Using the concept of informative genomic segment to investigate microbial diversity of metagenomics sample

Qingpeng Zhang, C. Titus Brown

Department of Computer Science,Michigan State University

v.20141001

Abstract

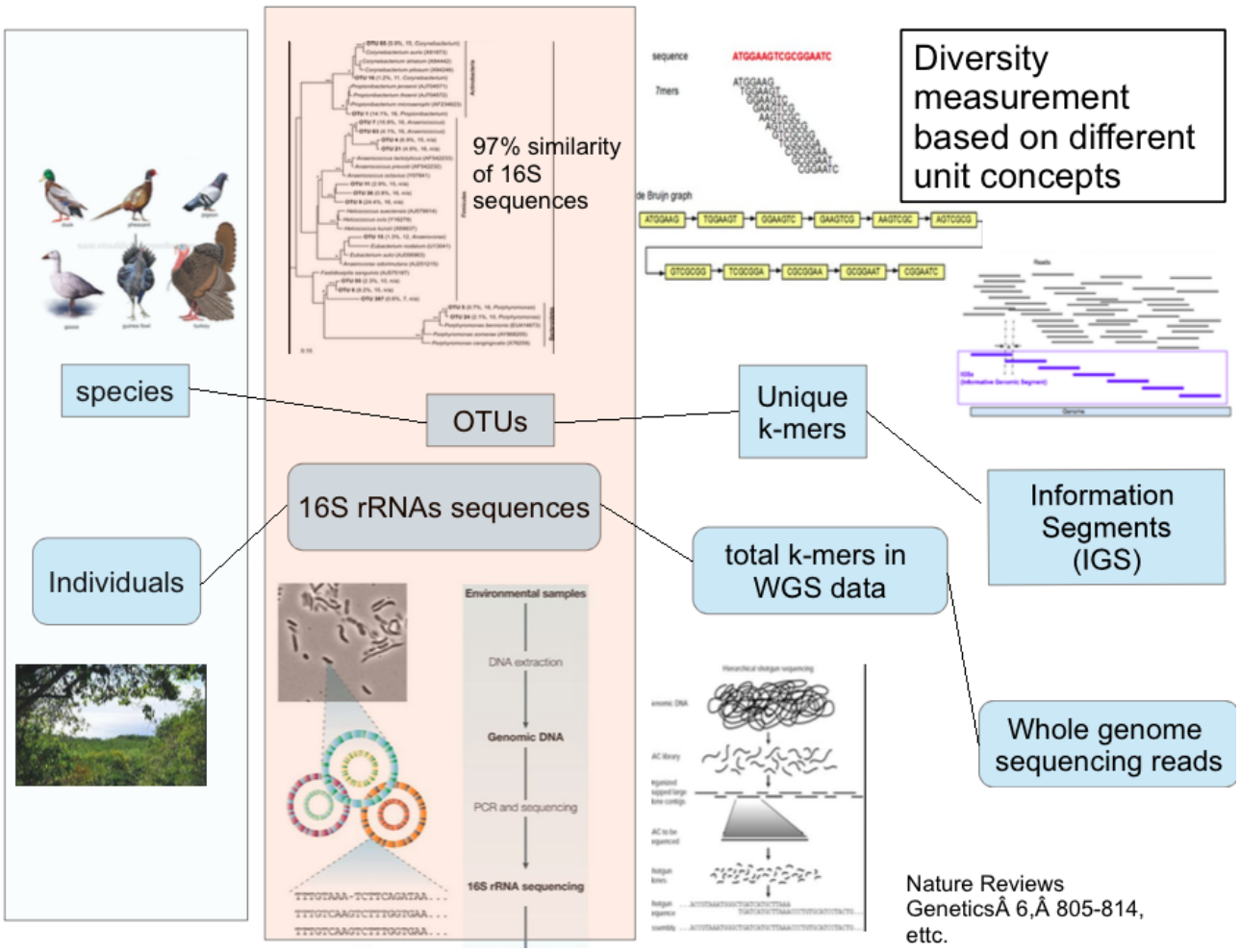
In almost all the metagenomics projects, diversity analysis plays an important role to supply information about the richness of species, the species abundance distribution in a sample or the similarity and difference between different samples, all of which are crucial to draw insightful and reliable conclusion. Traditionally especially for amplicon metagenomics data set, OTUs(Operational Taxonomic Units) based on 16S rRNA genes are used as the cornerstone for diversity analysis. Here we propose a novel concept - IGS (informative genomic segment) and use IGS as a replacement of OTUs to be the cornerstone for diversity analysis of whole shotgun metagenomics data sets. IGSs represent the unique information in a metagenomics data set and the abundance of IGSs in different samples can be retrieved by the reads coverage through an efficient k-mer counting method. This samples-by-IGS abundance data matrix is a promising replacement of samples-by-OTU data matrix used in 16S rRNA based analysis and all existing statistical methods can be borrowed to work on the samples-by-IGS data matrix to investigate the diversity. We applied the IGS-based method to several simulated data sets and a real data set - Global Ocean Sampling Expedition (GOS) to do beta-diversity analysis and the samples were clustered more accurately than existing alignment-based method. We also tried this novel method to Great Prairie Soil Metagenome Grand Challenge data sets. Furthermore we will show some preliminary results using the IGS-based method for alpha-diversity analysis. Since this method is totally binning-free, assembly-free, annotation-free, reference-free, it is specifically promising to deal with the highly diverse samples, while we are facing large amount of “dark matters” in it, like soil.

Background

In traditional ecology, the concept of species is used to investigate diversity. In microbial ecology, the concept of OTU is used to investigate microbial diversity. OTU is mostly used for 16S data sets. And binning reads into OTU is typically required for OTU-based diversity analysis. Here we propose a new concept - IGS to replace the concept of OTU in 16S based diversity analysis and the concept of species in traditionaly ecology. Refer to the figure below. Also see more details about the background in Qingpeng's conprensive example proposal. https://github.com/qingpeng/2013-diversity/blob/master/Docs/proposal.pdf

In [49]:
```
from IPython.display import Image
Image(filename="../figure/species_OTU_IGS.png",width=800)
```

Out[49]:



The concept of IGS(informational genomic segment)

In classic ecology dealing with macroorganisms, diversity measurement is based on the concept of ”species”. For 16S rRNA amplicon metagenomics data set, it is based on the concept of ”OTUs”. When the concept of OTUs does not work for large shotgun metagenomics data set, in the beginning we proposed that the concept of k-mers(a DNA segment with the leng of k) can be used to measure diversity. K-mers can be considered as the atom of information in DNA sequences. One of the composition-based approaches to binning is to use the k-mer as the signature. Suppose the sizes of microbial genomes are similar and the difference between genomic content of microbial genomes is similar, the number of distinct k-mers in the sequence data set is related to the number of species in a sample. However, because of sequencing error, which is unavoidable due to the limit of sequencing technology, this k-mer based analysis doe not work well. One sequencing error on a read will generate at most k erroneous k-mers. In metagenomics data set with low coverage, most of the distinct observed k-mers are from sequencing errors.

Next we turned our gaze to the upper level - reads. A novel method termed as ”digital normalization” was developed to remove abundant reads before assembly. However it also supplies a novel way to distil information from reads by decreasing the bad influence of sequencing errors so that we can use those informative reads to measure the microbial diversity. We term those informative reads as IGS(informative genomic segment), which can be considered as a segment of DNA on a microbial genome. Those IGSs should be different enough to represent the abstract information a genome contains. Suppose microbial genomes contain similar number of those IGSs, as they contain similar number of distinct k-mers, the number of IGSs will be related to the species richness in a sample, and the abundance distribution of IGSs will be related to species evenness in a sample. Many classic diversity estimation methods based on OTUs level described in sections above can be borrowed to estimate the diversity of IGSs and the diversity of actual species subsequently.

IGS may be a good concept in whole genome shotgun metagenomic diversity analysis, especially while facing large ammount of "dark matters", unkonwn species. We don't care about species, we only care about how much information there is in the sample.

For alpha diversity, we can generate a list of IGSs and the respective abundance in a sample. Then existing estimators like Chao's can be used to estimate total number of IGSs in the sample. Rarefaction curve based on number of IGSs can also be genereated.

For beta diversity, here we will generate a samples-by-IGS data matrix, as a replacement of samples-by-OTU data matrix in 16s based analysis and samples-by-species data matrix in traditional ecology.

```
Species_ID FieldA FieldB FieldC FieldD
species1   3       2       1       0
species2   3       2       1       0
species3   5       1       2       1
species4   5       1       2       1
species5   3       2       0       1
```

```
OTU_ID Sample1_ID Sample2_ID Sample3_ID
OTU1   3           4           2
OTU2   2           5           0
OTU3   3           1           4
...
```

From that samples-by-IGS data matrix, we can use existing methods to calculate similarity/disimilarity/distance between samples and do ordination. QIIME and Mother can do this kind of jobs pretty well.

With the samples-by-IGS data matrix, it is also possible to calculate similarity between IGSs and do ordination, which is a potential approach to classify IGS(reads).

Potential applications

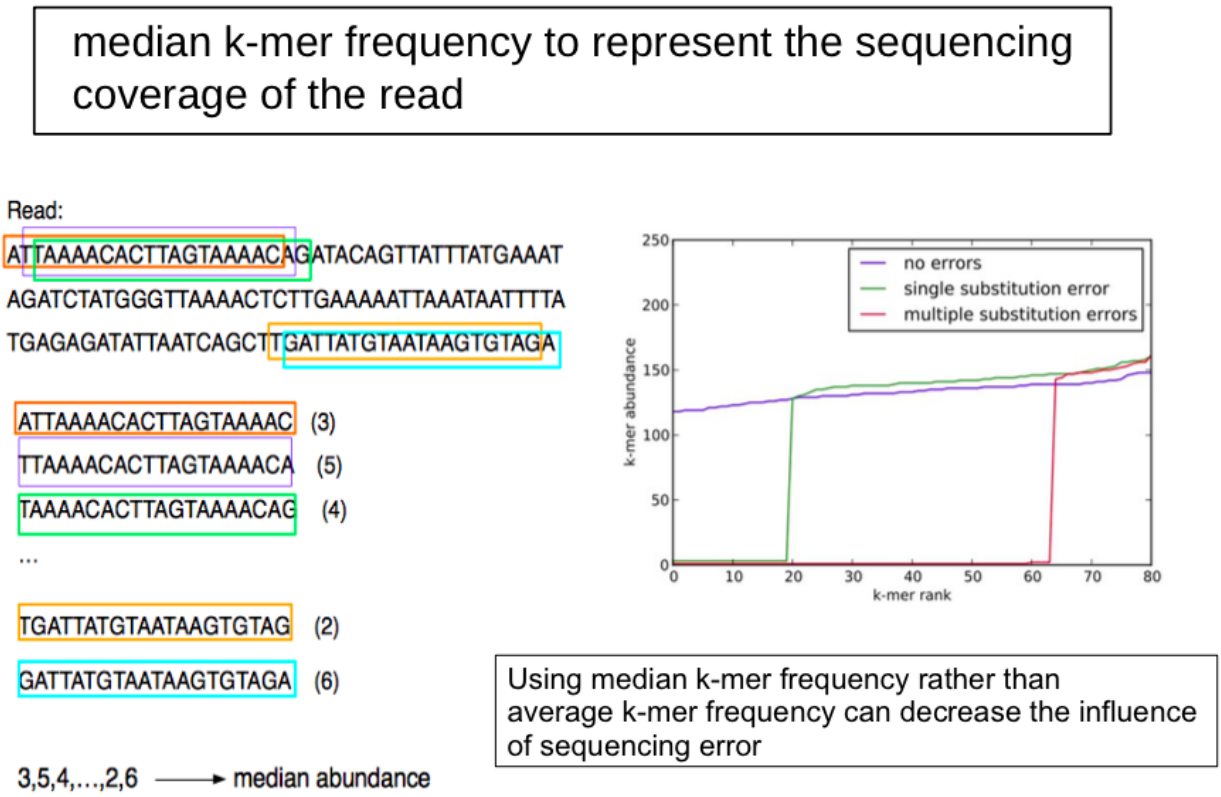
1. alpha-diversity analysis (1 sample)
    - richness/evenness
    - rarefaction curve
    - sequencing depth evaluation
    - genome size estimation
    - better choosing diginorm parameters(size of hashtables, etc.)
2. beta-diversity (multiple samples)
    - sample by sample comparison, clustering, ordination after getting segment-count table
3. other potential applications:
    - reads binning/classification (after clustering)(if number of samples is small, may not be effective)
    - extract IGS(reads) according different filters (shared by all samples, or some specific samples,)
    - co assembly (by extracting the reads with total coverage across samples > 10, for example)

