
874 fc.plot.data$FC_lower_mod[fc.plot.data$FC_lower > 1] <-
875     fc.plot.data$FC_lower[fc.plot.data$FC_lower > 1]-1
876 fc.plot.data$FC_lower_mod[fc.plot.data$FC_lower < 1] <-
877     -((1/fc.plot.data$FC_lower[fc.plot.data$FC_lower < 1])-1)
878 fc.plot.data$FC_upper_mod[fc.plot.data$FC_upper > 1] <-
879     fc.plot.data$FC_upper[fc.plot.data$FC_upper > 1]-1
880 fc.plot.data$FC_upper_mod[fc.plot.data$FC_upper < 1] <-
881     -((1/fc.plot.data$FC_upper[fc.plot.data$FC_upper < 1])-1)
882
883 # prep abundance data for plotting
884 ab.plot.data$group <- factor(ab.plot.data$group, levels=unique(ab.plot.data$group))
885 ab.plot.data$sub_group <- factor(ab.plot.data$sub_group, levels=unique(ab.plot.data$sub_group))
886 ab.plot.data$Case_status <- dplyr::recode(ab.plot.data$Case_status, '1'='PD', '0'='NHC')
887 ab.plot.data$Case_status <- factor(ab.plot.data$Case_status,
888                                   levels=rev(unique(ab.plot.data$Case_status)))
889 ab.plot.data$plot <- factor(ab.plot.data$plot, levels=unique(ab.plot.data$plot))
890 ab.plot.data.melt <- reshape2::melt(ab.plot.data)
891 ab.plot.data.melt <- ab.plot.data.melt[!is.na(ab.plot.data.melt$value),]
892 ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log2(Relative abundances)'] <-
893     log2.trans(ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log2(Relative abundances)'])
894 ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log(Bias-corrected abundances)'] <-
895     ab.plot.data.melt$value[ab.plot.data.melt$plot == 'log(Bias-corrected abundances)']+10
896
897 # merge data
898 plot.data <- merge(ab.plot.data.melt, fc.plot.data, all=TRUE, sort=FALSE)
899 plot.data$plot <- factor(plot.data$plot, levels=unique(plot.data$plot))
900
901 # add gene annotations to KOs
902 plot.data$variable <- factor(gsub('K02535', 'K02535 (lpxC)', plot.data$variable),
903                             levels=gsub('K02535', 'K02535 (lpxC)',
904                                           unique(plot.data$variable)))
905 plot.data$variable <- factor(gsub('K04744', 'K04744 (lptD, imp, ostA)', plot.data$variable),
906                             levels=gsub('K04744', 'K04744 (lptD, imp, ostA)',
907                                           unique(plot.data$variable)))
908 plot.data$variable <- factor(gsub('K09949', 'K09949 (lpxI)', plot.data$variable),
909                             levels=gsub('K09949', 'K09949 (lpxI)',

```

