For each value of R\_0, perform 1000 realizations. For each of the 1000 realizations, you let the population reproduce until there are 6,000 live viruses (ones that have not had their "burst day"). Take M=8,16,32,...,256,512 samples, and for each number M of samples, calculate A\_{m} (m = 1,2,...,M) by counting the mutable (non-founder) ancestors that have m descendants in the sample. Multiply each A\_{m} by mu = 0.0551, to get the mean of a Poisson distribution, and sample each Poisson distribution to derive eta\_{m} (m = 1,2,...,M) for the realization. If eta\_{m} = 0 (m = 2,...,M), the realization "fails". Count the failed realizations. Otherwise, if the realization succeeds, calculate eta\_tilde\_{m} (m = 1,2,...,M\_tilde) from eta\_{m} (m = 1,2,...,M). Plug eta\_tilde\_{m} into the various estimates of R\_0 and estimated errors for the estimates, and push each number onto its respective vector. At the end of the (1000) realizations, you calculate the fraction of failed realizations. For the successful realizations, you calculate from the vectors the mean R\_0 estimate, the sample standard deviation R\_0 estimate, and the mean estimated error of the R\_0 estimate.

--

Generate an array of R\_0 values to test. For each value of R\_0, perform 1000 realizations. Each realization consists of generating a random seed, letting a population reproduce from a single founder to 6000 live (un-burst) viruses, and calculating estimates for each individual realization of R\_0, the sample standard deviation R\_0 estimate, and the mean estimated error of the R\_0 estimate. I am concerned that <https://www.youtube.com/watch?v=V2f-MZ2HRHQ> You cannot calculate sample standard deviation of the R\_0 estimate until you have the entire sample of 1000 R\_0 estimates. The procedure that I intended follows. For each successful realization (some eta\_{m} > 0 for m > 1), calculate the estimated R\_0 and its estimated error, as an actual investigator would, given the site-frequency spectrum. Push the estimated R\_0 and the estimated error in R\_0 onto separate arrays. After the last realization for the value of R\_0, calculate for the sample arrays: (1) the mean of the ln R\_0 estimates; (2) the standard deviation s of the ln R\_0 estimates (using sqrt( n – 1 ) not sqrt( n ), to avoid downward bias); and (3) the mean s^ of the estimated errors. Our different definitions of s^ are probably responsible for the unexpected behavior of s^, although the simplified estimate of MLE error should be implemented. You will need the Hessian for the multiple founder case, so do not delete the code. These estimates are pushed onto an array. Each realization is outputted into a raw\_data CSV file and the calculated means are outputted into a means CSV file.

The population reproduces as follows: first a single founder virus is created and added to a new array which will contain unburst viruses. While the size of this array is below the 6000 threshold, the list is first sorted by age The unburst (“live” is a more suggestive term than “unburst”?) viruses is a linked list not an array (you probably have this correct) because each of the frequent insertions needs to invoke a constant time-cost. If you are repeatedly sorting the entire sorted list, however, your code is unnecessarily slow. I used a merge sort to place only the new daughters, already sorted by their burst-day, into the list of live viruses. The merge starts from the last burst-day backwards to reduce the merge time. and then a random number, according a Poisson distribution, of daughters is added to the oldest virus in the array, with each daughter assigned a burstday randomly according to a Gamma distribution. These daughters are then added to the array and the parent virus is removed. If, at any point, the array becomes empty – i.e. the founder virus’s lineage goes extinct – a new random seed is generated and the process begins anew.

The calculations proceed as follows: first, M samples are taken from the live array and A\_{m} (m = 1,2,...,M) is calculated by counting the non-founder ancestors that have m descendants in the sample. Although immaterial in our present study, A\_{1} (“the mutable ancestors”) includes the samples themselves (according to the convention that an individual is included among its “ancestors”). Please ensure that your code includes the samples in A\_{1}. Eta\_{m} (m = 1,2,...,M) is calculated by sampling Poisson distributions around mu = 0.0551 multiplied by each A\_{m}. Eta\_tilda\_{m} (m = 1,2,...,M\_tilde) is then calculated by folding eta\_{m}. From eta\_tilda\_{m}, R\_0 is estimated by maximum likelihood estimation, with and without an Euler-Maclaurin approximation, and the method of moments. To save computing time, the unapproximated likelihoods and the function h(r) are only calculated for values in the R\_0 array. Each value of the likelihood function or h(r) minus the sum of eta\_tilda\_{m} from m=2,3,…,M-2 is placed in an ordered array and the index of the maximum (for MLE) or minimum (for method of moments) for method of moments, presumably you mean the minimum (absolute) difference between *h*( *r* ) and the actual sum of eta\_tilda\_{m} from m=2,3,…,M-2 is found. The corresponding index of the R\_0 array is then pushed onto an output array. Standard deviations are calculated from these estimated R\_0’s and pushed onto the same output array.

Once the entire simulation is complete, these arrays are outputted directly into a “raw\_data” CSV file. Means of this array are then taken and sample standard deviations are calculated. These results are then pushed onto another array and outputted into a “means” CSV file.