How did we come up with the concept of IGS?

median k-mer frequency of a read can represent sequencing depth

In [50]:
```
Image(filename="../figure/median_kmer_frequency_represent_coverage.png",width=600)
```

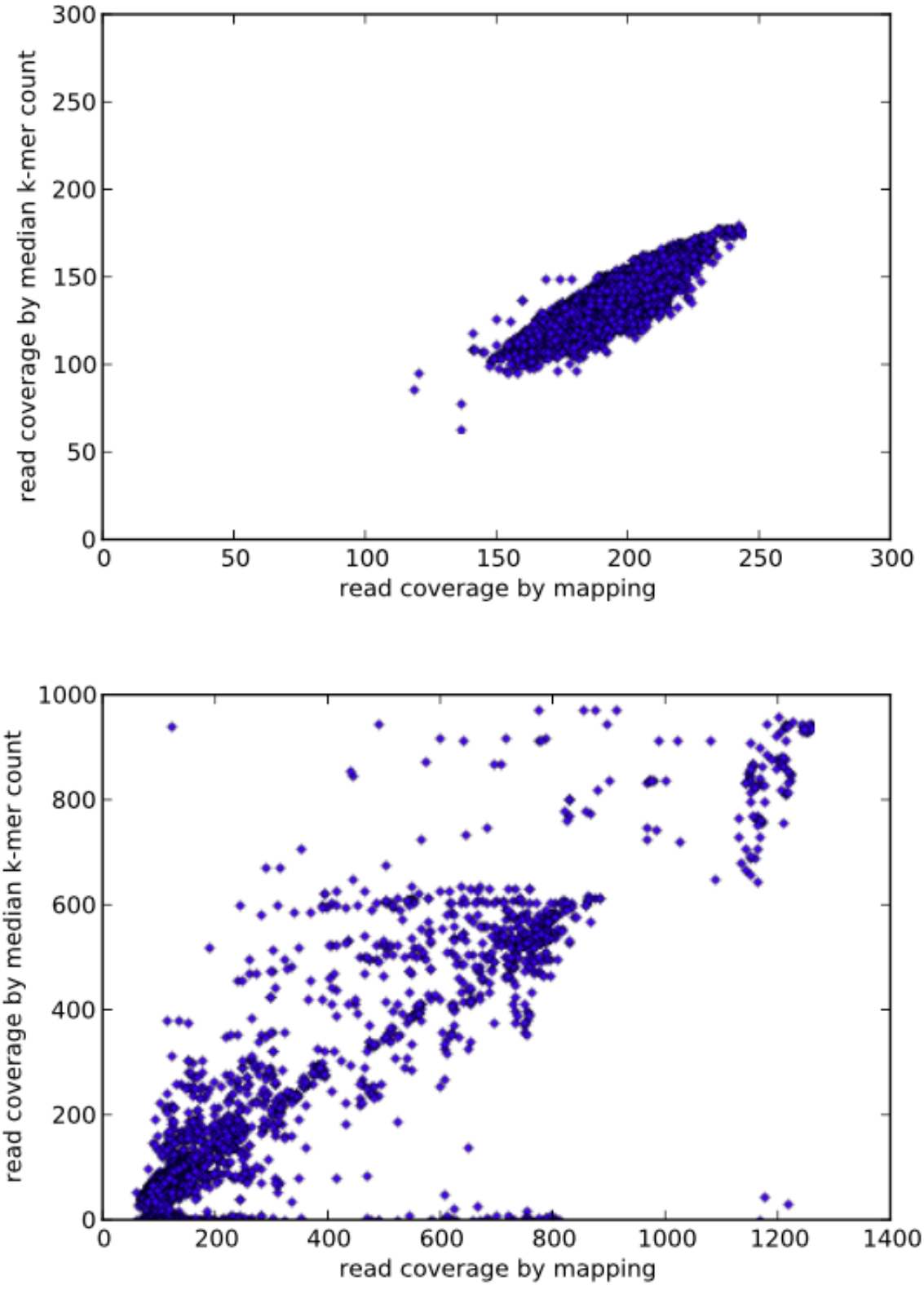
Out[50]:



Side note: Using median k-mer frequency can decrease the influence of sequencing error, but can not elimite the influence of errors. This can cause some problems in the following analysis, which will be discussed in details.

In [51]:
```
Image(filename="../figure/correlation_mapping_median_coverage.png")
```

Out[51]:



Mapping and k-mer coverage measures correlate for simulated genome data and a real E. coli data set (5m reads). Simulated data r2 = 0.79; E. coli r2 = 0.80. (Brown et al., 2012)

Using simulated and real genomic data sets, with the help of khmer to get median k- mer abundance, we find that median k-mer abundance correlates well with mapping-based coverage.

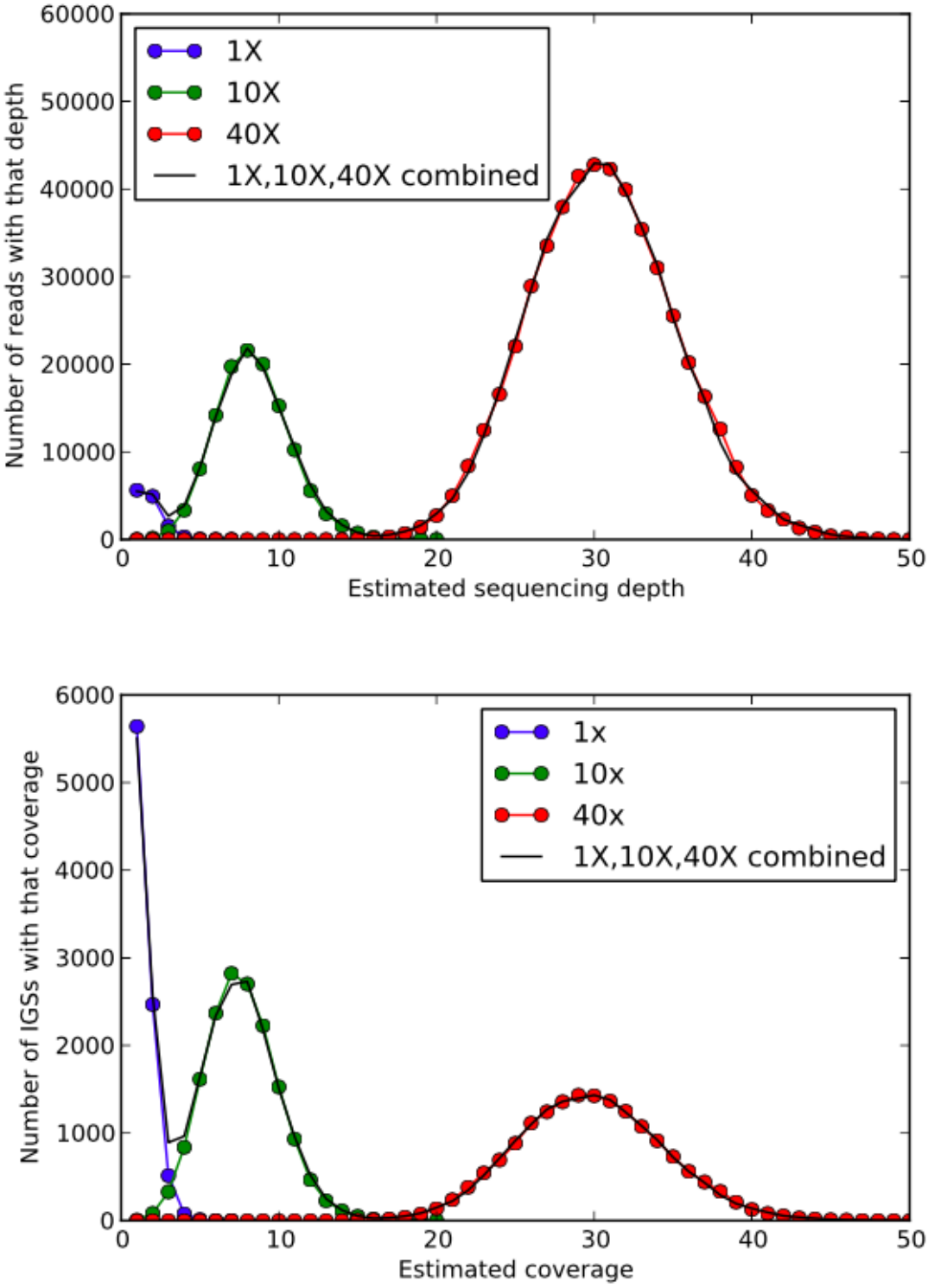
Side note: We can also see that the coverage by median k-mer count is generally lower than the read coverage by mapping. So estimation of sequencing coverage by median k-mer count is underestimated. This can cause some problems in later analysis, which will be discussed later.

IGS(informative genomic segment) can represent the novel information of a genome

From discussion above, median k-mer abundance can represent sequencing depth of a read. For a sequencing reads data set with multiple species, the sequencing depth of a read is related to the abundance of species where the read originates.

In [52]:
```
Image(filename="../figure/from_reads_to_IGS.png")
```

Out[52]:



The figure above shows the abundance distribution with different sequencing depth of reads from 4 simulated sequencing data sets - 3 sequencing data sets generated with different sequencing coverage(1x, 10x, 40x) from 3 simulated random genomes respectively and 1 combined data set with all the previously mentioned data sets. No error is introduced in these simulated data sets. Obviously the reads from the three data sets can be separated by estimated sequencing depth. The combined data set can be considered as a sequencing data set with three species with different abundance.

Each point on the curve shows that there are Y reads with a sequencing depth of X. In other word, for each of those Y reads, there are X-1 other reads that cover the same DNA segment in a genome that single read originates. So we can estimate that there are Y/X distinct DNA segments with reads coverage as X. We term these distinct DNA segments in species genome as IGS(informative genomic segment). We can transform the figure in upper position to show the number of IGSs and their respective reads coverage, as shown in figure in lower position. We sum up the numbers of IGSs with different reads coverage for each data set and get the result as shown in below. The sum numbers of IGSs here essentially are the areas below each curve in the figure.

