

Spherical: an iterative workflow for assembling metagenomic datasets

Authors: Thomas Hitch¹ and Christopher J Creevey¹

Affiliation: ¹Institute of Biological, Environmental and Rural Sciences (IBERS),
Aberystwyth University, Aberystwyth, SY23 3FG, UK.

Contact information: Chris Creevey, chc30@aber.ac.uk

Keywords: metagenome, *de novo* assembly

Abstract:

The consensus emerging from microbiome studies is that they are far more complex than previously thought, requiring deep sequencing. As deep sequenced datasets provide greater coverage than previous datasets, recovering a higher proportion of reads to the assembly is still a challenge. To tackle this issue, we set of to identify if multiple iterations of assembly would allow for otherwise lost contigs to be formed and studied and if so, how successful is such an avenue at improving the current methodology.

A simulated metagenomic dataset was initially used to identify if multiple iterations of assembly produce useable contigs or mis-assembled artefacts were produced. Once we had confirmed that the secondary iterations were producing both accurate contigs without a reduction in contig quality we applied this methodology in the form of *Spherical* to 3 metagenomic studies.

The additional contigs produced by *Spherical* increased the number of reads aligning to an identified gene by 11-109% compared to the initial iterations assembly. As the size of the dataset increased, as did the amount of data multiple iterations were able to add.

Availability

Spherical is implemented in Python 2.7 and available for use under a MIT licence agreement at: <https://github.com/thh32/Spherical>

29 **Introduction:**

30

31 Over the last 10 years researchers have utilised high-throughput sequencing to
 32 investigate the structure and function of increasing larger numbers of microbial
 33 communities from environments across the globe (Krohn-Molt et al. 2013; van der
 34 Lelie et al. 2012; Modi et al. 2013). While these studies have provided unique and
 35 novel insights into the workings of these communities, there is a growing consensus
 36 that the tools available are not capturing the full functional diversity of the data being
 37 generated (Nagarajan & Pop 2013). Increasing the proportion of genome assembled
 38 is a challenge, and resolving this issue is very needed, with the ever increasing
 39 amounts of sequencing data becoming available. Additionally, it would also enable us
 40 to retrieve further more from data already available in repositories.

41

42 Mathematically, *de novo* assembly of a genome falls within the class of problems for
 43 which no efficient algorithm is known (NP-hard) (Myers. 1995; Medvedev et al. 2007),
 44 leading to the proposal of a variety of heuristic solutions (Charuvaka & Rangwala
 45 2011). These have ranged from simple overlap-layout-consensus approaches, where
 46 sequencing reads with overlapping regions are joined together into contigs (Myers.
 47 1995) to more complex approaches such as de Bruijn graphs (Idury et al. 1995).
 48 Genomic assemblers based on de Bruijn graphs break each read into smaller strings
 49 of K length (kmers) and connections are made between overlapping kmers (Schatz
 50 et al. 2010). Paths of least resistance through the kmer graph represent contigs
 51 (Compeau et al. 2011).

52

53 Generally, these assemblers have been designed with a single-genome in mind
 54 where it could be guaranteed that all sequences generated belonged to the same
 55 species. However, as sequencing approaches for sampling the genomic information
 56 of entire microbial communities (metagenomics) began to emerge (Venter et al.

2004), it was clear that new approaches may be necessary (Lai et al. 2012). By far the biggest issue with metagenomic datasets stem from the uneven distribution of species in natural communities leading to varying depths of sequencing coverage of each (Nagarajan & Pop 2013). This leads to over-sequencing of dominant species in the community and results in heavily fragmented assemblies of the genomes of minority species, if they can be assembled at all (Bergeron et al. 2007).

Promising solutions to dealing with this problem use a 'divide and conquer' approach to break the data into more easily manageable pieces (Bergeron et al. 2007). For example the data from environmental samples can be split into "bins" representing different taxa from the community (Mohammed et al. 2011). Sequence reads can be sorted into bins based on properties such as kmer-frequency or the percentage of Guanine and Cytosines (GC) they contain (Dro & Mchardy 2012). This has the potential to increase the assembly rate of low abundance species, however it depends heavily on accurately partitioning the data (Wang et al. 2012). Indeed, bins of metagenomic data produced in this manner may represent a single species or an entire phylum depending on the complexity of the community (Wang et al. 2012).