```

910         unique(plot.data$variable)))
911 plot.data$variable <- factor(gsub('K19005', 'K19005 (ltaS)', plot.data$variable),
912                             levels=gsub('K19005', 'K19005 (ltaS)',
913                                         unique(plot.data$variable)))
914 plot.data$variable <- factor(gsub('K03739', 'K03739 (dltB)', plot.data$variable),
915                             levels=gsub('K03739', 'K03739 (dltB)',
916                                         unique(plot.data$variable)))
917 plot.data$variable <- factor(gsub('K06078', 'K06078 (lpp)', plot.data$variable),
918                             levels=gsub('K06078', 'K06078 (lpp)',
919                                         unique(plot.data$variable)))
920 plot.data$variable <- factor(gsub('K17236', 'K17236 (araQ)', plot.data$variable),
921                             levels=gsub('K17236', 'K17236 (araQ)',
922                                         unique(plot.data$variable)))
923 plot.data$variable <- factor(gsub('K16213', 'K16213 (cbe, mbe)', plot.data$variable),
924                             levels=gsub('K16213', 'K16213 (cbe, mbe)',
925                                         unique(plot.data$variable)))
926 plot.data$variable <- factor(gsub('K17234', 'K17234 (araN)', plot.data$variable),
927                             levels=gsub('K17234', 'K17234 (araN)',
928                                         unique(plot.data$variable)))
929 plot.data$variable <- factor(gsub('K00702', 'K00702 (E2.4.1.20)', plot.data$variable),
930                             levels=gsub('K00702', 'K00702 (E2.4.1.20)',
931                                         unique(plot.data$variable)))
932 plot.data$variable <- factor(gsub('K00266', 'K00266 (gltD)', plot.data$variable),
933                             levels=gsub('K00266', 'K00266 (gltD)',
934                                         unique(plot.data$variable)))
935 plot.data$variable <- factor(gsub('K18933', 'K18933 (mfnA, adc)', plot.data$variable),
936                             levels=gsub('K18933', 'K18933 (mfnA, adc)',
937                                         unique(plot.data$variable)))
938 plot.data$variable <- factor(gsub('K08281', 'K08281 (pncA)', plot.data$variable),
939                             levels=gsub('K08281', 'K08281 (pncA)',
940                                         unique(plot.data$variable)))
941 plot.data$variable <- factor(gsub('K01087', 'K01087 (otsB)', plot.data$variable),
942                             levels=gsub('K01087', 'K01087 (otsB)',
943                                         unique(plot.data$variable)))
944 plot.data$variable <- factor(gsub('K00697', 'K00697 (otsA)', plot.data$variable),
945                             levels=gsub('K00697', 'K00697 (otsA)',
946                                         unique(plot.data$variable)))
947 plot.data$variable <- factor(gsub('K06214', 'K06214 (csgG)', plot.data$variable),
948                             levels=gsub('K06214', 'K06214 (csgG)',
949                                         unique(plot.data$variable)))
950 plot.data$variable <- factor(gsub('K04336', 'K04336 (csgC)', plot.data$variable),
951                             levels=gsub('K04336', 'K04336 (csgC)',
952                                         unique(plot.data$variable)))
953 plot.data$variable <- factor(gsub('K04334', 'K04334 (csgA)', plot.data$variable),
954                             levels=gsub('K04334', 'K04334 (csgA)',
955                                         unique(plot.data$variable)))
956 plot.data$variable <- factor(gsub('K21963', 'K21963 (ecpR, matA)', plot.data$variable),
957                             levels=gsub('K21963', 'K21963 (ecpR, matA)',
958                                         unique(plot.data$variable)))
959 plot.data$variable <- factor(gsub('K20038', 'K20038 (cutC)', plot.data$variable),
960                             levels=gsub('K20038', 'K20038 (cutC)',
961                                         unique(plot.data$variable)))
962 plot.data$variable <- factor(gsub('K05245', 'K05245 (caiT)', plot.data$variable),
963                             levels=gsub('K05245', 'K05245 (caiT)',

```