In [53]:
```
Image(filename="../figure/number_of_IGS_table.png")
```

Out[53]:

Table 1: **Total number of IGSs in different simulated reads data sets.**

Data set	total number of IGSs
1X depth	8714
10X depth	16321
40X depth	16794
1X,10X,40X combined	41742

Even though the datasets have different sequencing depth like 10X and 40X, they have similar numbers of IGSs. Dataset with 1X sequencing depth has fewer IGSs because the depth is not enough to cover all the content of the genome(63.2%) Essentially it is the maximum number of segments with length L on a genome out of which no two segments share any single k-mer. See Figure below. Assume the species genome is totally random, which is the case in the simulated data set, the number of IGSs(N) in a species genome is related to the size of genome(G), read length(L) and k size(k), which can be denoted as

N =G/(L-k+1)

For the simulated genome with size of 1M bps, read length as 80bps, k-mer size as 22bps,expected number of IGSs is

1000000/(80 - 22 + 1) = 16949,

pretty close to observed value.

In [54]:
```
Image(filename="../figure/IGSs_figure.jpg",width=800)
```

Out[54]:

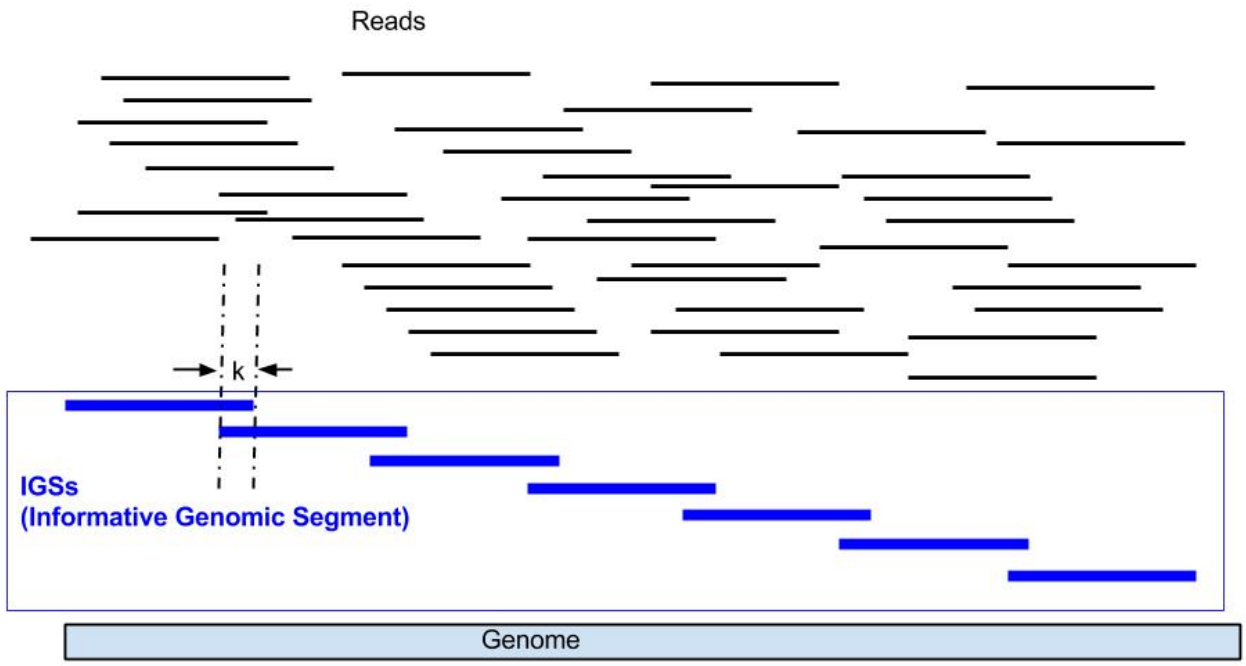


Figure: IGSs are the segments with specific length on genome that do not share any k-mers with each other.

Using IGS to do alpha-diversity

See expanded results and discussion: http://nbviewer.ipython.org/github/qingpeng/2013-diversity/blob/master/notebook/2013-diversity-summary_alpha_diversity.ipynb (http://nbviewer.ipython.org/github/qingpeng/2013-diversity/blob/master/notebook/2013-diversity-summary_alpha_diversity.ipynb)

Basically the abundance distribution of IGSs with different coverage in a sample data set is acquired using the method shown above, like:

```
3 23
4 24
5 25
6 25
...
```

Here 23 IGSs with coverage as 3, this number is calculated from dividing the total number of reads with coverage as 3, which is 69, by the coverage 3: 69/3. Similarly there are 96/4 = 24 IGSs with coverage as 4.

If we draw an analogy between IGSs and OTUs, this is like there are 23 different OTUs with 3 reads mapped to, and 24 different OTUs with 4 reads mapped to.

Then list all the different IGSs and the corrensponding count,and we can get a long list with each IGS and the corresponding coverage.The coverage of an IGS can be considered as the abundance of such IGS in a sample. The list looks like:

```
#IGS_ID sample_ID
1     3
2     3
3     3
...
23    3
24    4
25    4
...
47    4
48    5
...
...
```

This list is the couterpart of an OTU table in OTU based diversity analysis.

With such table at hand, numerous existing statistical methods and software packages can be used to investigate the alpha diversity.

In the experiments shown below, QIIME package was used.

Simulated data sets

Simulated sequencing reads of e.coli

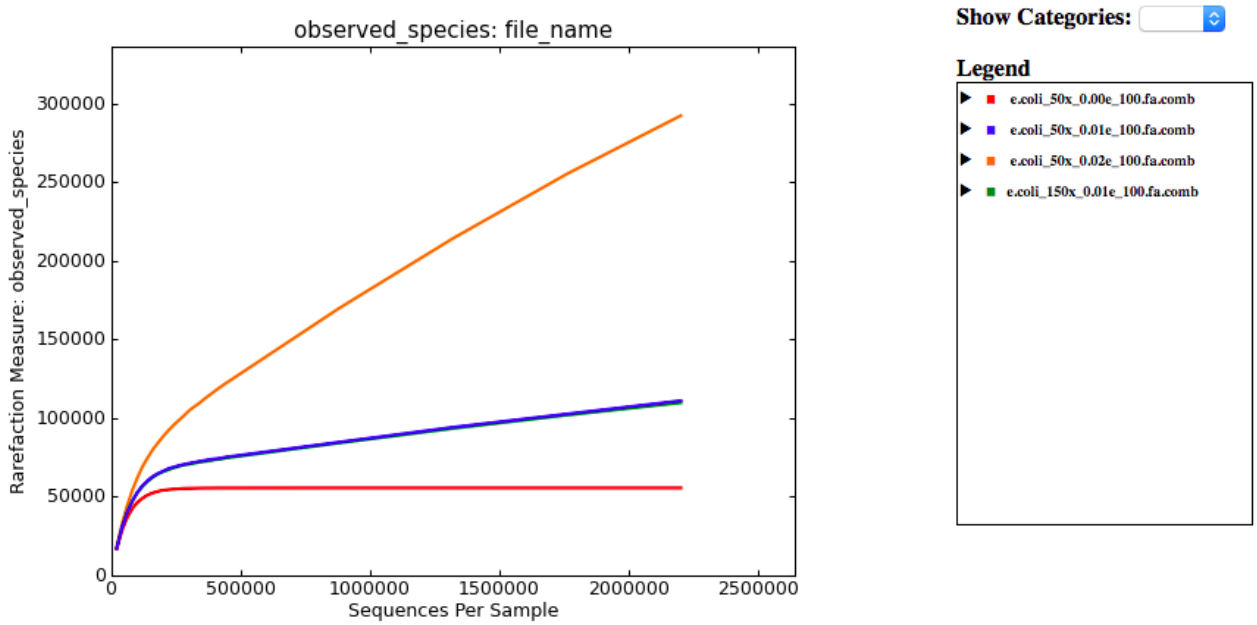
Here we simulated 4 sequencing reads data sets with read length as 100bp of e.coli with different sequencing depth(50x and 150x) and different sequencing error rate(1%,2% and 0%).

The details about the 4 data sets can be shown as the "mapping file" used in QIIME pipeline:

```
[qingpeng@dev-intel14 Full_Curve]$ more e.coli_Map.txt
#SampleID   BarcodeSequence LinkerPrimerSequence    Description coverage
error_rate  file_name
sampleA A   A   e.coli_150x_0.01e_100   150x    0.01    e.coli_150x_0.01e_10
0.fa.comb
sampleB A   A   e.coli_50x_0.00e_100    50x 0.00    e.coli_50x_0.00e_100.fa.
comb
sampleC A   A   e.coli_50x_0.01e_100    50x 0.01    e.coli_50x_0.01e_100.fa.
comb
sampleD A   A   e.coli_50x_0.02e_100    50x 0.02    e.coli_50x_0.02e_100.fa.
comb
```

In [55]:
```
Image(filename="../figure/e.coli_alpha_observed_full_curve.png",width=800
)
```

Out[55]:



Rarefaction curve of observed IGS (here, "species" are "IGSs")

We can see for reads data set without sequencing error, the IGSs become saturated at about 300000 reads.

size of e.coli genome: 4639675bp

The coverage when IGSs become saturated is:

```
300000x100/4639675 = 6
```

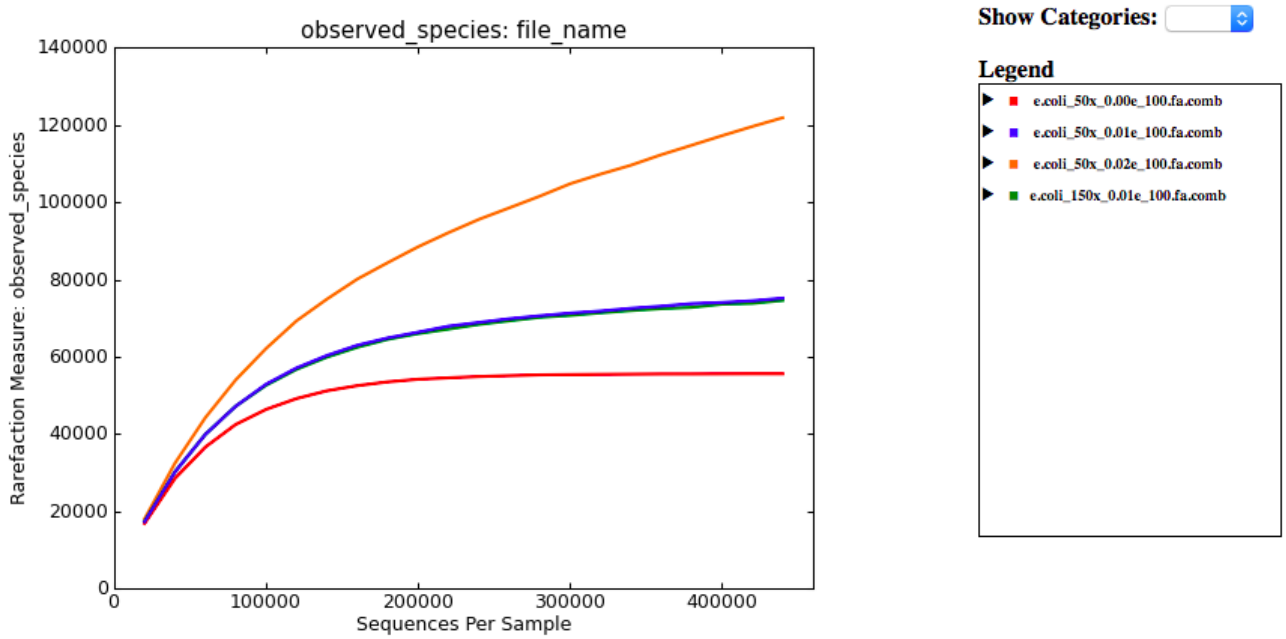
The expected number of IGSs in e.coli genome is:

```
4639675/(100-20+1) = 57279
```

Here the saturation number of IGSs is about 55500, which is close to expected number. The cause of the difference still needs further investigation. The choice of k size may be a factor.

