An accurate genetic clock

David Hamilton *

*Department of Mathematics, University of Maryland, College Park

Our method for "Time to most recent common ancestor" TMRCA of genetic trees for the first time deals with natural selection by apriori mathematics and not as a random factor. Bioprocesses such as "kin selection" generate a few overrepresented "singular lineages" while almost all other lineages terminate. This non-uniform branching gives greatly exaggerated TMRCA with current methods. Thus we introduce an inhomogenous stochastic process which will detect singular lineages by asymmetries, whose "reduction" then gives true TMRCA. Reduction implies younger TMRCA, with smaller errors. This gives a new phylogenetic method for computing mutation rates, with results similar to "pedigree" (meiosis) data. Despite these low rates, reduction implies younger TMRCA, with smaller errors. We establish accuracy by a comparison across a wide range of time, indeed this is only y-clock giving consistent results for 500-15,000 ybp. In particular we show that the dominant European y-haplotypes R1a1a & R1b1a2, expand from c3700BC, not reaching Anatolia before c3300BC. This contradicts current clocks dating R1b1a2 to either the Neolithic Near East or Paleo-Europe. However our dates match R1a1a & R1b1a2 found in Yamnaya cemetaries of c3300BC by Svante Pääbo et al, together proving R1a1a & R1b1a2 originates in the Russian Steppes.

Molecular clock | Genetic tree | y-haplotype | Kin Selection

Abbreviations: TMRCA, STR, SNP, R1b1a2, R1a1a

Introduction

The genetic clock, computing *TMRCA* by genetic mutations, was conceived by Emile Zuckerkandl and Linus Pauling[30] on empirical grounds. However work on genetic drift by Motto Kimura[15] gave a theoretical basis and formula. Soon after pioneering work by L.L. Cavalli-Sforza [6], correlated genetic drift to age of lineages for human populations. Suppose at position j on the genome is distinguished by number x which in the next generation has mutation $x \to x \pm 1$ occurring at rate μ_j. Measuring total variance V from the mode [22] one finds that the TMRCA $= V/(\sum_j \mu_j)$. This method and variations (denoted as KAPZ) is used to estimate the TMRCA of y(chromosome) haplotypes defined by a SNP (single nucleotide polymorphism) mutation.

In practice sample sizes were too small to compute accurate mutation rates from "meiosis", i.e. father-son pairs[4]. Alternatively, estimating rates from genetic lineages of known age gave rates with significant discrepancies between different lineages. Indeed for the y-clock these "phylogenetic" rates are often 2 times larger than those from meiosis, while the opposite may be true for other clocks [2], [9], [14], [16].

For the Y-chromosome we show that the mutation rates are essentially constant, at least for the time scale 500- 15,000 ybp, and over different lineages. However KAPZ cannot give accurate TMRCA, i.e. one needs deeper mathematics to deal with non-uniform branching. Also there is a paradox: we can accurately estimate the mutation rates of "short tandem repeat" (STR) at different **D**NA **Y**-chromosome **S**egments (DYS). But we find they can differ by more than a factor of 100, so over a very long time scale we expect their rates to vary as the genomes geometry changes. Also we find knowing the average mutation rate does not give accurate TMRCA.

Of course it was noticed that the mathematics underlying KAPZ is most accurate for large populations, indeed continuous distributions, whereas actual populations are small. In this case the same stochastic model generates many discrete distributions, indicating a need for Bayesian methods. These use Monte-Carlo simulations of all possible genealogical trees giving the present sample data, then find TMRCA by a maximum likehood estimate (MLE). An example of this for the y-clock is BATWING[30]. However we shall see that Bayesian methods exaggerate the TMRCA even more than KAPZ. Also MLE is known for large confidence intervals. So our approach is different.

In particular for the y(chromosome)-clock the results have not been reliable. (Similar discrepancies occur for the mitrochondrial clock for "out of Africa", or for the allele clock for human-chimpanzee divergence [9], birds [15], bacteria[19].) A KAPZ due to Zhivotovsky [29] was applied to the y-haplotype R1b1a2 by Myres [18] giving 9000BC, standard deviation $\sigma = 2000$. Now for BATWING the *TMRCA* is often greater than KAPZ, e.g. for the Cinnioglu[7] study of Anatolian DNA both methods were applied to the same data and mutation rates. For R1b1a2 the KAPZ has TMRCA 9800BC compared with 18,000BC for BATWING. Balaresque [3] used BATWING to give an origin for R1b1a2 in Neolithic Anatolia c6000BC, but their statistics was disputed by Busby [5]. In verifying the accuracy of our method we simultaneously resolve the problem of the expansion of European y-haplotypes, for example R1b1a2.