```

964         unique(plot.data$variable)))
965
966 # create breaks and break labels for plot
967 breaks <- c(-25,-20,-15,-10,-3,-2,-1,0,1,2,7.5,10,15,20)
968 break_labels <- c(paste(breaks[1:4],'\n(',
969                       gsub('e-0','e-',formatC(2^breaks[1:4],format='e',digits=0)),
970                       ')',sep=''),
971                  paste(gsub('1','0', abs(breaks[5:10])+1), 'x',sep=''),
972                  paste(breaks[11:14]-10,'\n(',round(exp(breaks[11:14]-10),1),')',sep=''))
973
974 # create plot
975 g <- ggplot(data=plot.data[grep('log', plot.data$plot),],
976            aes(x=variable, y=value, fill=as.character(Case_status))) +
977   geom_boxplot(notch=FALSE, outlier.size=0.5) +
978   geom_errorbar(inherit.aes=FALSE,
979               data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
980               aes(x=variable, ymin=FC_lower_mod, ymax=FC_upper_mod,
981                  color=color, linetype=line),
982               width=0, position=position_dodge(0.75), size=0.75) +
983   geom_point(inherit.aes=FALSE,
984             data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
985             aes(x=variable, y=FC_mod, color=color, pch=line),
986             position=position_dodge(0.75), size=1.75) +
987   geom_hline(data=plot.data[plot.data$plot=='Absolute fold change with 95%CI',],
988            aes(yintercept=0), size=0.5, linetype='dashed', alpha=0.5) +
989   facet_nested(group + sub_group ~ plot, scales='free', space='free_y', switch='y',
990              strip=strip_nested(text_y=list(element_text(angle=0))),
991              labeller=labeller(group=label_wrap_gen(width=10),
992                               sub_group=label_wrap_gen(width=10))) +
993   scale_x_discrete(position='bottom') +
994   scale_y_continuous(position='right', breaks=breaks, labels=break_labels) +
995   coord_flip() +
996   scale_fill_manual(values=c("#E69F00", "#00BFC4")) +
997   scale_color_manual(values=c("blue", "red"), labels=c("elevated","depleted")) +
998   scale_linetype_manual(values=c("11", "solid")) +
999   scale_shape_manual(values=c(16, 15)) +
1000  guides(fill=guide_legend(order=1, title="Subject group", title.position="top"),
1001         color=guide_legend(order=2, title="Fold change direction", title.position="top"),
1002         linetype=guide_legend(title="Fold change source", title.position="top", reverse=TRUE),
1003         pch=guide_legend(title="Fold change source", title.position="top", reverse=TRUE)) +
1004  theme(legend.position="top", legend.key=element_blank(), legend.title=element_text(size=8),
1005        legend.text=element_text(size=8),
1006        axis.title.x=element_blank(), axis.title.y=element_blank(),
1007        axis.text.y=element_text(size=8),
1008        strip.text=element_text(size=8),
1009        strip.background=element_rect(fill='gray90', color='gray'),
1010        strip.placement="outside", panel.spacing.y=unit(0.5, "lines"))
1011 ggsave(
1012 'PDShotgunAnalysis_out/5.Gene_pathway_associations/KO_pathway_distributions_foldchanges.pdf',
1013 g, device='pdf', width=12, height=20)

```

## Secondary analyses (replication driven)

The following analyses were conducted for the purposes of replicating prior 16S results.

### Replicating signal for SILVA v 132 *Prevotella*

To test if *Prevotella* species previously classified into the sub-genus “*Prevotella*” in SILVA v 132 (used in previous 16S analysis from *Wallen et al. 2020 npj Parkinsons Dis*) would be significantly associated with PD as a group in this dataset, collapsed relative abundances of *Prevotella* species in this dataset that were mapped to ASVs in *Prevotella* sub-genus of *Wallen et al. 2020 npj Parkinsons Dis* datasets (*P. buccalis*, *P. timonensis*, *P. bivia*, *P. disiens*, and *P. oralis*) into one group and tested for association with PD using linear regression (via `lm` function) with `log2` transformed relative abundances (as done with `MaAsLin2`).