In [56]:
```
Image(filename="../figure/e.coli_alpha_observed_beginning_curve.png",width=800)
```

Out[56]:



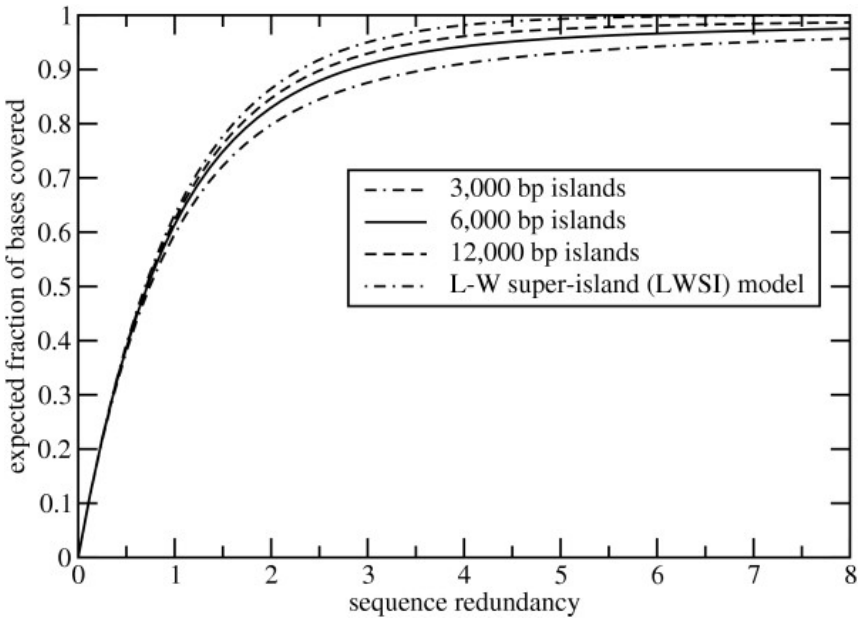
The curve for low coverage. (every 100000 is about 2X depth)

Here we can see the saturation of IGSs for the curve without error (red color) is exactly like the typical Lander & Waterman curve about the sequencing depth. (as below)

This is another proof that IGS can represent the information in a genome well.

In [57]:
```
Image(filename="../figure/lander_saturation.png",width=400)
```

Out[57]:



The haunting sequencing error

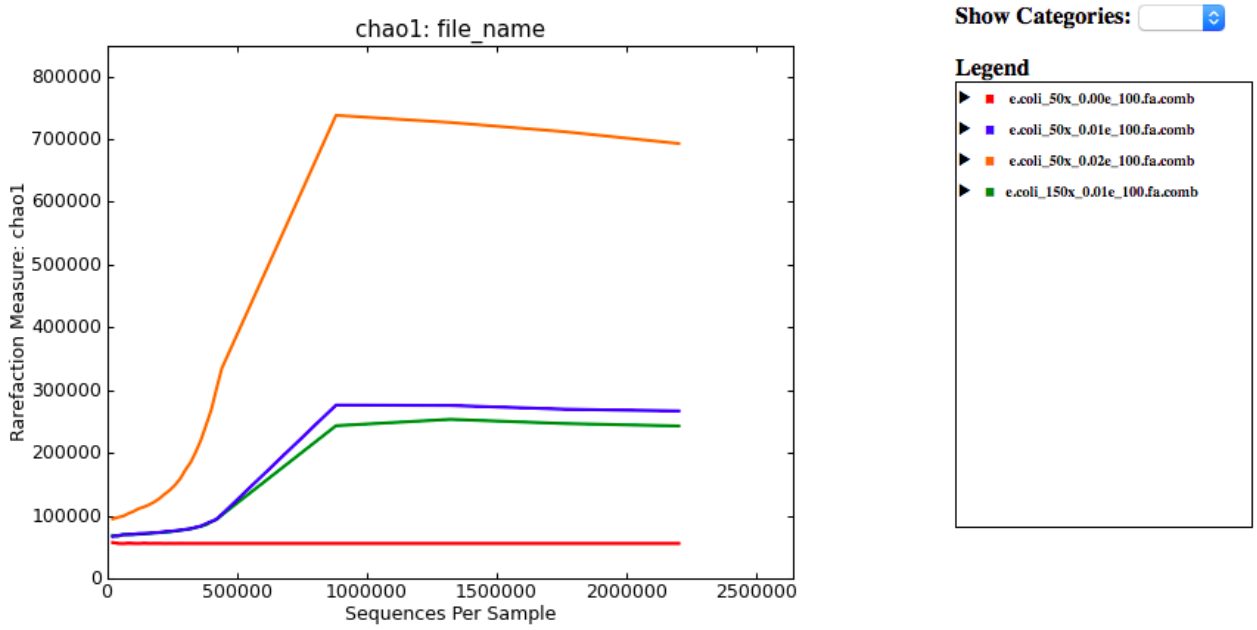
From the IGSs saturation curve of data set with sequencing errors, we can see the sequencing error still causes trouble.

The number of IGSs is continuing rising with a sequencing error rate as 1%. And it is pretty sensitive to error rate. The saturation curve with error rate as 2% increases much faster than 1%.

Chao1 estimator

In [58]:
```
Image(filename="../figure/e.coli_alpha_chao1_full_curve.png",width=800)
```

Out[58]:



This is intersting. After consuming about 8000000 reads, the estimated total number of IGSs becomes stable by Chao1 estimator.

I will try more estimators to check the curve.

Next we tried to do some error correction before applying this method to remove the influence of errors.

Try error correction beforehand

Here we used an in-house error correction tool to process the simulated data set.

The curve of corrected data with original 1% error rate is perfect now. It looks like the influence of sequencing error is dramatically eliminated after such error correction method. After error correction, the remaining errors will no longer pose a threat to the median kmer counting.

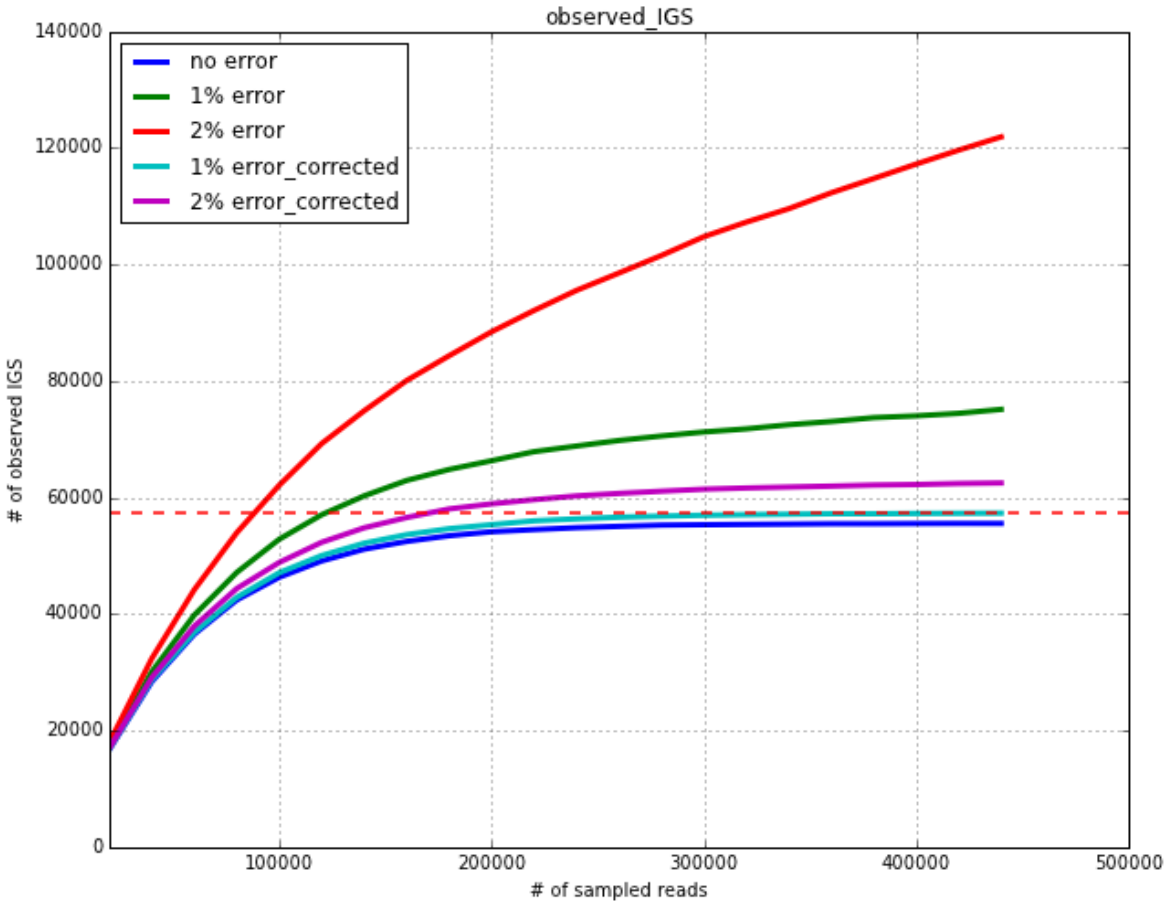
For 2% error rate, after error correction, the curve is better than that of 1% before correction.

The Chao1 estimators works so good after error correction, too good to be true. I will look into this.

