Minutes for Borrelia Project Meeting

SFU

June 3, 2017

1 Details

1. **May 12**
   Changing standard deviation of DNA fragment size in ART simulator from 30 to 10 because there are reads which are generated from true variants but do not map to true variants. Calculations done by Elijah showed that it should be around 10. After changing to 10, this error did not occur.

   Just in case this happens, find the maximum number of mismatches in that data set, max. Penalized the true variants by incrementing its objective value by max+1.

2. **May 16**
   Further investigated into an example of the simulated data in which the recall is low. We found out that if we let the ILP searches for more solutions(with most aggressive settings and solution gap=0), it does find the optimal solution. Statistics in $simulated\_stats2$ folder on Github are produced based on setting variants predicted to be equal to the solution output by the ILP which matches the true variants. This boosts the precision and recall values.

   We ran the ILP on real data and observed reads which map to a variant with 16 mismatches were also reported. We believe that this should not be used in our program. Therefore, we limit the number of mismatches in Bowtie. We tried limiting to 3,2,1,0 mismatches. Even if we limit to 0 mismatches, the ILP finds multiple optimal solutions. Also, we noticed that the number of optimal solutions decrease for some genes from 3mm to 0 mm. This is expected as we were being more stringent. However, whether being more stringent is "good", we cannot be sure as we tested on real data. Hence, we test on simulated data and initial observation shows that limiting to 2mm gives slightly better statistics i.e. recall, precision, total variation distance etc. This statistics are on Github, $simulated\_stats2$ folder. But we still have the case that there are multiple optimal solutions.

   Cedric and I analyzed one of the genes pepX on real data. We found that among the optimal solutions, there is one which is {pepX_1} and another one which is {pepX_1, pepX_17}. We find that the reads in concerned, as a whole maps well to pepX_1 and maps poorly to pepX_17. We were wondering why {pepX_1, pepX_17} is an optimal solution. The ILP is constructed in a way that for each read r, the ILP will try to pair r with a variant v in which number of mismatches of r mapping to v is the minimum across all variants. So if a read r maps to pepX_1 and pepX_17 with mismatches 0 and 2 respectively, r is "assigned" to pepX_1 in this case. I think the solution {pepX_1, pepX_17} exists because there exists a read which maps better to pepX_17, with one less mismatch compare to the case if we map to pepX_1. Therefore this solution will have same objective value as solution {pepX_1}.

3. **May 17**
   Cedric has an idea to discriminate between solutions using methods similar to GAML.

   (a) Define the set of optimal solutions OPT=$\{opt_1, opt_2, ...\}$. Define the matrix of mismatches M_i for each opt_i. The idea is to come up with a score for each opt_i, where given a set of variants opt_i, how likely it is to observe the reads.

(b) For each opt_i, we compute the likelihood $P(R|opt_i) = \prod_{\forall r \in R} P(r|opt_i)$.

(c) We want to find an opt_j which maximizes $\prod_{\forall r \in R} P(r|opt_i) \Leftrightarrow$ minimizes $\sum_{\forall r \in R} -logP(r|opt_i)$

(d) $P(r|opt_i) = \sum_{\forall v \in opt_i} \frac{H(r,v)}{(2 \cdot 76 \cdot |opt_i|)}$, where $H(r,v) = (0.99)^{(76-m)} \cdot (0.01)^{(m)}$ and m=number of mismatches mapping r to variant v, which can be found in M_i.

4. **May 19**
Among simulations which are predicted correctly, about 10-20% of the true solutions do not have maximum likelihood.

Cedric's initial observation suggests that the true variants are covered by solutions which have close likelihood score to the maximum likelihood score. His idea is to take the union of variants of solutions which have likelihood score within ϵ% of the maximum likelihood score.

We analyzed the gene rplB for sample SRR2034333, in which there are two optimal solutions {rplB_1} and {rplB_94} which have same likelihood score. These two variants differ by a SNP at position 609. The reads do not cover this position, hence there is no way to distinguish between these two variants. By having existing strains as prior, we can heuristically choose the one which favor prior observation.

5. **May 23**
The union method gives very high recall statistics, but not precision.

Discussion with Leonid:

(a) Document reasons for changing standard deviation in ART from 30 to 10

(b) Replace simulated data on Github with new simulated data

(c) Update statistics based on new simulated statistics

(d) Check that there are no more negative numbers in Δ(objective value)

(e) Elijah: Make the simulation fully deterministic by translating Bash to Python

(f) Consider modifying objective function with linear combination: number of mismatches + k $\cdot$ number of variants where k can be calibrated by simulated data

(g) Consider the modified objective function inspired by minimum description length formalism(Leonid will think about this)

(h) Consider ML directly as first stage optimization instead of second stage, encoded as an ILP

Done:

(a) Reorganized github folders and scripts

(b) Documented minutes in past discussions

(c) Updated statistics and uploaded to simulated_stats folder

(d) Uploaded latest simulated data

(e) So far no negative $\Delta objective$, will try more simulations

6. **May 24**
We need to wrap up the project in 2 weeks.

To do

(a) Compute some statistics to see whether there is a clear separation of solutions based on likelihood. If yes, get the percentage of cutoff, group solutions with likelihood score within the cutoff.

(b) Compute coverage for all samples.

(c) Consider a prior on number of strains?

(d) Generate simulated data on 3 coverages(30X, 100X, 300X)

(e) Issue about two optimal solutions with one variant each, both having same likelihood score. This could result in possibility of new strains. Check by hand in the end and heuristically favor existing strains.