```
1 ##### REPLICATING SILVA V 132 PREVOTELLA #####
2
3 # define target species to be collapsed
4 target_taxa <- c("Prevotella_buccalis","Prevotella_timonensis",
5                 "Prevotella_bivia","Prevotella_disiens","Prevotella_oralis")
6
7 # collapse species relative abundances into one group
8 ra.mod <- data.frame(otu_table(ra.ps.s)/100)
9 colnames(ra.mod) <- sapply(strsplit(as.character(colnames(ra.mod)), "s__"), function(x){x[2]})
10 ra.mod <- data.frame(target_cluster=rowSums(ra.mod[,colnames(ra.mod) %in% target_taxa]),
11                    ra.mod[,!(colnames(ra.mod) %in% target_taxa)])
12
13 # log2 transform
14 log2.ra <- data.frame(apply(ra.mod, 2, log2.trans))
15
16 # perform linear regression
17 ra.lm <- lm(log2.ra$target_cluster ~ Case_status + collection_method + seqs_scaled,
18            data=data.frame(sample_data(ra.ps.s)))
19
20 # coalesce results
21 FC <- paste(round(2^summary(ra.lm)$coefficients[2,1],2), ' [',
22            round(2^confint(ra.lm)[2,1],2), '- ', round(2^confint(ra.lm)[2,2],2), ']', sep='')
23 mod.results <- data.frame(Grouping="Prevotella (SILVAv132)",
24                            `N PD`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1] > 0),
25                            `N NHC`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0] > 0),
26                            `RA in PD`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1]),
27                                                format='e',digits=1),
28                            `RA in NHC`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0]),
29                                                format='e',digits=1),
30                            Beta=round(summary(ra.lm)$coefficients[2,1],2),
31                            SE=round(summary(ra.lm)$coefficients[2,2],2),
32                            P=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
33                            FC=FC,
34                            check.names=FALSE)
35
36 # write results
37 # create workbook
38 wb <- createWorkbook()
39 # add worksheet, write data, and format output
40 addWorksheet(wb, 'Prevotella (SILVAv132)')
```

```

41 writeData(wb, 'Prevotella (SILVAv132)', mod.results, keepNA=TRUE, colNames=TRUE)
42 setColWidths(wb, 'Prevotella (SILVAv132)', cols=seq_len(ncol(mod.results)),
43             widths=c(18, rep(7,7), 13)) ### format cells
44 addStyle(wb, 'Prevotella (SILVAv132)', cols=seq_len(ncol(mod.results)),
45         rows=1:(nrow(mod.results)+1), gridExpand=TRUE, style=center, stack=TRUE)
46 addStyle(wb, 'Prevotella (SILVAv132)', cols=seq_len(ncol(mod.results)),
47         rows=1, style=bold, stack=TRUE) ### font
48 addStyle(wb, 'Prevotella (SILVAv132)', cols=seq_len(ncol(mod.results)),
49         rows=c(1,2,(nrow(mod.results)+2)), ### borders
50         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
51 # convert numbers from strings back to numbers
52 convertNum(mod.results, wb, 'Prevotella (SILVAv132)', TRUE)
53 # save workbook
54 saveWorkbook(wb,
55             'PDSshotgunAnalysis_out/6.Secondary_analyses/Prevotella_SILVA_v_132.xlsx',
56             overwrite=TRUE)

```

## Replicating signal for opportunistic pathogen cluster

After correlation networks were constructed, we noted a poly-microbial cluster of species (cluster 17 in the PD network) that resembled that of a cluster of opportunistic pathogens noted in our previous 16S analysis (*Wallen et al. 2020 npj Parkinsons Dis*). As done in *Wallen et al. 2020*, to test if all of the species in this cluster are significantly enriched in PD (since only *P. asaccharolytica* was the only member of this cluster prevalent enough to be tested in MWAS), collapsed relative abundances of cluster members (as shown in PD network cluster 17) and tested for association with PD using linear regression with log2 transformed relative abundances (as done in *MaAsLin2*) adjusting for total sequence count per sample and collection method. **\*\*Please note that the cluster numbers may be assigned differently by different operating systems, e.g 13 instead of 17**

```

1  ##### ASSOCIATION OF OPP. PATH. CLUSTER WITH PD #####
2
3  ##### FULL CLUSTER #####
4
5  # collapse species relative abundances into one group
6  ra.mod <- data.frame(otu_table(ra.ps.s)/100)
7  colnames(ra.mod) <- sapply(strsplit(as.character(colnames(ra.mod)), "s_"),
8                          function(x){x[2]})
9  ra.mod <- data.frame(target_cluster=rowSums(ra.mod[,colnames(ra.mod) %in%
10                                         pd.clusters$names[pd.clusters$membership == 17]]),
11                    ra.mod[,!(colnames(ra.mod) %in%
12                             pd.clusters$names[pd.clusters$membership == 17])])
13
14 # log2 transform
15 log2.ra <- data.frame(apply(ra.mod, 2, log2.trans))
16
17 # perform linear regression
18 ra.lm <- lm(log2.ra$target_cluster ~ Case_status + collection_method + seqs_scaled,
19            data=data.frame(sample_data(ra.ps.s)))
20
21 # coalesce results
22 FC <- paste(round(2^summary(ra.lm)$coefficients[2,1],2), ' [',
23            round(2^confint(ra.lm)[2,1],2), '-', round(2^confint(ra.lm)[2,2],2), ']', sep='')
24 mod.results <- data.frame(Grouping="All 19 species in cluster #17",
25                          `N PD`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1] > 0),

```