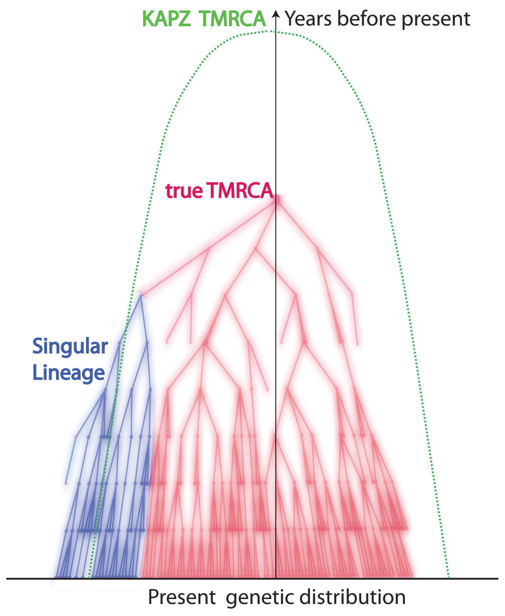


Fig. 1. Random tree

Singular Lineages

A fundamental problem is that present populations have highly overrepresented branches we call *singular lineages*. A well known example is the SNP L21 which is a branch of R1b1a2. Individuals identified as L21 are often excluded from R1b1a2 analysis because they skew the results. Such a singular lineage causes the variance to be much greater, even though the original *TMRCA* remains unchanged, see figure 1.

For Bayesian methods such lineages are very unlikely giving an even greater apparent *TMRCA*. However one cannot deal with singular branches by excluding them. For one thing, our method will show that 50% of markers show evidence of singular side branches, i.e. more than a SD from expected. Excluding them would also remove some of the oldest branches and produce a *TMRCA* which is too young. Now these singular lineages are very (mathematically) unlikely to arise from the stochastic system which is the mathematical basis of KAPZ (or the equivalent Monte-Carlo process modeling BATWING). We believe that the standard stochastic process is perturbed by "improbable" biological processes.

First, the Watson-Galton process[18] implies lineages almost certainly die out. Conversely, natural selection causes some branches to flourish, e.g. the "kin selection" of W.D. Hamilton[13], shows kin co-operation gives genetic advantages. Consider three examples with well developed DNA projects. Group A of the Hamiltons has approximately 100,000 descended from a Walter Fitzgilbert c 1300AD. Group A of the Macdonalds has about 700,000 descendants from Somerfeld c1100AD, and Group A of the O'Niall has over 6 million descendants from Niall of the Seven Hostages, c300AD. These are elite groups with all the social advantages. One sees lines of chieftains, often polygamous. We emphasize kin selection because it seems dominant over natural selection for recent branching, certainly we do not think the O'Niall are genetically superior! Natural selection would cause similar branching over longer time scales. Our model has many extinct twigs with a few successful branches, whereas current models assume a uniform "star radiation".

Reduction of Singular Lineages

Although our method is for general molecular clocks to be specific we focus on the y-clock. Consider **D**NA **Y**-chromosome **S**egments (DYS) counting the "short tandem repeat" (STR) number of nucleotides. One uses many of these DYS microsatellites, marked by $j = 1, ...N$, each individual i, $1 = 1, ..n$, has STR number $x_{i,j}$. The Y-chromosome is passed unchanged from father to son, except for mutations $x_{i,j} \to x_{i,j} \pm 1$ occurring at rate μ_j.

Modelling singular lineages requires a new stochastic system where instead of a single patriarch we imagine many "virtual patriarchs each originating at a different time and giving a fixed proportion of the present population. Solving for these times and proportions is an inversion problem. But inversion is unstable for such systems, also there is no unique solution. However it turns out that, up to a standard deviation, most DYS markers show at most one singular branch which is found from asymmetries in the distribution. These singular branches are then *reduced* revealing the original lineage. We then compute a branching time t_j for each marker j. Now the nonuniform branching process causes the t_j to be randomly distributed so their mean is not the TMRCA see figure 2. Large errors in mutation rates means one cannot simply take the max t_j to be the *TMRCA*. Instead stochastic simulations of the branching process, using robust statistics to avoid outliers, find the most likely *TMRCA*. The effect of reduction is dramatic, e.g. the TMRCA for R1b1a2 changes from 5500BC(KAPZ) to 3700BC after singular reduction, using the same markers and mutation rates, see Figure 3 and Table 1.