For more details, check: http://nbviewer.ipython.org/github/qingpeng/2013-diversity/blob/master/notebook/2013-diversity_saturation_curve.ipynb

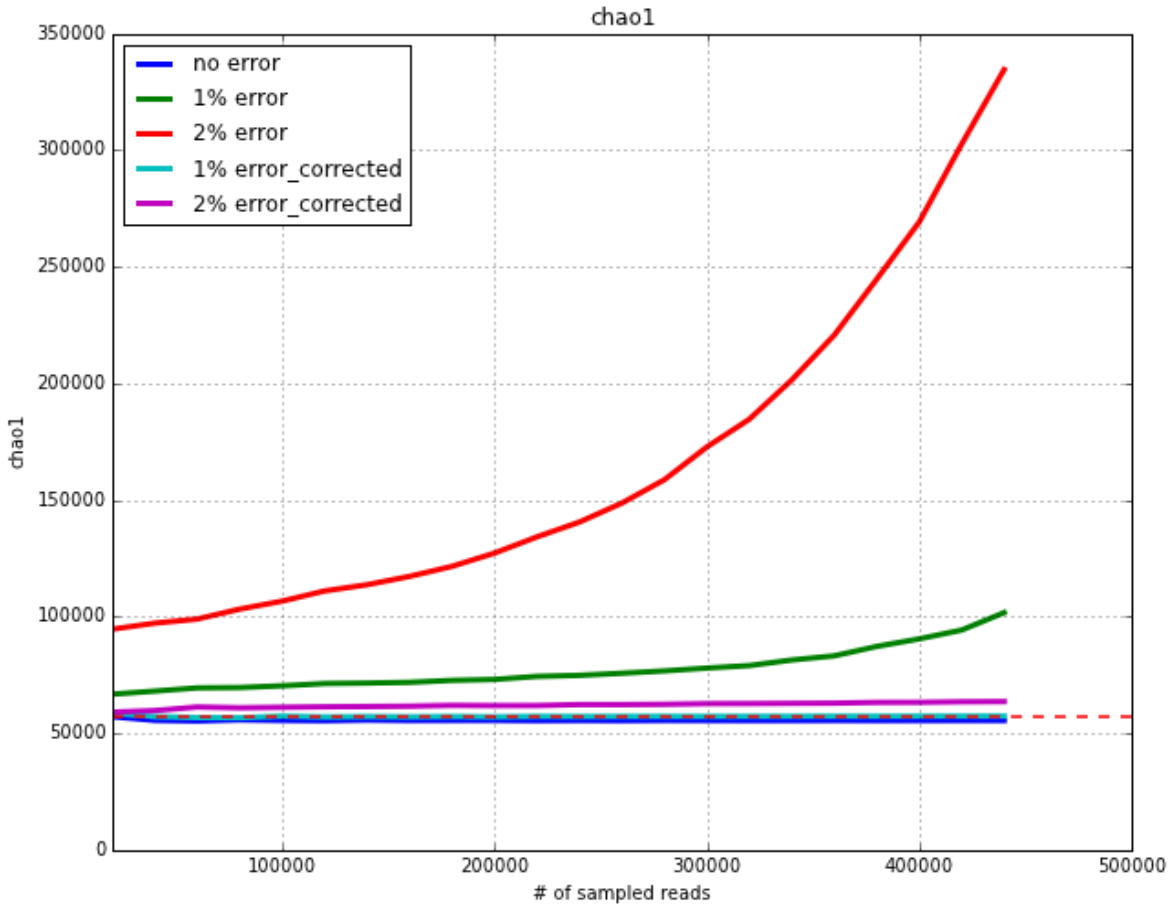
In [59]:
```
Image(filename="../figure/alphs_error_correct.png",width=800)
```

Out[59]:



In [60]:
```
Image(filename="../figure/alphs_error_correct_chao1.png",width=800)
```

Out[60]:



Other thoughts about the solution to solve the error problem

1. estimating error rate
    - We are working on another method to estimate error rate of a sequencing data set
2. fitting the curve with error rate
    - with estimated error rate, fitting the curve. (Is there any existing method to do this kind of fitting?)
3. Nonpareil
    - investigate if we can borrow some statistics from Nonpareil paper
4. we can observe that there are still many IGSs with coverage as 1 in a high sequencing depth data set. Obviously in this case, most of them are from sequencing error. As discussed in previous section, if there are muliple error in a read, the median k-mer count will not represent the sequencing depth well. So we can try:
    - Don't use the IGSs with coverage as 1, since they are not reliable. However, for low coverage dataset, this will discard many true 1-coverage IGSs. If 1-coverage IGSs are not reliable, what about 2-coverage IGSs? or higher coverage? Can we only use the IGSs with higher coverage to do the estimation? Try other estimators, Chao1 does not work. Try some others.
    - Can we use other k-mer count to represent the sequencing depth? like highest count? uppper 25% count? So it can tolerate more errors in one single read? (For now, we don't plan to use this approache. Since this will cause some other problems.)

Brief Summary of IGS-based alpha diversity analysis

- The sequencing error causes difficults to do quantitive analysis. If the quantitive analysis is more accurate, this method can supply a promising approach to do sequencing depth estimation or genome size estimation. Doing error correction beforehand is a promising approach.
- Although the quantitive analysis is not accurate, we can still have some idea about the richness or evenness of a metagenomic sample. With further improvement, like more sorphisticated statistical model, the influence of sequencing errors can be predicted. so the estimation can be more accurate.
- It is difficult to estimate the exact richness of a sample. But we can compare the richness of different samples. We can answer the question like which sample are more diverse, or has more species. (Testing with simulated data sets is required...)
- More simulated data set, like metagenomics data set. Here the test data sets are just for one species.

Other simulated datasets,(to be done)

Real data sets

MetaHIT

GOS

GPGC (later)

Results to be added...

Using IGS to do beta-diversity

As in alpha diversity analysis, OTU table is also a cornestone for beta diversity analysis. As long as we get a reliable OTU table, there are existing pipelines to do the beta diversity analysis.

A typical OTU table across different samples is like this, which is also called samples-by-OTU data matrix.

```
OTU_ID Sample1_ID Sample2_ID Sample3_ID
OTU1   3          4          2
OTU2   2          5          0
OTU3   3          1          4
...
```

Like a OTU table, we hope to have the IGS table for the IGSs:

```
IGS_ID SampleA SampleB SampleC SampleD
IGS1   5       1       2       1
IGS2   5       1       2       1
...
```

So now the problem is how we can generate a sample-by-IGS data matrix as the couterpart of samples-by-OTU data matrix so many of the existing tools/methods used for OTU-based diversity can be borrowed for this kind of IGS-based analysis, just as what is shown above for alpha diversity analysis.

Firstly, as how we get the coverage of a read from a sample dataset in this sample dataset, we can get the coverage of a read from a sample A dataset in another sample B dataset. We can still use the median k-mer count to represent the coverage. The basic idea is the same.

Because a read must derive from a segment in the genome of some species in a sample, if a read **R** from sample A with a coverage **C_A** in sample A has a coverage as **C_B** in sample B, that means that segment of genome in sample A from which read **R** derive also exists in sample B. That genomic segment has a coverage as **C_A** in sample A and has a coverage as **C_B** in sample B. Roughly there should be about **C_A** reads (read **R** should be one of them) in sampleA covering that genomic segment and **C_B** reads in sampleB covering that genomic segment. Meanwhile, the **C_A** reads in sampleA should all have a coverage as **C_B** in sampleB, just like read **R** as one of them. Similary, the **C_B** reads in sampleB should all have a coverage as **C_A** in sampleA.

Ok, now let's make an example.

Suppose there are 6 reads in sample A, all have a coverage as 3 in sampleA, and have a coverage as 2 in sampleB.

According to the discussion about IGS in previous section, the 6 reads cover 2 IGSs with a coverage as 3 for each IGSs. There should be 4 reads in sampleB covering the exact same 2 IGSs, with a coverage as 2 in sampleB.

So now we have 2 distinct IGSs with redundancy as 3 and 2 in the two samples respectively.

Note: small number is used in the analysis above as example, but it should be emphasized that the analysis is based on large number statistically.

Let's expand this example from 2 samples to 4 samples(A,B,C,D), as shown in figure above.

Let's say we find 10 reads in sampleA, with coverage as 5-1-2-1 in samples A-B-C-D respectively. (We call "5-1-2-1" "coverage spectrum" across samples.) So there should be **about** 2 reads in sampleB, 4 reads in sampleC, 2 reads in sampleD, all of which have a "coverage spectrum" as "5-1-2-1". Basically these 18 reads altogether cover 2 distinct IGSs, which apparently exist in all the 4 samples. The 2 distinct IGSs has a redundancy as 5,1,2,1 in the 4 samples respectively.

If we draw an analogy between IGSs and OTUs, this is like there are 2 OTUs, both with 5,1,2,1 reads mapped to in sample A,B,C,D respectively.

Like a OTU table, here we can have the IGS table for the two IGSs:

```
IGS_ID SampleA SampleB SampleC SampleD
IGS1   5       1       2       1
IGS2   5       1       2       1
```

In [61]:
```
Image(filename="../figure/IGS_compare.jpg", width=800)
```

Out[61]:

In the figure above: - black: informational genome - blue: reads, with the coverage spectrum across samples - green: IGSs - red: potential genome of seperate species

We just discussed the situation of IGS_3 and IGS_4 above.

For those reads shown in the figure, finally we can get a IGS table like:

IGS_ID	SampleA	SampleB	SampleC	SampleD
IGS1	3	2	1	0
IGS2	3	2	1	0
IGS3	5	1	2	1
IGS4	5	1	2	1
IGS5	3	2	0	1

This is just like the couterpart of OTU table for 16S based analysis:

OTU_ID	SampleA	SampleB	SampleC	SampleD
OTU1	3	2	1	0
OTU2	3	2	1	0
OTU3	5	1	2	1
OTU4	5	1	2	1
OTU5	3	2	0	1

Or, species table for traditional ecology:

This is just like the couterpart of OTU table for 16S based analysis:

Species_ID	FieldA	FieldB	FieldC	FieldD
species1	3	2	1	0
species2	3	2	1	0
species3	5	1	2	1
species4	5	1	2	1
species5	3	2	0	1

Anyway, as long as we have a table/matrix like this, we can use existing statistical tool / software package to do beta diversity analysis - including clustering and ordination.

Simulated data sets

We built 3 series of synthetic data sets: 10 species

SampleA:

- species IDs: 1,2,3,4,5,6,7,8,9,10
- relative abundance: 20:18:16:4:3:2:2:2:2:2

SampleB:

- species IDs: 1,2,3,14,15,16,17,18,19,20
- relative abundance: 20:18:16:4:3:2:2:2:2:2

SampleC:

- species IDs: 21,22,3,4,5,6,7,8,9,10
- relative abundance: 2:2:2:2:2:3:4:16:18:20

Basically:

- A and B high overlap on individual level, low overlap on species level
- A and C high overlap on species level, low overlap on individual level
- B and C low overlap on species level and low overlap on individual level

With error and 6X sequencing depth for rarest species

6X 60:54:48:12:9:6:6:6:6:6

Justifying the Concept of IGS

Hypothesis. For coverage spectrum as S1:S2:S3 across samples, the number of reads in these samples with such spectrum should have a ratio as S1:S2:S3

This is discussed as above with simple example with small number. As mentioned before, that is totally ideal situation, based on statistics of large number.