```

26     `N NHC`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0] > 0),
27     `RA in PD`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1]),
28                         format='e',digits=1),
29     `RA in NHC`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0]),
30                         format='e',digits=1),
31     Beta=round(summary(ra.lm)$coefficients[2,1],2),
32     SE=round(summary(ra.lm)$coefficients[2,2],2),
33     P=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
34     FC=FC,
35     check.names=FALSE)
36
37 #### REDUCED CLUSTER (no PD-associated species included) ####
38
39 # collapse species relative abundances into one group
40 ra.mod <- data.frame(otu_table(ra.ps.s)/100)
41 colnames(ra.mod) <- sapply(strsplit(as.character(colnames(ra.mod)), "s__"), function(x){x[2]})
42 ra.mod <- data.frame(target_cluster=rowSums(ra.mod[, (colnames(ra.mod) %in%
43                                             pd.clusters$names[pd.clusters$membership == 17]) &
44                                             !(colnames(ra.mod) %in%
45                                             nodes.s$Id[grep("Yes", nodes.s`PD-associated`)]))],
46                      ra.mod[,!(colnames(ra.mod) %in%
47                                pd.clusters$names[pd.clusters$membership == 17]) |
48                                colnames(ra.mod) %in%
49                                nodes.s$Id[grep("Yes", nodes.s`PD-associated`)]])
50
51 # log2 transform
52 log2.ra <- data.frame(apply(ra.mod, 2, log2.trans))
53
54 # perform linear regression
55 ra.lm <- lm(log2.ra$target_cluster ~ Case_status + collection_method + seqs_scaled,
56            data=data.frame(sample_data(ra.ps.s)))
57
58 # coalesce results
59 FC <- paste(round(2^summary(ra.lm)$coefficients[2,1],2), ' [',
60            round(2^confint(ra.lm) [2,1],2), '-',round(2^confint(ra.lm) [2,2],2),']',sep='')
61 mod.results <- rbind(mod.results,
62                    data.frame(Grouping="Cluster #17 excluding P. asaccharolytica",
63                                `N PD`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1] > 0),
64                                `N NHC`=sum(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0] > 0),
65                                `RA in PD`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 1]),
66                                                    format='e',digits=1),
67                                `RA in NHC`=formatC(mean(ra.mod$target_cluster[sample_data(ra.ps.s)$Case_status == 0]),
68                                                    format='e',digits=1),
69                                Beta=round(summary(ra.lm)$coefficients[2,1],2),
70                                SE=round(summary(ra.lm)$coefficients[2,2],2),
71                                P=formatC(summary(ra.lm)$coefficients[2,4],format='e',digits=1),
72                                FC=FC,
73                                check.names=FALSE))
74
75 #### P. ASACCHAROLYTICA ONLY ####
76
77 # perform linear regression
78 ra.lm <- lm(log2.ra$Porphyromonas_asaccharolytica ~ Case_status + collection_method + seqs_scaled,
79            data=data.frame(sample_data(ra.ps.s)))

```

