Supplemental Material:

Resource heterogeneity structures aquatic bacterial communities

Mario E. Muscarella, Claudia M. Boot, Corey D. Broeckling, Jay T. Lennon

DOM Annotations — The following parameters were used when annotating DOM components using RAMClustR (Broeckling et al., 2014):


2) MSFinder formula results were imported with database priority set to: UNPD MINE ChEBI T3DB STOFF HMDB KNApSAcK FooDB NAPDDB DrugBank SMPDB YMDB BMDB ECMDB PubChem PlantCyc Urine Serum LipidMAPS Feces NA CSF Saliva

3) RAMClustR *annotate* function annotation priority was assigned from highest priority to lowest: MSFinder structure, MSFinder formula, interpretMSSpectrum

4) Synonyms were retrieved using the RAMClustR *get.synonyms* function within RAMClustR, calling the Chemical Translation Service (Wohlgemuth *et al.*, 2010) as well as database API for Pubchem (Kim *et al.*, 2019), LipidMaps (Fahy *et al.*, 2007), and HMDB (Wishart *et al.*, 2018), when appropriate
**Sequencing Results** — After quality controls and removing singletons, we had recovered the following numbers of sequences from each site based on the active community:

Ann Lake - 96963
Canyon Lake - 79973
Howe Lake - 64182
Ives Lake - 101741
Lily Lake - 150702
Mountain Lake - 96933
Pony Lake - 127352
Rush Lake - 84750
Second Pine Lake - 89873
Upper Pine Lake - 111968
**Transformed Data Table** — We Box-Cox transformed resource concentration and OTU richness to meet the model assumptions regarding the distribution of residuals. We used a Shapiro Test (*shapiro.test* function in the stats R package) to determine while model parameters were significantly different from normal and therefore needed to be transformed.

DOC: $W = 0.63753$, p-value $= 0.0001555$

OTU Richness: $W = 0.69123$, p-value $= 0.0006898$

Transformed values have been Box-Cox transformed and also centered and scaled

<table>
<thead>
<tr>
<th>Lake</th>
<th>DOC (mg C L$^{-1}$)</th>
<th>Transformed DOC</th>
<th>Rarified OTU Richness</th>
<th>Transformed OTU Richness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann</td>
<td>5.97</td>
<td>-0.44</td>
<td>933</td>
<td>0.32</td>
</tr>
<tr>
<td>Canyon</td>
<td>7.23</td>
<td>0.08</td>
<td>1046</td>
<td>0.69</td>
</tr>
<tr>
<td>Howe</td>
<td>7.04</td>
<td>0.02</td>
<td>808</td>
<td>-0.34</td>
</tr>
<tr>
<td>Ives</td>
<td>6.91</td>
<td>-0.02</td>
<td>766</td>
<td>-0.65</td>
</tr>
<tr>
<td>Lily</td>
<td>14.35</td>
<td>1.25</td>
<td>1710</td>
<td>1.47</td>
</tr>
<tr>
<td>Mountain</td>
<td>5.27</td>
<td>-0.86</td>
<td>830</td>
<td>-0.20</td>
</tr>
<tr>
<td>Pony</td>
<td>28.99</td>
<td>1.77</td>
<td>2344</td>
<td>1.64</td>
</tr>
<tr>
<td>Rush</td>
<td>4.22</td>
<td>-1.77</td>
<td>706</td>
<td>-1.22</td>
</tr>
<tr>
<td>Second Pine</td>
<td>6.26</td>
<td>-0.30</td>
<td>743</td>
<td>-0.85</td>
</tr>
<tr>
<td>Upper Pine</td>
<td>7.84</td>
<td>0.28</td>
<td>739</td>
<td>-0.88</td>
</tr>
</tbody>
</table>
**Total Bacterial Community** — In addition to sequencing the active bacterial community, we sequenced the total community. We used a DNA based approach to characterize the composition of the total by sequencing the 16S rRNA gene. We extracted total nucleic acids using the MoBio Power Water RNA extraction kit (Carlsbad, CA). Nucleic acid extracts were cleaned via ethanol precipitation and DNA extracts were treated with RNase (Invitrogen) to degrade residual RNA. Once DNA samples were cleaned and quantified, we amplified, sequences, and processed the data in the exact same pipeline as the RNA based approach. In addition, both the total and active community samples were clustered into OTUs at the same time so that OTUs could be compared. All initial sequence processing was completed using the software package mothur (version 1.40.5; Schloss et al., 2009).
Total Community and Resource-Diversity Relationship — First, we tested the hypothesis that resource heterogeneity affects bacterial community evenness to compare with the results from the active community. We used linear models to determine if the number of DOM components would affect the evenness of bacterial communities. We removed singleton OTUs (i.e., those found only once across all samples), and we subsampled using rarefaction to correct for differences in sample size due to sequencing depth (Hughes and Hellmann, 2005; James and Rathbun, 1981). We calculated species evenness using Simpson’s evenness (Smith and Wilson, 1996). We calculated the number of DOM components as the number of distinct DOM peaks observed in each sample. As predicted and observed based on the active microbial community, bacterial evenness was affected by the number of DOM components. OTU evenness was positively related to the number of DOM components ($r^2 = 0.58, p = 0.01$).
Next, we tested the hypothesis that DOM heterogeneity affects community beta-diversity by comparing DOM composition to total bacterial community composition. We used distance-based redundancy analysis (dbRDA; Legendre and Anderson, 1999) to test for relationships between resource composition and community composition, and we used principal coordinates analysis (PCoA), based on relative abundances and Bray-Curtis dissimilarity, to decompose DOM composition into orthogonal linear components (Legendre and Legendre, 2012). To represent the DOM composition, we used the DOM PCoA axis scores for each sample. As the response in the dbRDA model, we relativized OTU abundances and used Bray-Curtis distances to compare community composition across samples. Significance tests of our dbRDA model were conducted based on 10,000 permutations. All calculations were done in the R statistical environment (R Core Team, 2012) using the ‘vegan’ package (Oksanen et al., 2013). As predicted and observed based on the active microbial community, we observed relationships between DOM composition and the beta-diversity of the total community. Based on the dbRDA models, DOM composition explained 45% of the variation in bacterial community composition ($p = 0.03$).
**Supplemental Tables**