Digital normalization is another method that tries to deal with unevenly sampled data (Brown & Crusoe 2014). In this approach, sequence reads are discarded based on the local coverage of their kmers (Brown & Crusoe 2014). This has the effect of reducing coverage of over-represented taxa, and "normalising" the coverage to make it more even (Brown & Crusoe 2014). While this pre-processing step allows for reduction of the datasets size, it does not reduce the complexity of metagenomes to be assembled.

Using a similar approach to 'divide-and-conquer' large genomic datasets, SLICEMBLER aims to improve deep sequenced (>800x coverage) single genome

85 assemblies (Mirebrahim et al. 2015). The approach taken here is to separate the
 86 input data into pre-determined "slices" based on coverage (Mirebrahim et al. 2015).
 87 Each slice is then assembled separately and frequently occurring strings (FOS) are
 88 identified between the sub-assemblies and merged, effectively scaffolding them
 89 together to produce a final assembly (Mirebrahim et al. 2015). This approach works
 90 well for deep sequenced genomic datasets where coverage is known, however in
 91 metagenomics datasets from uncharacterised microbial communities, coverage is
 92 generally unknown (Peng et al. 2012).

93

94 Whilst these approaches are an improvement upon single-genome based methods,
 95 they still do not generate an assembly that utilises 100% of the reads from the data
 96 or produce complete genomes without manual curation (Hess et al. 2011).

97

98 Here we propose an iterative workflow that tries to address this problem with
 99 metagenomic assemblies. Based on, and extending the 'divide-and-conquer'
 100 principle, the workflow (called "Spherical") can be applied to any assembly method.
 101 Spherical is designed to work with datasets that cannot be assembled in a single
 102 step due to data volume or complexity but could equally be applied to smaller and
 103 simpler datasets.

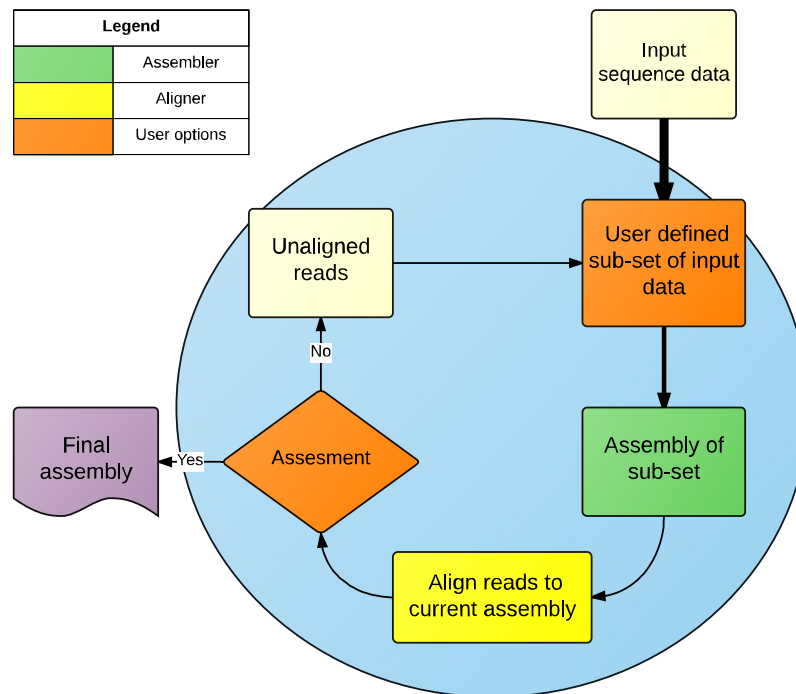
104

105 **Spherical workflow:**

106 Spherical uses an iterative workflow which tries to capture the reads not utilised by
 107 an assembly, and these are reapplied to subsequent rounds of assemblies. The
 108 outline of the approach is summarised in Figure 1.

109

110



111

112 **Figure 1:** A flow-chart of the steps used by *Spherical*. The all process within the blue
 113 circle occur within *Spherical* whilst the input and output files are outside. The width of
 114 the arrows indicate the possible decrease in file size depending on user sub-sample
 115 selection. The ‘user defined criteria’ is defined as any user option which indicates a
 116 point at which *Spherical* should stop iterating.