```

80
81 # coalesce results
82 FC <- paste(round(2^summary(ra.lm)$coefficients[2,1],2), ' ',
83             round(2^confint(ra.lm)[2,1],2), '-', round(2^confint(ra.lm)[2,2],2), ']', sep='')
84 mod.results <- rbind(mod.results,
85 data.frame(Grouping="P. asaccharolytica only",
86 `N PD`=sum(ra.mod$Porphyromonas_asaccharolytica[sample_data(ra.ps.s)$Case_status == 1] > 0),
87 `N NHC`=sum(ra.mod$Porphyromonas_asaccharolytica[sample_data(ra.ps.s)$Case_status == 0] > 0),
88 `RA in PD`=formatC(mean(ra.mod$Porphyromonas_asaccharolytica[sample_data(ra.ps.s)$Case_status == 1]),
89                    format='e', digits=1),
90 `RA in NHC`=formatC(mean(ra.mod$Porphyromonas_asaccharolytica[sample_data(ra.ps.s)$Case_status == 0]),
91                    format='e', digits=1),
92 Beta=round(summary(ra.lm)$coefficients[2,1],2),
93 SE=round(summary(ra.lm)$coefficients[2,2],2),
94 P=formatC(summary(ra.lm)$coefficients[2,4], format='e', digits=1),
95 FC=FC, check.names=FALSE))
96
97 # write results
98 # create workbook
99 wb <- createWorkbook()
100 # add worksheet, write data, and format output
101 addWorksheet(wb, 'Cluster 17')
102 writeData(wb, 'Cluster 17', mod.results, keepNA=TRUE, colNames=TRUE)
103 setColWidths(wb, 'Cluster 17', cols=seq_len(ncol(mod.results)),
104             widths=c(24, rep(7,7), 13)) ### format cells
105 addStyle(wb, 'Cluster 17', cols=seq_len(ncol(mod.results)),
106         rows=1:(nrow(mod.results)+1), gridExpand=TRUE, style=center, stack=TRUE)
107 addStyle(wb, 'Cluster 17', cols=seq_len(ncol(mod.results)),
108         rows=1, style=bold, stack=TRUE) ### font
109 addStyle(wb, 'Cluster 17', cols=seq_len(ncol(mod.results)),
110         rows=c(1,2,(nrow(mod.results)+2)), ### borders
111         gridExpand=TRUE, style=horizontal_border_med, stack=TRUE)
112 # convert numbers from strings back to numbers
113 convertNum(mod.results, wb, 'Cluster 17', TRUE)
114 # save workbook
115 saveWorkbook(wb,
116             'PDSHOTGUNAnalysis_out/6.Secondary_analyses/Cluster_17_PDvsNHC.xlsx',
117             overwrite=TRUE)

```

## R session information

```

1 ##### R SESSION INFO #####
2
3 sessionInfo()

## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Red Hat Enterprise Linux
##
## Matrix products: default
## BLAS/LAPACK: /data/rc/apps/rc/software/OpenBLAS/0.2.20-GCC-6.4.0-2.28/lib/libopenblas_haswellp-r0.2.
##
## locale:

```

