Supp infor and Figure legends, including the main fig legend

Figure 2- A) Metahit BGC distribution per cohort , B) Taxonomy results (Breakdown of bacteria) associated with the identified BGCs C) Metahit Lefse CD vs C results, 120 pvalue .01 and LDA >2

Figure 3 C) Lefse CD vs C iHMP2

Supp 2 : A) iHMP BGC distribution B) BGC per phylum

Supp1 A) Heatmap MetaHIT CD vs HC, B) Metahit Lefse results in HC vs UC

***12.2. Identification module***

***antiSMASH, parsing, classification of BGCs per the rules we decided, de-replication, species identification (parameters for blast, etc.)***

To identify biosynthetic gene clusters (BGCs), the identification module starts with deploying antiSMASH v3.0 with the following parameters: “—smcogs --clusterblast” on contigs that are 5 KB or longer for each SPAdes assembly for a given sample. antiSMASH will output a GenBank file, identifying, annotating, and comparing identified BGCs against a database of antiSMASH-predicted BGCs. Using the GenBank file from each antiSMASH run, we used Python and Python module Biopython to parse out each identified BGCs and their nucleotide sequences to create a master FASTA file of BGCs for a single sample. Each master BGCs FASTA file per sample is then combined with others from the same cohort to produce a master FASTA file of BGCs for a given cohort.

To remove duplicated BGCs identified from different samples, we launched blastn on our master BGCs FASTA file against itself. We produced a network from the Python module networkx based on the blastn results with BGCs as nodes and edges connecting the nodes at 95% percent identity and 95% coverage. For each hub in the network, we chose a representative BGC based on the longest in sequence length. We combined these representative BGCs with the BGCs that had no BLAST hits using our cutoffs (95% percent identity and 95% coverage) to create our de-replicated BGC FASTA file for each cohort.

To determine the taxonomic identity of the classified BGCs, we used blastn to query the nucleic acid sequences of the identified, de-replicated BGCs against all genomic sequences from NCBI RefSeq project (**DATE**). We ran blastn with the following parameters: “-perc\_identity 90 -max\_target\_seqs 500”. In addition, the BGC must have a coverage of >=50%. If multiple hits passed our parameter cutoffs, we filtered first by taking the hit with the highest coverage, then highest bitscore, and if there are still ties we assigned the first hit to appear in the BLAST results to the BGC. If a BGC has no BLAST hits or hits that passed our cutoff, we classified the taxonomy as `unassigned.`

To reduce the total number of BGC classes for plotting purposes, we grouped the de-replicated BGCs based on six (6) major BGC class: NRPS, Hybrid PKS-NRPS, PKS, RiPPs, Terpene, and Others. The NRPS class membership included: "nrps", "nrps-bacteriocin", "nrps-ladderane", "ladderane-nrps", "nrps-lantipeptide", "arylpolyene-nrps". The Hybrid PKS-NRPS class membership included: “nrps-t1pks" and "transatpks-otherks-nrps". The PKS class membership included: "t2pks", "transatpks", "t1pks", "lantipeptide-t2pks", "otherks", "otherks-transatpks", "otherks-t1pks". The RiPPS class membership included: "microcin", "bacteriocin", "lantipeptide", "lassopeptide", "bacteriocin-lantipeptide", "sactipeptide", "thiopeptide", "bacteriocin-proteusin", "glycocin", "microcin-lassopeptide", "proteusin-bacteriocin". The Terpene class membership comprised solely of “terpene”. Lastly, the Others class membership included: "arylpolyene", "other", "resorcinol", "butyrolactone", "ladderane", "siderophore", "hserlactone", "bacteriocin-arylpolyene", "amglyccycl", "ladderane-arylpolyene","phenazine".

***12.3. Quantification module***

***Mapping BGCs to all samples again, 50% coverage, rpkm, KW, LeFSe parameters, ML details,***

To quantify the abundance and prevalence of the BGCs discovered in the identification module, we queried the nucleic acid sequences of each BGC against the metagenomic filtered sequence reads for all samples using blastn with the following parameters: “-perc\_identity 95 -max\_target\_seqs 1000000”. This comprehensive method allowed us to detect BGCs even in samples where they are at very low abundance.

Using the results from the blastn search, we ran our quantifier algorithm written in Python to quantify the abundance of a given BGC per sample. Quantification of a given BGC in a sample involves calculating abundance with RPKM (Reads Per Kilobase Million) and coverage, the total percentage of a given BGC that reads of a sample have mapped to. Subsequently, we implemented a sample read coverage filtering parameter based on the positional mapping of the read in relation to the BGC. If a sample read partially maps to a given BGC at either end of the start or end coordinates, then we require the read to map 50% or more to a given BGC. Whereas, if the read maps within the start and end coordinates of a BGC, we require a 90% or more coverage. To calculate RPKM of a given BGC(i), we need to account for variation in sequencing depth and BGCs length. The equation below describes how RPKM is calculated,

totalHitsPerBGC, is the total number of reads mapped to BGC(i) at 95% percent identity with our positional read coverage cutoffs. This number is normalized by the length of BGC(i) per kilobase and the total number of the sample’s filtered reads per million. Coverage for a given BGC(i) is calculated using the equation below,

Furthermore, we considered a BGC to be present in a sample with a coverage of 50% or larger during the quantification module.