117

118 The *Spherical* workflow is composed of 5 steps:

119 **Step 1: Sub-sample selection**

120 The first step in *Spherical* is the optional initial sub-sampling of the sequencing data.

121 This can be advantageous when the working with very large datasets. In this process

122 a random sub-sample (defined by $-R$) is taken from the input sequencing data. Using

123 a subset fraction of 1 selects the entire input dataset instead. If only one value is

124 given to $-R$, then *Sphercial* will apply this sub-sample fraction at every iteration,

125 however the user also has the option of providing values to be used at each iteration.

126

127 **Step 2: Assembly**

128 The sub-sample is then assembled using the assembler of choice. Currently the
129 default assembler in Spherical is Velvet (Namiki et al. 2012).

130

131 **Step 3: Alignment**

132 When the assembly is completed Spherical uses Bowtie 2 (Langmead & Salzberg
133 2012) to align all reads previously unaligned (in iteration 1 this is all the reads) to the
134 contigs resulting from this assembly. All reads that do not align to the assembly
135 produced at this iteration are saved for the subsequent rounds. If a read aligns to the
136 assembly at this point, it is considered utilised and hence excluded from the
137 subsequent iterations.

138

139 **Step 4: Assessment**

140 The user can define two parameters to be used by Spherical to determine
141 completeness. The first is based on the number of iterations currently completed (-
142 iter). When spherical has completed all iterations defined by this value, it will halt
143 carrying out iterations, and move to step 5. The second option is based on the
144 proportion of reads currently utilised by the total assembly (-align). This is calculated
145 as the number of reads currently unaligned, divided by the total number of reads
146 initially provided. When Spherical determines that the alignment rate has been
147 reached it will halt carrying out iterations and move to step 5. The user must provide
148 these two parameters, the current default is 5 iterations or an alignment rate of 70%.
149 If neither of these criteria have been met, spherical will pass on all unaligned reads to
150 step 1 for another iteration.

151

152 **Step 5: Final output**

153 When Spherical has met the user-defined criteria for halting, Spherical will combine
154 all the assemblies from each iteration into a single file and calculates statistics such

155 as N50, lengths of longest and shortest contigs the standard deviation of the lengths
156 and the alignment rate both for each iteration and for the final assembly.

157

158 In the following sections we outline the principles behind Spherical and demonstrate
159 its use on three different metagenomic datasets of differing sizes and complexity. We
160 show that by taking an iterative approach the resulting assemblies use a greater
161 proportion of the original raw reads and in large datasets it allowed to retrieve more
162 information from the less-represented organisms in the community.

163

164 **Results**

165 **Simulated dataset analysis**

166 By using a simulated dataset (Mende et al. 2012) created from 400 species of
167 varying abundance we investigated the accuracy of assemblies from each iteration
168 from the Spherical workflow. We used BLASTN to identify the best matching genomic
169 region for each contig assembled. The quality of the reconstructed region was
170 assessed using the contig score (Mende et al. 2012) which calculates a value
171 representing the accuracy of the reconstruction. We found that there was no
172 observable decrease in the contig score as the number of iterations increased
173 (Supplementary Table 3).

174

175 **Metagenome dataset analysis**

176 We used three published metagenomic datasets (chicken ceacum (Qu et al. 2008),
177 human oral (Belda-Ferre et al. 2012) and groundwater from the Yucatan peninsula
178 (Moore et al. 2009)) of varying sizes to test Spherical (Table 2). For each we carried
179 out three assembly approaches, 1) Basic Velvet assembly, 2) Digital normalisation
180 followed by a Velvet assembly and 3) Spherical on the raw data with 5 iterations,
181 using Velvet as the assembler (Supplementary Table 1). We used the percentage of

the assembly to which no reads align as a measure of miss-assembly (the “false base rate”) and the percentage of reads that aligned as a measure of completeness.

Assembly quality for tested datasets

The Chicken Caecum microbiome was the smallest of the three datasets tested. As a result, all three assembly approaches produced very similar results (Supplementary Table 1). However the assembly from Spherical utilised 1% more of the raw reads than the other approaches. This was at the cost of slightly lowering the N50 (from 109 to 104) and increased false base rate (from 0.01% to 0.04%).