```

## [1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C
## [3] LC_TIME=en_US.UTF-8          LC_COLLATE=en_US.UTF-8
## [5] LC_MONETARY=en_US.UTF-8      LC_MESSAGES=en_US.UTF-8
## [7] LC_PAPER=en_US.UTF-8         LC_NAME=C
## [9] LC_ADDRESS=C                  LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8   LC_IDENTIFICATION=C
##
## attached base packages:
## [1] grid      stats      graphics  grDevices  utils      datasets  methods
## [8] base
##
## other attached packages:
## [1] igraph_1.2.6      Maaslin2_1.8.0    ANCOMBC_1.6.2     vcd_1.4-9
## [5] pairwiseCI_0.1-27 coin_1.4-2         survival_3.2-13   MCPAN_1.1-21
## [9] vegan_2.5-7       lattice_0.20-45   permute_0.9-7     ggrepel_0.9.1
## [13] ggvenn_0.1.9      ggfortify_0.4.14  ggh4x_0.2.1       ggplot2_3.3.6
## [17] scales_1.2.1      gridExtra_2.3     data.table_1.14.2 foreach_1.5.2
## [21] openxlsx_4.2.5    tibble_3.1.8      phyloseq_1.34.0   readxl_1.3.1
## [25] reshape2_1.4.4    dplyr_1.0.9
##
## loaded via a namespace (and not attached):
## [1] backports_1.4.1      Hmisc_4.6-0        plyr_1.8.6
## [4] splines_4.0.3       TH.data_1.1-0      lpsymphony_1.18.0
## [7] digest_0.6.29       htmltools_0.5.2    fansi_1.0.3
## [10] magrittr_2.0.3      checkmate_2.0.0    cluster_2.1.2
## [13] doParallel_1.0.17   Biostrings_2.58.0  matrixStats_0.61.0
## [16] MCMCpack_1.6-1      sandwich_3.0-1     jpeg_0.1-9
## [19] colorspace_2.0-3    rbibutils_2.2.7    xfun_0.29
## [22] crayon_1.5.0        jsonlite_1.8.0     libcoin_1.0-9
## [25] biglm_0.9-2.1       Exact_3.1           zoo_1.8-9
## [28] iterators_1.0.14    ape_5.6-1          glue_1.6.2
## [31] gtable_0.3.0        zlibbioc_1.36.0    XVector_0.30.0
## [34] MatrixModels_0.5-0 Rhdf5lib_1.12.1    DEoptimR_1.0-10
## [37] BiocGenerics_0.36.1 abind_1.4-5         SparseM_1.81
## [40] mvtnorm_1.1-3       DBI_1.1.2           rngtools_1.5.2
## [43] Rcpp_1.0.8          htmlTable_2.4.0    magic_1.6-0
## [46] foreign_0.8-82      proxy_0.4-26        Formula_1.2-4
## [49] stats4_4.0.3        getopt_1.20.3       htmlwidgets_1.5.4
## [52] RColorBrewer_1.1-3 modeltools_0.2-23   pkgconfig_2.0.3
## [55] nnet_7.3-17         utf8_1.2.2          tidyselect_1.1.2
## [58] rlang_1.0.4         munsell_0.5.0       cellranger_1.1.0
## [61] tools_4.0.3         cli_3.3.0           generics_0.1.3
## [64] ade4_1.7-18         evaluate_0.15       biomformat_1.18.0
## [67] stringr_1.4.0       fastmap_1.1.0       yaml_2.3.5
## [70] mcmc_0.9-7          knitr_1.37          robustbase_0.93-9
## [73] zip_2.2.0           purrr_0.3.4         rootSolve_1.8.2.3
## [76] nlme_3.1-155        doRNG_1.8.2         quantreg_5.88
## [79] mcprofile_1.0-1     compiler_4.0.3      rstudioapi_0.13
## [82] png_0.1-7           e1071_1.7-9         pcaPP_1.9-74
## [85] DescTools_0.99.44  stringi_1.7.6       gsl_2.1-7.1
## [88] Matrix_1.4-0        nloptr_1.2.2.2     microbiome_1.12.0
## [91] multtest_2.46.0     vctrs_0.4.1        pillar_1.8.1
## [94] lifecycle_1.0.1     rhdf5filters_1.2.1  optparse_1.7.1
## [97] Rdpack_2.1.4        lmtest_0.9-39      lmom_2.8

```



```
## [100] R6_2.5.1          latticeExtra_0.6-29 IRanges_2.24.1
## [103] gld_2.6.4           codetools_0.2-18   boot_1.3-28
## [106] energy_1.7-10       MASS_7.3-55        rhdf5_2.34.0
## [109] withr_2.5.0         multcomp_1.4-18    S4Vectors_0.28.1
## [112] mgcv_1.8-39         expm_0.999-6       parallel_4.0.3
## [115] quadprog_1.5-8      rpart_4.1.16       tidyr_1.2.0
## [118] coda_0.19-4         class_7.3-20       rmarkdown_2.11
## [121] Rtsne_0.15         Biobase_2.50.0     base64enc_0.1-3
```