Get the number of reads with specific coverage spectrum across samples

	spectrum	sample1A	sample1B	sample1C
0	1-0-56	0	0	2
1	1-1-0	19649	19448	0
2	1-1-1	111	129	121
3	1-1-2	1	2	1
4	1-2-0	1125	4754	0
5	1-2-1	11	36	3
6	1-2-2	0	0	1
7	1-3-0	40	287	0
8	1-3-1	0	2	0

9	1-4-0	0	18	0	
10	2-0-0	20904		0	0
11	2-0-1	915	0	382	
12	2-0-2	1488		0	1454
13	2-0-3	2190		0	3418
14	2-0-4	2675		0	6035
15	2-0-5	2753		0	8104
16	2-0-6	2348		0	8925
17	2-0-7	1848		0	8583
18	2-0-8	1275		0	7134
19	2-0-9	782	0	5154	
20	2-0-10	415	0	3152	
21	2-0-11	210	0	1752	
22	2-0-12	76	0	909	
23	2-0-13	34	0	400	
24	2-0-14	19	0	167	
25	2-0-15	10	0	150	
26	2-0-16	25	0	177	
27	2-0-17	50	0	348	
28	2-0-18	106	0	688	
29	2-0-19	176	0	1238	
30	2-0-20	225	0	2100	
31	2-0-21	368	0	3326	
32	2-0-22	487	0	4862	
33	2-0-23	605	0	6692	
34	2-0-24	807	0	9348	
35	2-0-25	942	0	12331	
36	2-0-26	1087		0	15263
37	2-0-27	1278		0	18354
38	2-0-28	1419		0	21005
39	2-0-29	1388		0	23311
40	2-0-30	1519		0	25164
41	2-0-31	1517		0	26785
42	2-0-32	1529		0	27481
43	2-0-33	1379		0	26828
44	2-0-34	1228		0	26067
45	2-0-35	1140		0	23845
46	2-0-36	985	0	21812	
47	2-0-37	795	0	18797	
48	2-0-38	668	0	16541	
49	2-0-39	495	0	13199	
50	2-0-40	400	0	11038	
51	2-0-41	292	0	8676	
52	2-0-42	193	0	6467	
53	2-0-43	148	0	5018	
54	2-0-44	111	0	3337	
55	2-0-45	78	0	2433	

```
56  2-0-46  49  0   1773
57  2-0-47  35  0   1105
58  2-0-48  28  0   948
59  2-0-49  20  0   554
... ... ... ...
```

From the table above, the ratio of number of reads across samples with specific coverage spectrum correlates to the spectrum.

For low coverage reads, the correlation is not that obvious. This is related to the underlying concept about how to get IGS, from the concept of median coverage of a read. May need some statistics here.

Count the number of IGSs with that spectrum of coverage across samples

As discussed above, the ratio of number of reads across samples with specific coverage spectrum correlates to the spectrum in some degree, but not exactly.

So we do a normalizaton to the number of IGSs.

```
count = total number of reads with that spectrum / sum of numbers in spectru
m
```

In [62]:
```
MH=pd.read_csv('../data/sample1_ABC.freq.IGS_abund',delimiter=' ',names =
 ['IGS','count'])
MH
```

Out[62]:

28	0-0-29	95.241379
29	0-0-30	101.300000
...	...	...
9086	54-27-0	0.024691
9087	54-28-0	0.060976
9088	54-29-0	0.120482
9089	54-30-0	0.392857
9090	54-31-0	0.047059
9091	54-32-0	0.081395
9092	54-33-0	0.126437
9093	54-34-0	0.090909
9094	54-35-0	0.134831
9095	54-36-0	0.111111
9096	54-37-0	0.076923

9097	54-39-0	0.053763
9098	54-40-0	0.074468
9099	54-41-0	0.178947
9100	54-42-0	0.531250
9101	54-43-0	0.175258
9102	55-32-0	0.057471
9103	55-34-0	0.022472
9104	55-35-0	0.022222
9105	55-36-0	0.109890
9106	55-39-0	0.021277
9107	55-41-0	0.218750
9108	55-42-0	0.175258
9109	56-31-0	0.011494
9110	56-32-0	0.011364
9111	56-35-0	0.032967
9112	56-36-0	0.086957
9113	56-37-0	0.118280
9114	56-38-0	0.031915
9115	56-39-0	0.021053

9116 rows × 2 columns

List the IGSs and abundance in samples, (typical species-sample matrix as in classical ecology)

In [63]:
```
MH=pd.read_csv('../data/sample1_ABC.freq.IGS',delimiter=' ',names = ['IGS
_ID','sample1A_freq','sample1B_freq','sample1C_freq'])
MH
```

Out[63]:

28	29	0	0	1
29	30	0	0	1
...	...	...	...	...
898139	898140	50	35	0
898140	898141	50	35	0
898141	898142	50	36	0
898142	898143	50	36	0
898143	898144	50	37	0

898143	898144	50	37	0
898144	898145	50	37	0
898145	898146	50	38	0
898146	898147	50	38	0
898147	898148	50	39	0
898148	898149	50	40	0
898149	898150	50	41	0
898150	898151	50	42	0
898151	898152	51	29	0
898152	898153	51	30	0
898153	898154	51	32	0
898154	898155	51	33	0
898155	898156	51	34	0
898156	898157	51	35	0
898157	898158	51	36	0
898158	898159	51	37	0
898159	898160	51	38	0
898160	898161	51	39	0
898161	898162	51	40	0
898162	898163	51	40	0
898163	898164	51	41	0
898164	898165	51	42	0
898165	898166	52	32	0
898166	898167	52	33	0
898167	898168	52	38	0
898168	898169	53	40	0

898169 rows × 4 columns

With such IGS table, next we can use existing tools(like QIIME, Mothur) to do beta diversity analysis.

GOS data sets: Sorcerer II Global Ocean Sampling Expedition

– http://www.plosbiology.org/article/info%3Adoi%2F10.1371%2Fjournal.pbio.0050077

- 2007
- 454 sequencing, longer reads
- use 44 samples in testing

Using the pipeline as shown above to Get the distance matrix. (Here we use braycurtis metric.)

- Get the Samples-by-IGS table
- Get the distance matrix

For detailed pipeline, see: http://nbviewer.ipython.org/github/qingpeng/2013-diversity/blob/master/notebook/2013-diversity-summary_GOS.ipynb

In [64]:
```
import pandas as pd
import numpy as np
import scipy
from IPython.display import HTML
from scipy.cluster.hierarchy import linkage, dendrogram
from scipy.spatial.distance import pdist, squareform
from IPython.display import Image
from pandas import *
```

In [65]:
```
GOS=pd.read_csv('../data/matrix.out',delimiter=' ',header=None)
#label=pd.read_csv('../data/config-GOS.txt',delimiter='  ',header=None)
label = GOS[0]
label_list = []
for i in GOS[0]:
    label = i.split('.')[0]
    label_list.append(label)
len(label_list)
GOS=GOS.ix[:,1:46]
GOS.columns=label_list
GOS.index=label_list
GOS=1-GOS
GOS
```

Out[65]:

GS006	0.000012	0.000559	0.000057	0.000030	0.000000	0.000015	0.000016	0.061459
GS007	0.000019	0.000044	0.000011	0.000006	0.000099	0.000382	0.000101	0.058424
GS008	0.000025	0.000066	0.000070	0.000098	0.000039	0.000270	0.000135	0.014119
GS009	0.000018	0.000064	0.000040	0.000056	0.000000	0.000040	0.000042	0.020456
GS010	0.000051	0.000223	0.000264	0.000209	0.000059	0.000135	0.000114	0.014359
GS011	0.000006	0.000010	0.000000	0.000020	0.000000	0.000020	0.000021	0.001967
GS012	0.000006	0.000000	0.000009	0.000000	0.000000	0.000000	0.000031	0.000051

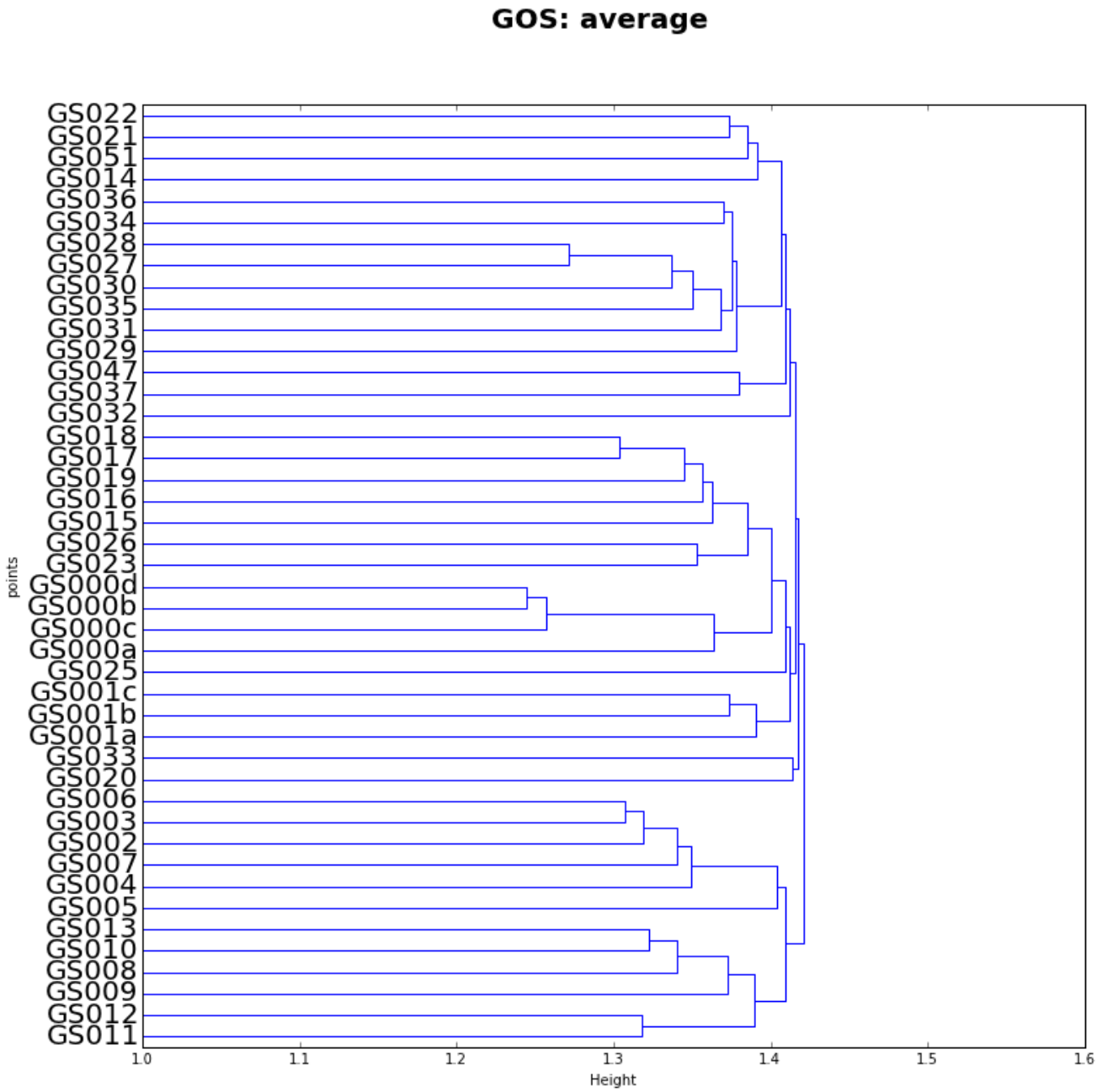
GS013	0.000028	0.000094	0.000077	0.000072	0.000082	0.000585	0.000273	0.012446
GS014	0.003086	0.010769	0.013403	0.010863	0.000252	0.000362	0.001865	0.000037
GS015	0.006198	0.017812	0.017787	0.019169	0.000399	0.001154	0.003875	0.000050
GS016	0.004597	0.016513	0.018208	0.016723	0.000654	0.000837	0.004245	0.000048
GS017	0.014642	0.041793	0.033322	0.043415	0.001351	0.002284	0.010382	0.000093
GS018	0.008084	0.022602	0.014933	0.024350	0.001055	0.001893	0.006845	0.000027
GS019	0.005547	0.015660	0.009168	0.016927	0.000725	0.001364	0.005419	0.000029
GS020	0.000016	0.000049	0.000064	0.000034	0.000000	0.000000	0.000011	0.000000
GS021	0.001206	0.004901	0.004804	0.004289	0.000099	0.000224	0.000690	0.000000
GS022	0.001221	0.004837	0.003737	0.005228	0.000251	0.000382	0.001387	0.000000
GS023	0.004374	0.011304	0.005280	0.012831	0.000557	0.001113	0.004490	0.000000
GS025	0.000564	0.003457	0.001150	0.002279	0.000339	0.000741	0.000391	0.000009
GS026	0.002600	0.007096	0.002874	0.008673	0.000370	0.000734	0.002962	0.000055
GS027	0.000925	0.003250	0.002090	0.003502	0.000140	0.000242	0.001199	0.000118
GS028	0.000859	0.003174	0.001651	0.003347	0.000121	0.000174	0.001255	0.000095
GS029	0.000327	0.001260	0.000806	0.001203	0.000026	0.000079	0.000363	0.000019
GS030	0.002111	0.007170	0.006314	0.007191	0.000141	0.000277	0.001320	0.000010
GS031	0.005142	0.014146	0.016115	0.015161	0.000062	0.000166	0.000501	0.000361
GS032	0.000052	0.000501	0.000254	0.000335	0.000058	0.000028	0.000009	0.000096
GS033	0.000017	0.000137	0.000014	0.000002	0.000007	0.000003	0.000000	0.000000
GS034	0.001425	0.001936	0.001404	0.001945	0.000110	0.000177	0.000750	0.000009
GS035	0.000265	0.000824	0.000771	0.000956	0.000063	0.000030	0.000231	0.000026
GS036	0.000245	0.000981	0.000628	0.000853	0.000031	0.000057	0.000434	0.000023
GS037	0.000081	0.000424	0.000254	0.000378	0.000011	0.000045	0.000128	0.000000
GS047	0.000606	0.001572	0.000654	0.001670	0.000041	0.000160	0.000860	0.000000
GS051	0.000919	0.003398	0.001953	0.003455	0.000259	0.000294	0.001691	0.000009