The human oral dataset was a larger dataset, and as a result we observed a greater variability in how the different assembly approaches performed. Spherical was able to increase both the N50 (from 190 to 234) and the alignment rate (from 13% to 24.6%) compared to the next best approach (basic velvet assembly), however the false base rate also increased (from 0.02% to 0.19%).

The groundwater dataset was the largest tested. The alignment rate of Spherical increased (from 52.8% to 59.7%) and false base rate decreased (from 3.86% to 2.89%), however *Sphericals* N50 was significantly reduced (from 330 to 211) in comparison to the normalised assembly.

Effect of sub-sample size on the resulting assembly

To study the effect of altering the sub-sample size we used the largest of the metagenomics datasets tested (Groundwater) and with sub-sample fractions of 1 (representing 1/1, i.e. all the raw data), 4 (representing ¼ of the raw sequencing reads) and 30 (representing 1/30 of the raw sequencing reads). As shown in Supplementary Table 1 the change in sub-sample size resulted in a small change in the quality of the resulting assembly; increasing the false base rate from 2.89% to 3.78%, and reducing the N50 from 211 to 189 and reducing the alignment rate from

59.7% to 49.8%. However even with these changes, the taxonomic profile of the each assembly did not differ, Supplementary Figure 1.

211

212 **Effect of multiple iterations of assembly**

213 The additional iterations employed by Spherical lead to an increase in the number of
214 reads that could be assigned to known genes (Figures 3,4 & 5). As shown in Figure 3
215 and 4, for small metagenomic datasets the taxonomic profile does not change across
216 iterations, however the iterative approach does allow for almost twice the number of
217 reads to be assigned to a taxonomic class (Table 1). In the groundwater dataset the
218 secondary iterations provided a different taxonomic profile compared to the initial
219 iteration (Figure 5).

220

221 **Table 1:** Number of reads assigned to an identified gene in each Spherical
222 assembly.

Dataset	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5
Cecum	164,266	11,317	3,398	2,309	1,427
Oral	126,388	43,721	33,110	30,907	30,032
Groundwater	130,451,683	27,463,983	12,121,835	8,586,974	7,783,534

223

224 As shown in Table 1, the additional iterations allowed for identification of 11%
225 additional reads in the chicken caecum dataset, 109% additional reads in the human
226 oral dataset and 43% additional reads in the groundwater dataset compared to the
227 first iterations assembly.

228

229

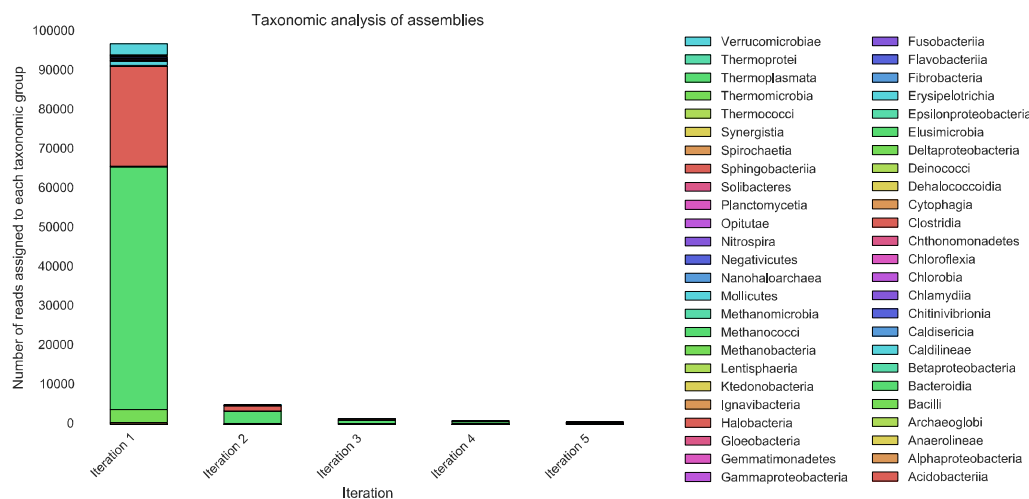


Figure 3: A taxonomic breakdown of each iteration of cecum dataset Spherical assembly at the class level.