**Table S1.** DOM Component Annotations. Annotations are provided for the 172 influential DOM components. DIM1 and DIM2 refer to the contribution of the DOM component to the DOM Principal Coordinates Analysis (Fig. 1; Fig. S3). Each component includes its retention time, an inferred molecular mass and formula, full elemental stoichiometry (number of C, H, N, O, P, and S atoms per molecule) and stoichiometric ratio (C:N:P). In addition, we provide some molecular summary information including, H/C which indicates the amount of hydrogen saturation, O/C which describes the degree of oxygenation, N/C which describes the nitrogen content (Roth et al., 2015). In addition, we provide the CHO index which is a proxy for the degradation potential (values between -1 and 0 are thought to be highly degradable) (Mann et al., 2015), the nominal oxidation state of carbon (NOSC) which provides information on the biogeochemical reactivity (Riedel et al., 2012), double bond equivalents (DBE), the difference between DBE and number of oxygen atoms (DBE-O) which is a rough measure for C = C bonds because it omits any possible C = O bonds (D’Andrilli et al., 2010). In addition, components molecular annotations are provided and organized based on chemical ontology and a PubChem URL is provided if available.

**Table S2.** The complete list of cosmopolitan taxa including taxonomy. Each OTU is classified based on the RDP taxonomy down to genus. OTUs are listed in order of total abundance across all sites.

**Table S3.** The complete list of resource generalist taxa including taxonomy. Each OTU is classified based on the RDP taxonomy down to genus. OTUs are listed in order of total abundance across all sites.
Supplemental Figures

Figure S1. Study system map of the Huron Mountain area. Image retrieved from Google Maps. Lakes used in this study are labeled.
Figure S2. Resource Correlations. The concentrations of dissolved organic carbon (DOC) and total nitrogen (TN) are highly correlated across lakes (rho = 0.97, p < 0.001).
Figure S3. Principal coordinates analysis (PCoA) ordination of dissolved organic matter (DOM) with influential DOM components. We identified influential DOM components as those correlated (rho > 0.70) with variation in the DOM PCoA axes. The distances between symbols represent the dissimilarity in DOM composition between lakes. Symbol sizes reflect variation in the concentration of dissolved organic carbon. Red vectors represent the correlation between each influential DOM component and the PCoA axes.
Figure S4. Bacterial community diversity relationships with resource concentration and the number of DOM components using untransformed data.
**Figure S5.** Bacteria-DOM Interactions. Interactions were calculated based on Spearman’s Rank-Order Regressions between bacterial OTU and DOM component abundances across sites. Significant interactions were determined using a randomization approach, and only significant interactions (i.e., those with $p < 0.05$ based on 1000 randomizations) with coefficients of determination $> |0.7|$ were considered. Interactions are ordered across columns and rows based on the number of interactions present. Therefore, rows are ordered from generalist (most interactions; on right) to specialist (least interactions; on left) – Consumer Strategy. Likewise, columns are ordered from available (most interactions; on top) to restrictive (least interactions; on bottom) – Resource Availability.
References