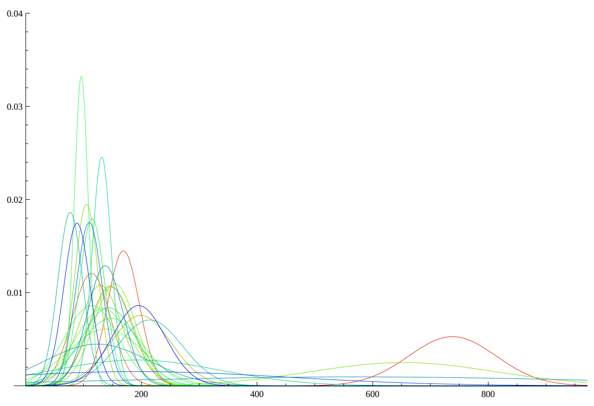


Fig. 2. Branching times t_j times(with errors) for R1b1a2 after reduction

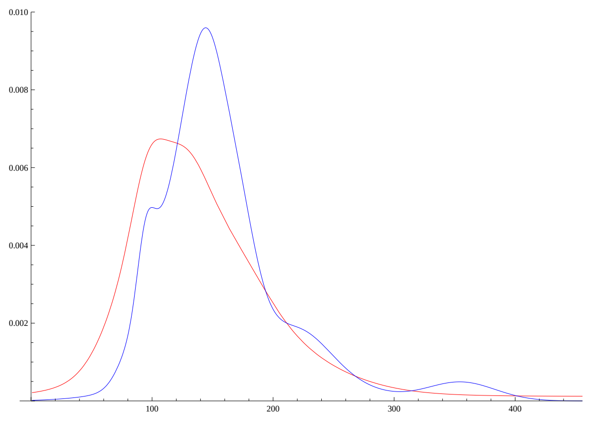


Fig. 3. R1b1a2 branching times before(blue) and after (red) reduction

Accurate Mutation rates

By relying on asymmetries of the distribution to find singular lineages we have to be aware that the mutation process itself might not be symmetric. Indeed if ignored we might be just detecting these asymmetries. So the symmetric model has to be changed so the probability of a mutation is

$$Pr[x_{i,j} \to x_{i,j} + 1] = \mu_{j,+1}\ , Pr[x_{i,j} \to x_{i,j} - 1] = \mu_{j,-1}\ .$$

If this marker is free from singular lineages we find that the ratio of the frequencies to the left and right of the mode is

$$\frac{P_{j,1}(t)}{P_{j,-1}(t)} = \frac{\mu_{j,1}}{\mu_{j,-1}}\ .$$

which is time independent. So using eight very large SNP projects we find enough markers free of singularities to compute these ratios and their standard deviations. See Supplementary Information (SI) where Figure 5 shows results. In particular about half the markers show asymmetric ratios are significant, i.e more than two SD from ratio 1. These asymmetric ratios play a very important role, for this ratio is all you need to detect a singular lineage and reduce it. Of course not knowing the exact asymmetric ratio means that bootstrap methods are used extensively for singular reduction, both to compute values and SD.

These methods also imply a new way of computing mutation rates. Previously, there were methods based on meiosos data or phylogenetic studies of family DNA projects (which gave quite different rates). We begin with 8 very large SNP projects from FTDNA using 37 markers, of course with unknown $TMRCA$. We first reduce singular lineages. Then taking asymmetry into account we find mutation rates are the fixed points of an iterative process. This takes about 3 iterates to converge. These mutation rates are normally distributed with mean and SD. Discarding markers with mutation $SD > 33\%$ leaves us with 29 markers. We find this advanced phylogenetic method gives mutation rates close to those obtained from meiosis and nearly 1/2 the values obtained from the usual phylogenetic method. Further validation comes from finding that the equivalence of our rates with meiosos implies *apriori* a human generation of c27 years.