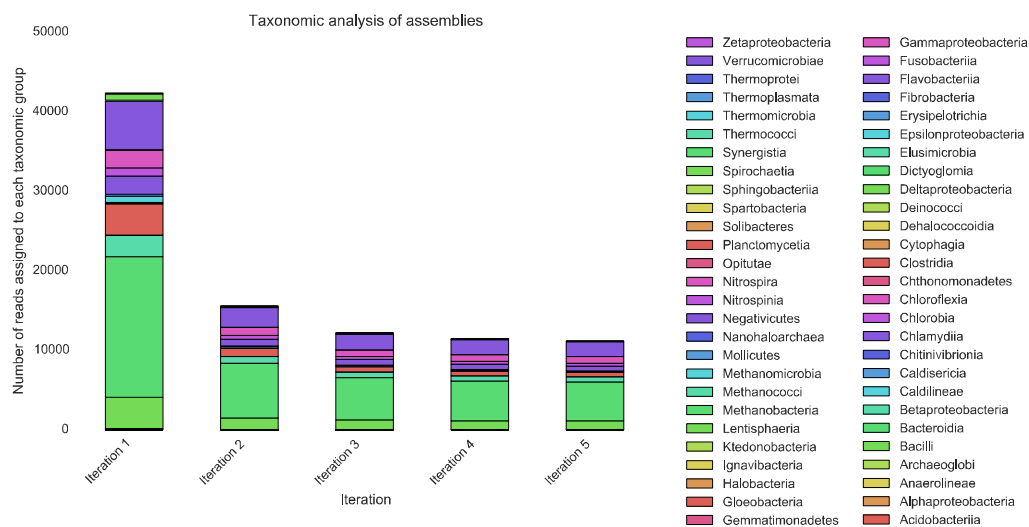


Figure 4: A taxonomic breakdown of each iteration of oral dataset Spherical assembly at the class level.

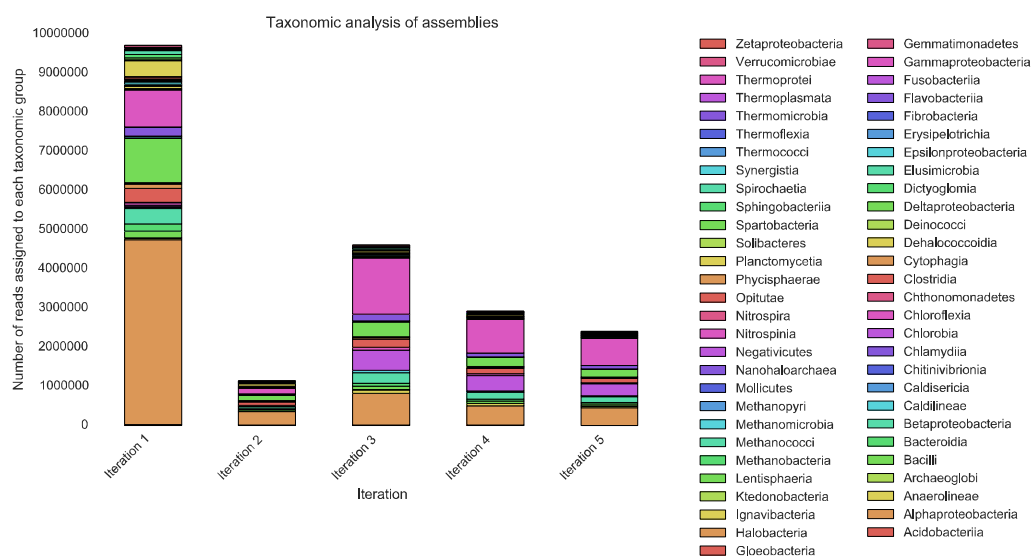


Figure 5: A taxonomic breakdown of each iteration of groundwater dataset *Spherical* (subset size = 1) assembly at the class level.

261

262

263

264

265

266

267

268

269

270

271

272

273 **Methods**

274 **Simulated dataset**

275 A simulated Illumina sequenced metagenome of 400 species, each species
276 abundance within the dataset differed to produce a more accurate representation of a
277 metagenomic dataset (Mende et al. 2012). The raw data was downloaded and
278 assembled using a kmer of 31 and subset size of 1 with Spherical. Contig scores
279 were calculated for each contig produced using the method described in Mende et al,
280 2012.