To determine differences in the RPKM abundances of IBD compared to healthy subjects in MetaHIT and iHMP, LefSe was ran on the resulting quantification module results using the “-o 1000000” flag with an alpha cutoff of 0.01 and an effect size cutoff of 2.0. Because LefSe only reports p-values of significant features, we calculated p-values for all features using Kruskal-Wallis test (kruskal.test) with normalized RPKM values.

ROC curves were constructed to evaluate the performance of Random Forest classifier using Leave-One-Out Cross Validation (LOOCV) with 'class\_weight' set to balanced aiming to identify the IBD status of a subject based on an individual’s biosynthetic gene cluster profile. The Random Forest classifier is implemented using RandomForestClassifier module in scikit-learn package. Whereas the LOOCV methodology is made available through the LeaveOneOut module in scikit-learn package. We hyper-parameter tuning the following parameters with the GridSearchCV module available from scikit-learn package: 'n\_estimators', 'max\_features', 'max\_depth', and 'criterion.' We have checked the performance for the classifier trained by samples from two different cohorts (MetaHIT and iHMP). In both cohorts, BGC profiles are summarizes as the RPKM of reads mapped to a BGC and normalized both by sample's total read count and BGC length.

Random forest was used to identify BGCs that best discriminated between healthy and IBD individuals in Spanish and American adults.

LOOV cross validation

**Software and Algorithms**

Identification and quantification module This paper https://github.com/donia-lab/smNRPS-IBD

R 3.4.4 R Development Core Team, 2017

<https://www.R-project.org>

ggplot2 (v2.2.1) Wickham, 2009

<https://cran.r-project.org/web/packages/ggplot2/index.html>

tidyverse (v1.2.1) Wickham, 2017

<https://cran.r-project.org/web/packages/tidyverse/index.html>

antiSMASH (v3.0.5) Weber et al., 2015 <https://antismash.secondarymetabolites.org>

SPAdes (v3.11.0) Bankevich et al., 2012 http://cab.spbu.ru/software/spades/

BLAST 2.7.1+ Camacho et al., 2009 ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/.

PRINSEQ-lite v0.20.4B Schmieder R. and Edwards R., 2011 http://prinseq.sourceforge.net/

fq2fa1.1.1 (The SRA Toolkit) https://github.com/ncbi/sra-tools

LEfSe (Galaxy Tool Version:1.0) Segata et al., 2011 http://huttenhower.sph.harvard.edu/galaxy/

metaphlan2 v2.6.0 Truong et al., 2015

https://bitbucket.org/biobakery/metaphlan2

Python v3.6.3 Python Core Team, 2015 https://www.python.org/

ggpubr v0.1.6 Kassambara,2017 https://cran.r-project.org/web/packages/ggpubr/index.html

Pandas McKinney,2010 v0.22.0 https://pypi.org/project/pandas/#files

scikit-learn v0.19.1 Pedregosa et al., 2011 http://scikit-learn.org/stable/

NumPy v1.14.2 Oliphant, 2006 http://www.numpy.org/

Biopython v1.72 Cock et al., 2009 https://biopython.org/

NetworkX v2.1 Hagberg A.A., Schult D.A., and Swart P.J., 2008

https://networkx.github.io/

pheatmap v1.0.10 Kolde,2018 https://cran.r-project.org/web/packages/pheatmap/index.html

Matplotlib Hunter,2007 https://matplotlib.org/

Code Availability

All the computational analyses were performed in Python version 3.6.3, apart from the transcriptomics RNA-seq processing which as done in R version 3.3.1 with Limma package version 3.28.21 and edgeR 3.14.0, and are available under GNU General Public License V3 in a GitHub project in the following url <https://github.com/saezlab/protein_attenuation>. Plotting was done using Python modules Matplotlib version 1.4.3 ([Hunter, 2007](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5660600/#bib18)) and Seaborn version 0.7.0. Generalised linear models were built using Python module Sklearn version 0.17.1 ([Pedregosa et al., 2011](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5660600/#bib36)). Data analysis and structuring was carried out using Python module Pandas version 0.18.1 ([McKinney, 2010](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5660600/#bib30)).

Random forest was used to identify those BGCs that best discriminated between healthy and Crohn's Disease individuals in Spanish adults. Samples membership in these two classes could be predicted with

ROC analysis. ROC curves are constructed to evaluate the performance of Random Forest classifier aiming to identify the IBD status of a subject based on his or her biosynthic gene cluster profile. We have checked the performance for classifier trained by samples from two different cohorts (MetaHIT and iHMP). In MetaHIT, BGC profiles are summarizes as the RPKM of reads mapped to a BGC and normalized both by sample's total read count and BGC length. In iHMP, BGC profiles are summarized as in MetaHIT with the additional of averaging the RPKM of

ROC curves are gained per site using 5-fold cross-validation. The IBD cases proportion in each subsample is set as the same with the overall proportion. Each curve is calculated based on the predicted probabilities of being IBD given by the classifier trained by the 4 out of 5 subsamples for the rest one subsample. A mean ROC curve is then given by averaging over all 5 individual fold ROC curves and an approximated 95% pointwise confidence interval is also constructed by using normal approximation and the sample means and variances. The sparse logistic regression is implemented using LogisticRegression module in scikit-learn package (http://www.csie.ntu.edu.tw/~cjlin/liblinear/).

The BGC profiles are summarized as the raw counts of genus-level taxa detected in at least one sample and these counts are further normalized per sample by the total counts of all taxa in that sample so that the resulting relative abundances sum up to 1.