Do clustering

In [66]:
```
figure(num=None, figsize=(12, 12))
R = dendrogram(linkage(GOS, method='average'),labels=label_list, leaf_font_size=20,orientation='left')

ylabel('points')
xlabel('Height')
xlim(1,1.6)
suptitle('GOS: average', fontweight='bold', fontsize=20)
```

Out[66]: <matplotlib.text.Text at 0x10c4cfa10>



better than old reads coverage method, compared method, and original GOS paper

It seems that it has better resolution than older reads coverage method, compared method, even the cluster in GOS paper(based on alignment)! (see the figures below)

See GS000a, GS000b, GS000c, GS000d are clustered together as compareads, but not in GOS papers. And 16 is clustered together with 15,17,18,19, rather than out of 23,26 as in compareads, and old reads coverage methods.

also, 51,22,21,14 clusters are coordinate to GOS cluster, but not to compareads method, which makes more sense.

compareads Methods

http://www.biomedcentral.com/1471-2105/13/S19/S10

In [67]:
```
Image(filename="../Figure/comp.png")
```

Out[67]:

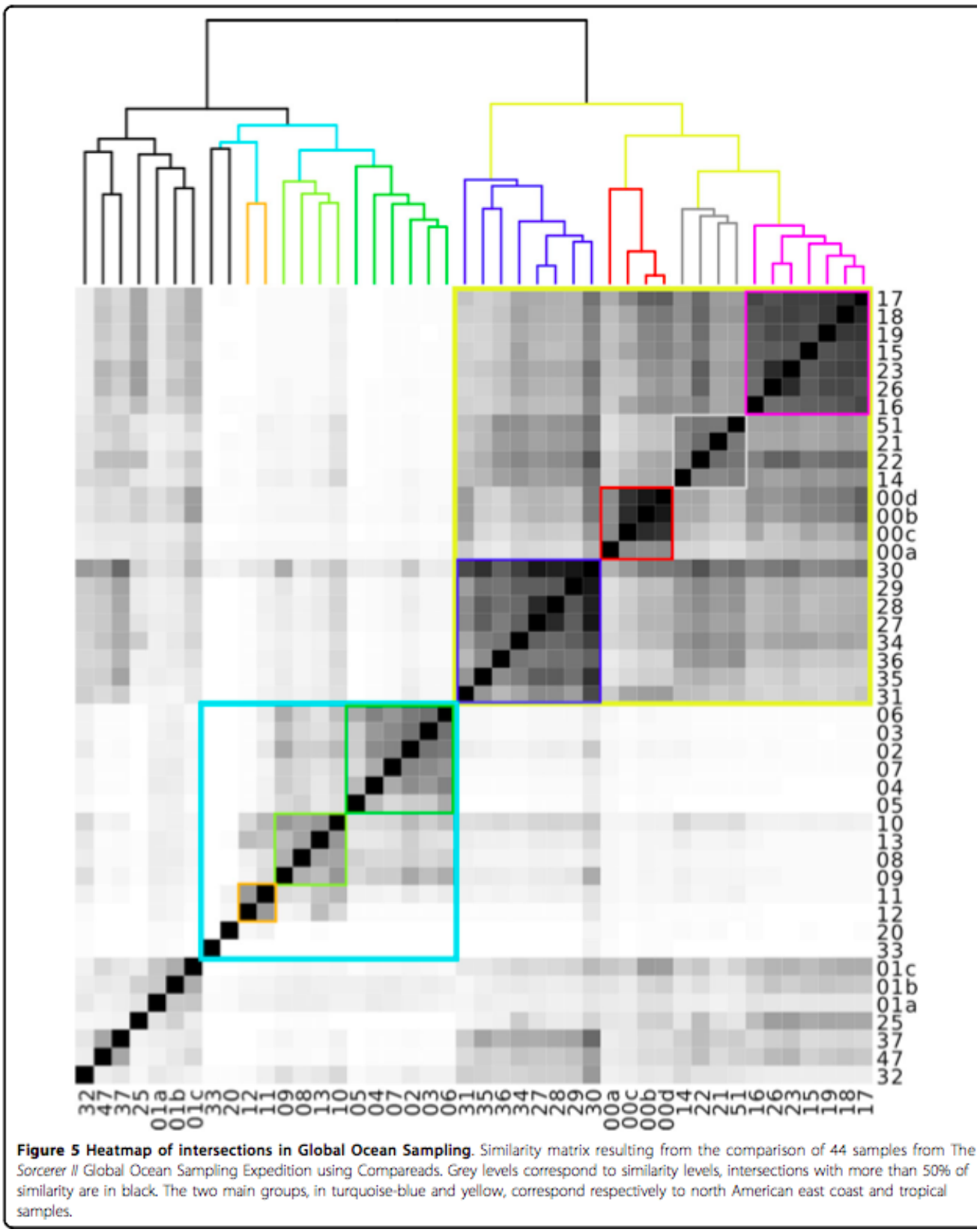
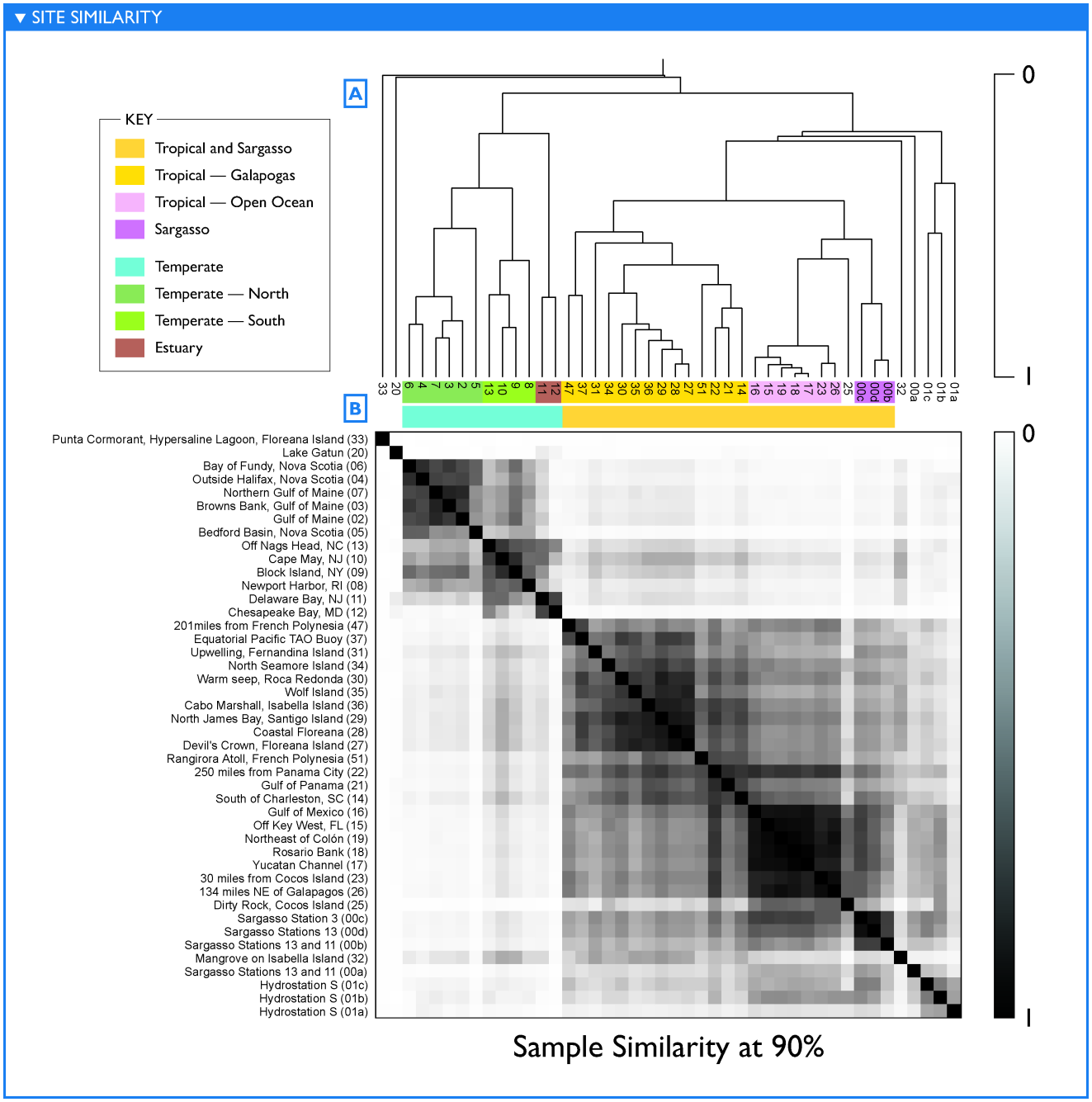


Figure 5 Heatmap of intersections in Global Ocean Sampling. Similarity matrix resulting from the comparison of 44 samples from The *Sorcerer II* Global Ocean Sampling Expedition using Compareads. Grey levels correspond to similarity levels, intersections with more than 50% of similarity are in black. The two main groups, in turquoise-blue and yellow, correspond respectively to north American east coast and tropical samples.