281

282 **Datasets**

283 We have selected 3 metagenomic samples from different environments to allow for
284 comparison of the methods on datasets with different sequencing depths and
285 complexity, Table 2.

286

287 All of the datasets used were obtained from MG-RAST (Meyer et al. 2008), Table 2,
288 and aimed to provide variation in sampling environments as to provide a robust
289 testing sample for *Spherical*.

290

291 **Table 2:** Information on each dataset

Dataset	Environment	Dataset size (Gbp)	MGRAS project ID	Citation
Chicken caecum dataset	Chicken Caecum	0.06	101	(Qu et al. 2008)
Human oral dataset	Human oral cavity	0.63	128	(Belda-Ferre et al. 2012)
Groundwater dataset	Groundwater from Yucatan Peninsula	29.00	5969	(Moore et al. 2009)

292

293

294 **Methods of assembly**

295 We chose digital normalisation as comparative method to *Spherical* due to its ability
296 to produce a subset of the dataset with uniform coverage without removing the
297 datasets complexity. SLICEMBLER was also considered however as covered earlier
298 metagenomics provides an unknown coverage due to the microbial population itself
299 being unknown and therefore cannot be supplied to SLICEMBLER.

300

301 The optimum kmer size was identified before the comparison assemblies by
302 assembly of each dataset using Velvet with a kmer of 21,31,41 and 51. The raw
303 reads were then aligned to each assembly using Bowtie2. The assembly with the
304 highest alignment rate was then selected and that kmer used for the three methods
305 being compared. All the methods used Velvet as the assembler, removing the

306 assembler as a variable. Velvet was selected as the base assembler due to being
307 open-source, allowing anyone access to it, as well as producing high quality
308 metagenomic assemblies (Shi et al. 2014; Namiki et al. 2012).

309

310 **Khmer - Digital normalization**

311 Digital normalization is based on the use of kmers and is part of the Khmer package.
312 Firstly Khmer breaks each read into kmers, producing a hash-table for the entire
313 dataset present in the dataset. Once counted, it removes reads consisting of
314 redundant kmers, reducing the dataset size whilst keeping sufficient data for an
315 accurate assembly. With each dataset we applied the "normalize-by-median.py"
316 script of Khmer, with a kmer of 20, 4 hash tables of size 32e9 and an ideal median of
317 20.

318

319

320

321 ***Spherical***

322 *Spherical* was run on each dataset with variable sub-set sizes, ranging from 1 to 30.
323 This allowed us to explore the effect of the sub-sampling function of *Spherical*.

324

325 **Basic assembly method**

326 The basic assembly method involved the entire dataset being entered into Velvet
327 with the optimum kmer and the expected coverage is automatically selected by
328 Velvet.

329

330 **Assembly quality**

331 As we are unable to produce contig scores for real metagenomic assemblies we
332 used the following statistics to provide insight in the quality of each assembly;
333 alignment rate, N50 and false base rate. The alignment rate identifies how much of

the raw data is accounted for by the assembly. The higher the alignment rate the more representative the assembly is of the entire dataset. N50 identifies how fragmented the assembly is by identifying average contig length. False bases (bases to which no read aligns) have no basis in the dataset and therefore indicate the rate of misassembled contigs within an assembly.

339

340 **RAM usage**

341 *Spherical* has been designed to reduce RAM usage, in the hope of overcoming the
342 issue of limited computing infrastructure. When comparing methods, maximum RAM
343 used was taken into account. For normalisation, RAM usage was monitored during
344 both the normalisation and assembly stages, the peak was then taken as max RAM
345 usage. The basic assembly methods RAM usage was also monitored to be used as a
346 base line for RAM requirement for assembly of the dataset.

347

348

349 **Taxonomic analysis**

350 Whilst statistics about the assembly allow us to identify how representative the
351 assembly is of the raw data, taxonomic annotation allows us greater insight into the
352 data. Each assembly was taxonomically annotated against the bacterial and archaeal
353 UNIPROT databases using RAPsearch (Zhao et al. 2012; Consortium 2015). The
354 output was then converted into a general feature format (GFF) file and filtered by
355 overlap using MGKIT. HTSeq-count was then used to identify the abundance of
356 phylums within each assembly (Anders et al. 2015).

