1. **CDS-wide dN/dS for each line**
   1. ~~Make 2 dicts~~
      1. ~~61 codons; N and S value for each, ignoring nonsense mutations~~
      2. ~~64 codons; N and S value for each~~
   2. ~~Add up N and S ∀ codons that landed a (fixed) mutatio~~n
   3. In the above, I was wrong—according to Yang 2006 book, actually you use the codons across the entire sequence, not just the codons that have fixed differences.
      1. So you can just use the effective number of synonymous and nonsynonymous sites that gdtools reports.
         1. One problem is that Mm uses an alternative Trp codon.
      2. HOWEVER. If you want implement the effect of the transition/transversion ratio, you would need to write your own counting algorithm. This would be a harder problem, because you would have to look at each codon-codon change AND nt-nt change together.
         1. One way to implement would be to have a dict of ALL codon-codon changes as keys, as values as synonymous or nonsynonymous. Unfortunately, that means 64 \* 9 = 576 kv pairs. Meanwhile the dict of nt-nt changes would be a lot smaller, 4 \* 3 = 12.
         2. Additionally, if you had a nt-nt change dict, you could do something more complex than the ts-tv ratio, because you would have a relative rate for EVERY nt-nt change.
         3. You could use the 576 and the 12 dict together to generation a weighted 64 dict
   4. Calculate DN / N and DS / S (i.e., if N = S, you wouldn’t have to do this step ∵ the expected ratio would already be 1:1
   5. Correction for possible multiple hits: dN = -3/4 ln (1 – 4/3 (DN / N))
      1. C~~onsider skipping this correction because the divergence is low and multiple hits are unlikely~~. That said, the effect of the correction is small when divergence is low, so don’t skip it!
2. **Calculate a “G score” for each gene and a corresponding *P*-value (obtained via simulations)**.
   1. For each strain:
      1. Make a dict of gene IDs and lengths.
      2. Use this information to run 10000 simulations in which the total number of mutations (some polymorphic with the values from the actual data, plus a number of fixed ones with values from the actual data) are placed at random across the total CDS (i.e., the probability of being place in any given gene is proportional ONLY to its length).
         1. Using only nonsynonymous mutations.
      3. For each gene, obtain (from 10000 simulations) mean and SD number of mutations. Use this mean and SD to calculate a Z score and corresponding *P*-val for the OBSERVED number of mutations.
      4. Do you have to worry about FDR? Possibly. You’re carrying out 500 tests… But they’re not all testing the same hypothesis? I think you DO have to do FDR. If you don’t, then you expect to see 25 (for syn3B; or 50, for syn1.0) “significant” parallel genes by chance alone.
3. **Use PCoA and perMANOVA to see if syn3B and syn1.0 cluster separately**.
   1. First, make a dataset for syn1.0 that ignores mutations if they’re in genes missing from syn3B.
   2. Make a gene-by-population matrix for each strain.
   3. Do QB things.
4. Values of N and S generated by gdtools COUNT -b. These values are incorrect because i> UGA codes for Trp in *Mycoplasma*. I have confirmed this for syn1.0. In fact, [89% of Trp codons in *M. mycoides*](http://www.kazusa.or.jp/codon/cgi-bin/showcodon.cgi?species=2103&aa=4&style=N) are UGA, only 11% are UGG.
   1. 3B
      1. N (possible nonsynonymous total): 1081335
      2. S (possible synonymous total): 287360
   2. Syn1.0
      1. N: 2079386
      2. S: 544861