Original GOS Paper

In [68]:
```
Image(filename="../Figure/GOS1.png")
```

Out[68]:



MetHit Human Gut metagenomics data set

– http://www.nature.com/nature/journal/v464/n7285/full/nature08821.html - 2010 - Illumina reads, shorter - 570GB - 124 European adults, here picked 39 samples, 25 with Inflammatory bowel disease (IBD)

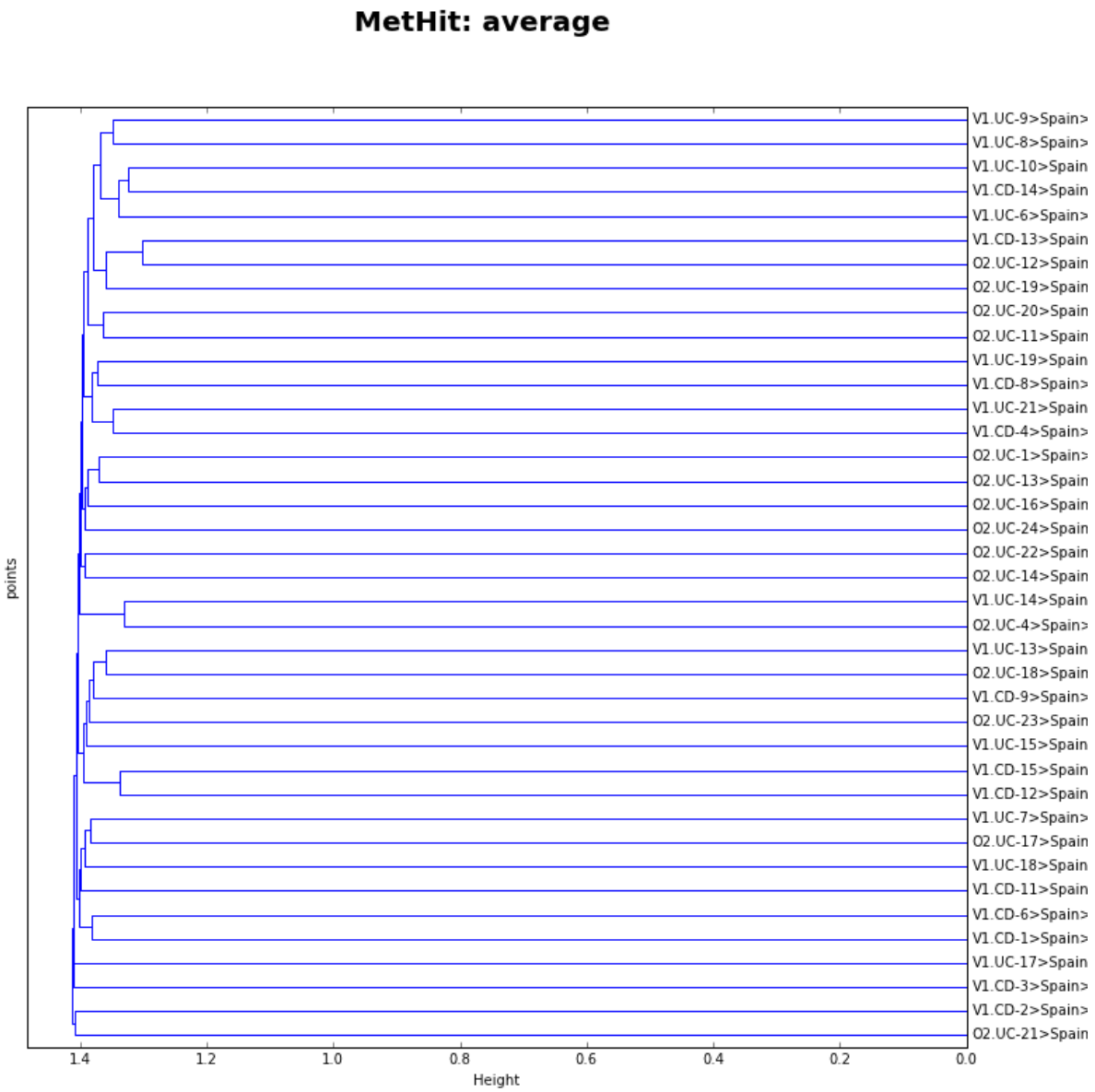
Using the method as shown above, also get the distance matrix and do ordination(PCoA).

Just there are 45,421,735 IGSs. So it took a while to finish the computing. But it is still doable.

For more details about analysis, see http://nbviewer.ipython.org/github/qingpeng/2013-diversity/blob/master/notebook/2013-diversity-MetHit.ipynb

In [69]:
```
Image(filename="../figure/metahit_cluster.png")
```

Out[69]:

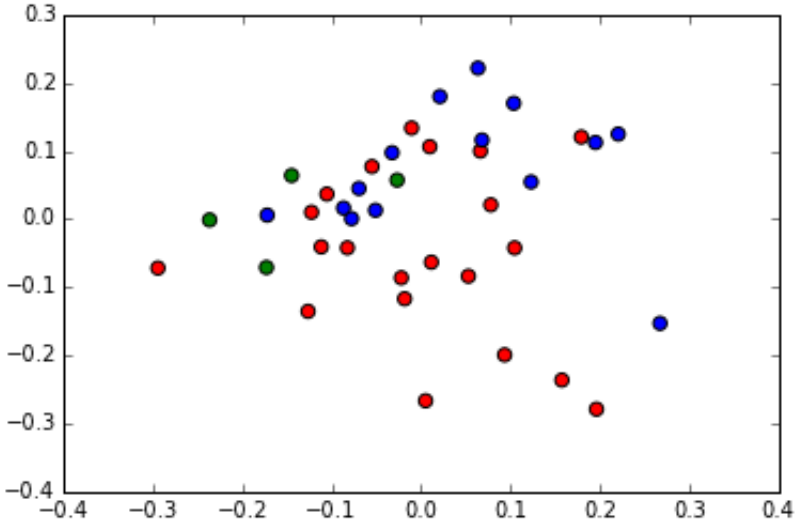


PCoA of samples according to disease

- RED: 21 ulcerative colitis
- GREEN: 4 Crohn's disease
- BLUE: 14 healthy individuals

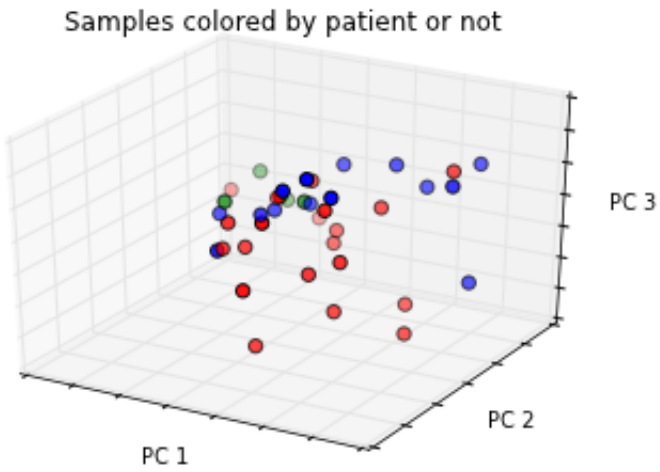
In [70]:
```
Image(filename="../figure/metahit_pca_2d.png")
```

Out[70]:



In [71]:
```
Image(filename="../figure/metahit_pca_3d.png")
```

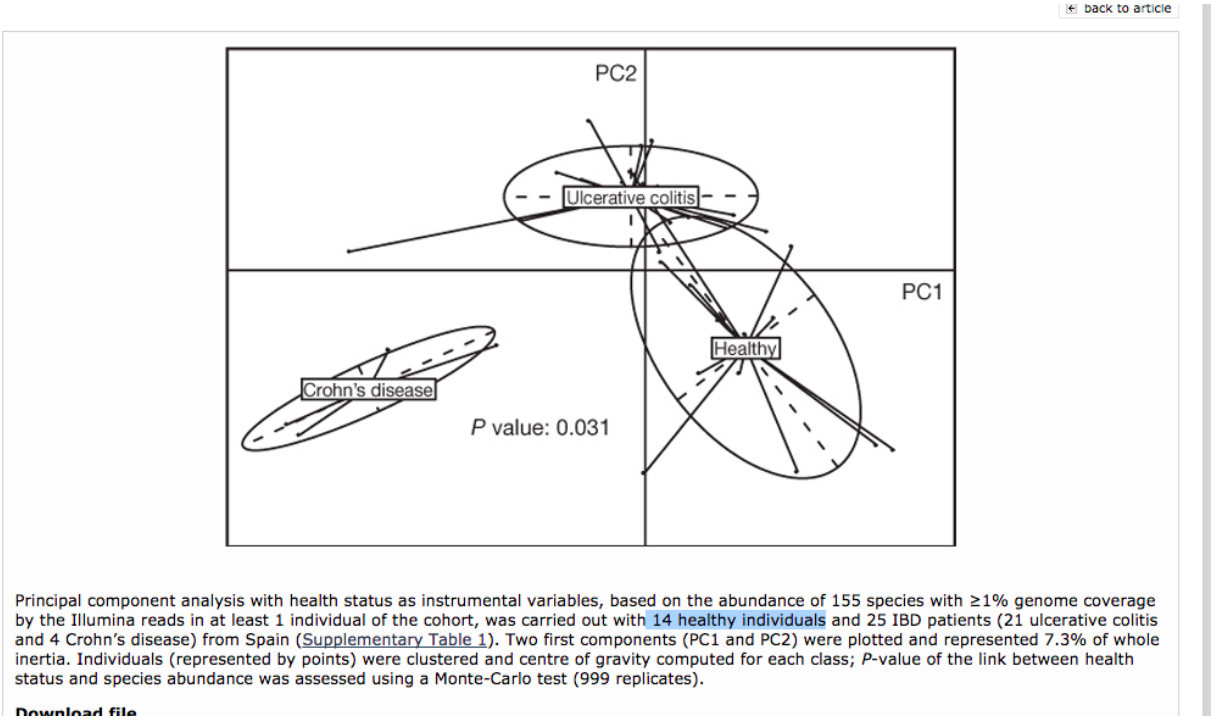
Out[71]:



- R statistic: An R value near +1 means that there is dissimilarity between the groups, while an R value near 0 indicates no significant dissimilarity between the groups. So here R=0.11 is not that promising. I am not sure if this is significant enough.
- P-value = 0.045 (using ANOSIM)
- Compared to 0.031 shown in Methit paper as above(the method to get this p-value may be different,I will check it), it is not bad.
    - Their method is based on mapping reads against 155 "core" species genomes.

In [72]:
```
Image(filename="MetHit.png")
```

Out[72]:



Principal component analysis with health status as instrumental variables, based on the abundance of 155 species with ≥1% genome coverage by the Illumina reads in at least 1 individual of the cohort, was carried out with 14 healthy individuals and 25 IBD patients (21 ulcerative colitis and 4 Crohn's disease) from Spain (Supplementary Table 1). Two first components (PC1 and PC2) were plotted and represented 7.3% of whole inertia. Individuals (represented by points) were clustered and centre of gravity computed for each class; *P*-value of the link between health status and species abundance was assessed using a Monte-Carlo test (999 replicates).

Download file

Other datasets:

- HMP
  - running
- EMP
- GPGC
  - 1,377,264,178 IGSs, running

In []:
```
```