357

358 **Effect of *Spherical* subset function**

359 To study the effect *Spherical* sub-sampling had on the output, dataset 3 was
360 assembled under 3 different sub-sampling levels. Firstly, the entire file was included
361 (sub-sampling = 1). This allowed us to study what the basic effect of multiple

iterations of assembly was on the output. Next, a quarter of all reads were used (sub-sampling = 4), allowing us to study the effect whilst significantly reducing RAM usage (but still aiming to produce a quality assembly). Finally, only one thirtieth of the reads were used (set-sampling = 30), this was aimed to study the extreme effects of sub-sampling a very small amount of reads and how this would effect the output assembly as well as RAM usage.

Effect of multiple iterations of assembly

The *Spherical* assembly of the Cecum, Oral and Groundwater datasets (sub-sampling = 1) were studied to understand the biodiversity within each iterations of assembly. The ability of multiple iterations to uncover low abundant species was also studied by taxonomically annotating (against bacterial and archaeal UNIPROT database using RAPsearch) each iteration and evaluating their taxonomic profile.

Acknowledgments:

We wish to thank the advice provided by Francesco Rubino to this project. TH was funded through the New Zealand fund for Global Partnerships in Livestock Emissions Research (GPLER). CJC was funded under the Biotechnology and Biological Sciences Research Council (BBSRC) Institute Strategic Programme Grant, Rumen Systems Biology, (BB/E/W/10964A01).

Disclosure declaration:

No conflicts of interest to declare.

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408 **References:**

409 Anders, S., Pyl, P.T. & Huber, W., 2015. Genome analysis HTSeq — a Python

410 framework to work with high-throughput sequencing data. , 31(2), pp.166–169.

411 Belda-Ferre, P. et al., 2012. The oral metagenome in health and disease. *The ISME*

412 *Journal*, 6(1), pp.46–56. Available at:

413 <http://www.nature.com/doi/10.1038/ismej.2011.85>.

414 Bergeron, A. et al., 2007. Divide and Conquer: Enriching Environmental

415 Sequencing Data. , (9).

416 Charuvaka, A. & Rangwala, H., 2011. Evaluation of short read metagenomic

417 assembly. *BMC genomics*, 12 Suppl 2(Suppl 2), p.S8. Available at:

418 <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3194239&tool=pmce>
419 [ntrez&rendertype=abstract](http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3194239&tool=pmce) [Accessed January 21, 2014].

420 Compeau, P.E.C., Pevzner, P. a & Tesler, G., 2011. How to apply de Bruijn graphs to
421 genome assembly. *Nature biotechnology*, 29(11), pp.987–91. Available at:
422 <http://www.ncbi.nlm.nih.gov/pubmed/22068540> [Accessed July 9, 2014].

423 Consortium, T.U., 2015. UniProt[®]: a hub for protein information. , 43(October 2014),
424 pp.204–212.

425 Dro, J. & Mchardy, A.C., 2012. axonomic binning of metagenome samples generated
426 by next-generation sequencing technologies. , 13(6), pp.646–655.

427 Hess, M. et al., 2011. Metagenomic discovery of biomass-degrading genes and
428 genomes from cow rumen. *Science (New York, N. Y.)*, 331(6016), pp.463–7.

429 Krohn-Molt, I. et al., 2013. Metagenome survey of a multispecies and alga-
430 associated biofilm revealed key elements of bacterial-algal interactions in
431 photobioreactors. *Applied and environmental microbiology*, 79(20), pp.6196–
432 206. Available at: <http://www.ncbi.nlm.nih.gov/pubmed/23913425> [Accessed
433 February 14, 2014].

434 Lai, B. et al., 2012. A de novo metagenomic assembly program for shotgun DNA
435 reads. *Bioinformatics (Oxford, England)*, 28(11), pp.1455–62. Available at:
436 <http://www.ncbi.nlm.nih.gov/pubmed/22495746> [Accessed January 23, 2014].

437 Langmead, B. & Salzberg, S.L., 2012. Fast gapped-read alignment with Bowtie 2.
438 *Nature Methods*, 9(4), pp.357–359.