Mothur Pipeline

make.contigs(file=HMWF.files, insert=30, deltaq=6, processors=8)
summary.seqs(fasta=HMWF.trim.contigs.fasta, processors=8)
screen.seqs(fasta=HMWF.trim.contigs.fasta, group=HMWF.contigs.groups, summary=HMWF.trim.contigs.summary, maxambig=0,maxlength=275, processors=8)
remove.seqs(accnos=HMWF.trim.contigs.bad.accnos, group=HMWF.contigs.groups)
system(mv HMWF.contigs.pick.groups HMWF.contigs.good.groups)
unique.seqs(fasta=HMWF.trim.contigs.good.fasta)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.fasta, processors=8)
count.seqs(name=HMWF.trim.contigs.good.names, group=HMWF.contigs.good.groups)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.fasta, count=HMWF.trim.contigs.good.count_table,processors=8)
align.seqs(fasta=HMWF.trim.contigs.good.unique.fasta, reference=silva.v4.fasta, flip=F,processors=8)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.align, processors=8)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.flip.accnos, processors=8)
screen.seqs(fasta=HMWF.trim.contigs.good.unique.align, count=HMWF.trim.contigs.good.count_table, minlength=250, maxhomop=8,processors=8)
remove.seqs(accnos=HMWF.trim.contigs.good.unique.bad.accnos, count=HMWF.trim.contigs.good.count_table)
system(mv HMWF.trim.contigs.good.pick.count_table
HMWF.trim.contigs.good.good.count_table)
count.groups(count=HMWF.trim.contigs.good.good.count_table)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.align,
count=HMWF.trim.contigs.good.good.count_table, processors=8)

screen.seqs(fasta=HMWF.trim.contigs.good.unique.good.align,
count=HMWF.trim.contigs.good.good.count_table, maxambig=0, maxhomop=8,
minlength=250, start=1965, end=11553, processors=8)

remove.seqs(accnos=HMWF.trim.contigs.good.unique.good.bad.accnos,
count=HMWF.trim.contigs.good.good.count_table)

system(mv HMWF.trim.contigs.good.good.pick.count_table
HMWF.trim.contigs.good.good.good.count_table)

summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.align,
count=HMWF.trim.contigs.good.good.good.count_table, processors=8)

filter.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.align, vertical=T, trump=.)

summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.fasta,count=HMWF.trim.contigs.good.good.good.count_table, processors=8)

unique.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.fasta,
count=HMWF.trim.contigs.good.good.good.count_table)

summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.count_table, processors=8)

pre.cluster(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.count_table, diffs=2,
processors=8)

summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster
.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.count_table,
processors=8)
chimera.vsearch(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.count_table,
dereplicate=t, processors=8)
remove.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.fasta,
accnos=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.accnos)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.count_table, processors=8)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.count_table, processors=8)
classify.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.count_table, reference=trainset16_022016.pds.fasta,
taxonomy=trainset16_022016.pds.tax, cutoff=80, probs=f, processors=8)
remove.lineage(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.count_table,
taxonomy=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pds.wang.taxonomy, taxon=Archaea-Mitochondria-Cyanobacteria/Chloroplast;Chloroplast-unknown-Eukaryota)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table, processors=8)
classify.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table, reference=trainset16_022016.pds.fasta,
taxonomy=trainset16_022016.pds.tax, cutoff=95, probs=t, processors=8)
remove.lineage(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table,
taxonomy=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pds.wang.taxonomy, taxon=unknown)
summary.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table, processors=8)
classify.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table, reference=trainset16_022016.pds.fasta,
taxonomy=trainset16_022016.pds.tax, cutoff=95, probs=t, processors=8)
remove.lineage(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denvo.vsearch.pick.pick.count_table,
taxonomy=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pds.wang.taxonomy, taxon=unknown)
ch.pick.pick.pick.count_table, reference=trainset16_022016.pds.fasta,
taxonomy=trainset16_022016.pds.tax, cutoff=80, probs=t, processors=8)
cluster.split(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
taxonomy=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pds.wang.taxonomy,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.pick.pick.count_table, splitmethod=classify, taxlevel=4, method=opti, processors=8)
make.shared(list=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.opti_mcc.list,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.pick.pick.count_table, label=0.03)
classify.seqs(fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.pick.pick.count_table, reference=trainset16_022016.pds.fasta,
taxonomy=trainset16_022016.pds.tax, cutoff=25, probs=t, processors=8)
classify.otu(list=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.opti_mcc.list,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.pick.pick.count_table,
taxonomy=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pds.wang.taxonomy, label=0.03)
get.oturep(list=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pick.opti_mcc.list,
fasta=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.pick.pick.pick.fasta,
count=HMWF.trim.contigs.good.unique.good.good.filter.unique.precluster.denovo.vsearch.pick.pick.pick.count_table, label=0.03, method=abundance)