(f) Simulations based on strains. Strain A 50%, strain B 25%, strain C 25%. 100X coverage in total, 50X coverage for A, 25X for each B and C.

(g) MetaSim?

(h) Negative difference in objective values

Done:
Investigating why negative differences in objective values happened. Observed bunch of reads not mapping to true variants, which is weird. Suspect that the functions which simulates data and generates matrix have bugs. Investigate tomorrow with Elijah

7. **May 25**
   Done

   (a) Wrote a script to download all samples, it's in pipeline folder.

   (b) Solved delta objective solution being sometimes positive and negative, due to incrementing the wrong count

   (c) Rewrote the ART simulations in python(part of it?)

   (d) Tried setting -k 0 and -s 10 in ART, no error where reads do not map to true variants. However, there are some issues in retrieving the number of mismatches for predicted solutions (all are -1?)

8. **May 26**
   To Do:

   (a) Try on different settings of art simulator and observe whether the issue where read does not map to true variants still happen

   (b) Calibrate the threshold for grouping maximum likelihood solutions

   (c) Different coverages and reproduce statistics

   (d) Migrate bash to python? Ask Leonid and Cedric

   (e) Make a master script to run all simulations and reproduce same statistics

   (f) Get Elijah together to clean code

   (g) Elijah's going to try on these settings of art-s 10 -k 0 on coverage 30X 100X 300X

   (h) Time and memory consumption

   Done:

   (a) Solved the weird issue in 7)d) due to numerical issues. Comparing a decision value with value = 1.0 to 1.0 gives false. Changed condition to be > 0.5, this works as all decision variables have maximum value = 1

   (b) Regenerated statistics(in simulated_stats) with art -k 0 -s 10, uploaded on Github. This statistics was produced by matching predicted solutions to true variants

   (c) Setting art -s 10 still gives the problem where a read does not map to true variants. (When it happens, it's only a pair of reads). Investigated and maximum mismatches is 3, hence changed the limit of mismatches to 3 when mapping using bowtie.

   (d) Setting art s-30 will give the same issue in (c). Using -k 3 and -s 10 will not have this issue.

   (e) Regenerated statistics using -k 3 and -s 10. Statistics were produced by matching predicted solutions to true variants

9. **May 29**
   To Do:

   (a) Calibrate the threshold for grouping maximum likelihood solutions

   (b) Different coverages and reproduce statistics

   (c) Migrate bash to python? Ask Leonid and Cedric

   (d) Make a master script to run all simulations and reproduce same statistics(partially)

(e) Elijah's going to try on these settings of art-s 10 -k 0 on coverage 30X 100X 300X

(f) Time and memory consumption

Done:

(a) Generated scatter plots for the likelihood of different solutions. Red dots represent those solutions in which their union contain true variants, starting from minimum negative log likelihood solution.

(b) Cleaned code and implemented the master script for running simulations(partially)

10. **May 30**
To Do:

(a) Calibrate the threshold for grouping maximum likelihood solutions

(b) Different coverages and reproduce statistics

(c) Migrate bash to python? Ask Leonid and Cedric

(d) Elijah's going to try on these settings of art-s 10 -k 0 on coverage 30X 100X 300X

(e) Time and memory consumption

Done:

(a) Cleaned code and implemented the master script for running simulations. Uploaded on github in simulation folder

(b) Changed a little bit of the scatter plots, if the union of all solutions do not cover true variants, then paint everything as green dots.

(c) Elijah implemented art_illumina for different coverages, instead of setting the number of reads to generate, he made use of the coverage parameter for art. Observation: Recall increases, close to 100 but precision decreases (as low as 87) as coverage increases from 30X to 300X.

11. **May 31**
To Do:

(a) Document change of codes due to coverage parameter

(b) Paired end reads approach: Change the generate_matrix function, use counting method to compute proportions and the Bayes'+binomial method. Observe which gives better statistics

(c) Change how we simulate reads, changing coverage parameter rather than setting number of reads

(d) Different coverages and reproduce statistics

(e) Calibrate the threshold for grouping maximum likelihood solutions (If paired end reads approach still do not give unique optimal solutions)

(f) Figures needed for paper: Refer to Leonid's google doc

(g) Migrate bash to python? Ask Leonid and Cedric

(h) Time and memory consumption

Done:

(a) Documented an extra parameter for run_sim.py, which is the coverage parameter

(b) Changed the generate_matrix function to accommodate paired end reads, implemented paired end reads method

(c) Implemented counting based proportion calculation method

12. **June 1, 2**
To Do:

(a) Compare the counting method of computing proportions vs the Bayes method and record results.

(b) Perform tests on 30X, 100X, and 300X coverage.

(c) Update Leonid and Cedric on the results.

(d) Calibrate the threshold for grouping maximum likelihood solutions (If paired end reads approach still do not give unique optimal solutions)

(e) Figures needed for paper: Refer to Leonid's google doc

(f) Migrate bash to python? Ask Leonid and Cedric

(g) Time and memory consumption

(h) Output statistics results as separate csv files, for plotting purposes

Done:

(a) Ran simulations on 30X,100X, 300X

(b) Migrated pipeline from bash to python

(c) Compared Bayes' and counting: Counting is better

(d) Tried on weighted Bayes, seems good

13. **June 3,4,5**
To Do:

(a) Output statistics results as separate csv files, for plotting purposes

(b) Proportion calculation: Weighted Bayes'?

(c) How to solve multiple optimal solution issue?

(d) Figures needed for paper: Refer to Leonid's google doc

(e) Time and memory consumption