439 van der Lelie, D. et al., 2012. The metagenome of an anaerobic microbial community
440 decomposing poplar wood chips. *PloS one*, 7(5), p.e36740. Available at:
441 <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3357426&tool=pmce>
442 [ntrez&rendertype=abstract](http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3357426&tool=pmce) [Accessed February 11, 2014].

443 Mende, D.R. et al., 2012. Assessment of metagenomic assembly using simulated
444 next generation sequencing data. *PloS one*, 7(2), p.e31386. Available at:
445 <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3285633&tool=pmce>

ntrez&rendertype=abstract [Accessed January 28, 2014].

Meyer, F. et al., 2008. The metagenomics RAST server – a public resource for the automatic phylogenetic and functional analysis of metagenomes. , 8, pp.1–8.

Mirebrahim, H., Close, T.J. & Lonardi, S., 2015. De novo meta-assembly of ultra-deep sequencing data. *Bioinformatics*, 31(12), pp.i9–i16. Available at: <http://bioinformatics.oxfordjournals.org/content/31/12/i9.full?etoc>.

Modi, S.R. et al., 2013. Antibiotic treatment expands the resistance reservoir and ecological network of the phage metagenome. *Nature*, 499(7457), pp.219–22. Available at: <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3710538&tool=pmc> ntrez&rendertype=abstract [Accessed January 20, 2014].

Mohammed, M.H. et al., 2011. SPHINX--an algorithm for taxonomic binning of metagenomic sequences. *Bioinformatics (Oxford, England)*, 27(1), pp.22–30. Available at: <http://bioinformatics.oxfordjournals.org/cgi/content/long/27/1/22> [Accessed February 19, 2014].

Nagarajan, N. & Pop, M., 2013. Sequence assembly demystified. *Nature reviews. Genetics*, 14(3), pp.157–67. Available at: <http://www.ncbi.nlm.nih.gov/pubmed/23358380> [Accessed January 21, 2014].

Namiki, T. et al., 2012. MetaVelvet: an extension of Velvet assembler to de novo metagenome assembly from short sequence reads. *Nucleic acids research*, 40(20), p.e155. Available at: <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3488206&tool=pmc> ntrez&rendertype=abstract [Accessed May 26, 2014].

Peng, Y. et al., 2012. IDBA-UD: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics (Oxford, England)*, 28(11), pp.1420–8. Available at: <http://www.ncbi.nlm.nih.gov/pubmed/22495754> [Accessed February 4, 2014].

Qu, a et al., 2008. Comparative Metagenomics Reveals Host Specific Metavirulomes

474 and Horizontal Gene Transfer Elements in the Chicken Cecum Microbiome.
 475 *Plos One*, 3(8), p.19. Available at: <Go to ISI>://WOS:000264412600031.
 476 Shi, W. et al., 2014. Methane yield phenotypes linked to differential gene expression
 477 in the sheep rumen microbiome. *Genome research*. Available at:
 478 <http://www.ncbi.nlm.nih.gov/pubmed/24907284> [Accessed June 10, 2014].
 479 Venter, J.C. et al., 2004. Environmental genome shotgun sequencing of the
 480 Sargasso Sea. *Science (New York, N.Y.)*, 304(5667), pp.66–74.
 481 Wang, Y. et al., 2012. MetaCluster 5.0: a two-round binning approach for
 482 metagenomic data for low-abundance species in a noisy sample. *Bioinformatics*
 483 (Oxford, England), 28(18), pp.i356–i362. Available at:
 484 <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3436824&tool=pmc>
 485 [ntrez&rendertype=abstract](http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3436824&tool=pmc&rendertype=abstract) [Accessed February 14, 2014].
 486 Zhao, Y., Tang, H. & Ye, Y., 2012. RAPSearch2: a fast and memory-efficient protein
 487 similarity search tool for next-generation sequencing data. *Bioinformatics*
 488 (Oxford, England), 28(1), pp.125–6. Available at:
 489 <http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3244761&tool=pmc>
 490 [ntrez&rendertype=abstract](http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3244761&tool=pmc&rendertype=abstract) [Accessed January 25, 2015].
 491
 492
 493