Results

Accuracy is verified by checking for consistency over the whole range of European history beginning with the medieval:

Group	*TMRCA*	*[SD]*	Origins
Hamilton	1358*AD*	[140]	Fitzgilbert *d*1330*AD*
Macdonald	900*AD*	[250]	Norse *c*800 − 1000*AD*
O'Niall	200*AD*	[225]	Niall 300AD/Conn100AD

Archeological finds convinced Marija Gimbutas to attribute Proto Indo-European (PIE) to the Yamnaya Culture c 3500BC of the Russian Steppes, see [12]. This is consistent with mainstream linguistic theory, some even wrote of linguistic DNA. But actual genetics was ignored because current genetic clocks for R1b1a2 pointed to the Renfrew Hypothesis that PIE spread from Neolithic Anatolia, c 6000BC [34]. Or Mesolithic or Paleolithic, depending on the genetic clock. However noone checked if their clock worked over the whole range of time for different lineages.

The next table shows the expansion times of the dominant European y-haplotypes R1b1a2 & R1a1a. These are very close to c3700BC, only Scandinavia is significantly later. This data is from FTDNA projects for region X only using individuals with named ancestor from X. These independent results agree within the standard deviation, with dates matching the Corded Ware Culture, a semi-nomadic people with wagons and horses who expanded west from the Urkraine c3500BC. This is consistent with the oldest R1b1a2, R1a1a skeletons being from the Yamnaya Culture, c 3300BC, see S. Pääbo et al [24].

Region	*R1b1a2[SD]*	*n*	*R1a1a [SD]*	*n*
All	3700*BC*[625]	460	3800*BC*[700]	1270
Russia	*NA*		3750*BC*[700]	337
Poland	3960*BC*[950]	65	4600*BC*[820]	876
Germany	2780*BC*[500]	438	3750*BC*[800]	190
Scandinavia	2550*BC*[500]	153	4500*BC*[1000]	140

An interesting intermediate step occurs between the medieval and eneolithic. The mythical Irish Chronicles relate that the O'Niall descend directly from the first Gaelic High Kings, which tradition dated c1300-1600BC. The O'Niall have the unique mutation M222 which is a branch of the haplotype L21. For L21, $n = 1029$, we compute $TMRCA = 1600BC$ and $SD = 320$. These are dates for proto Celtic, i.e. what archeologists call the pre Urnfelder Cultures, c. 1300-1600BC. Furthermore L21 is in turn a branch of haplotype P312 which we date to 2300BC. This date suggests the Bell Beaker Culture of Western Europe. Indeed the only known[24] Bell Beaker genome was found to be P312 with ^{14}C date 2300BC.

data	*Haplotype*	*n*	*TMRCA*	*[SD]*
Underhill	*R*1*a*1*a*1	974	2550*BC*	[400]
Rootsi	*G*2*a*2	536	18500*BC*	[3500]

Our method requires large data sets and many markers which means we have to rely on data from FTDNA, finding 29 useable markers out of standard 37 they use. In fact many researchers[3] have used FTDNA data. We think our method of reduction with robust statistics solves any problems with this data. To test this we compared our results with R1a1a1

Table 1. Major European SNP: Comparing Singular Reduction for 7, 15, 29 markers with KAPZ. Notice similar TMRCA for KAPZ and Singular Reduction, if there is little branching.

SNP	n	KAPZ	<SD	29 mk RSL	<SD	15 mk RSL	<SD	7 mk RSL	<SD
G2a2b	1221.	4840BC	1257	5359BC	900	8600BC	2120	4800BC	2050
R1b1a2	460.	5490BC	2144	3700BC	625	4300BC	950	5524BC	2000
R1a1a	1270.	3670BC	1066	3800BC	700	3200BC	840	3400BC	1500
I1	2898.	2400BC	1061	1800BC	400	2711BC	950	3500BC	1500
L21	1029.	3270BC	1063	1600BC	325	1700BC	400	1870BC	800
U 106	1533.	2530BC	628	2400BC	440	2500BC	600	1800BC	800
J 2	1241.	11700BC	2990	15500BC	2600	18500BC	3000	6100BC	2100
P 312	971.	2900BC	632	2240BC	420	2850BC	625	2600BC	900

data obtained from Underhill[27] with $n = 974$(which involved excluding his four M420 individuals and others with missing markers), and 15 useable markers. The result was $2550BC$ SD = 400, within the CI of our R1a1a results. Table 1 shows the results of extensive simulations using random subsets of our FTDNA data, for 29, 15 and 7 markers. For the same 15 markers as the Underhill[27] the different FTDNA data gives very similar 3300BC SD = 840 for R1a1a, verifying the correctness of using FTDNA data. However once you get down to 7 markers the confidence interval becomes large, e.g. R1a1a gives $3400BC$ SD = 1500. Also it becomes difficult to deal with outliers.

An example with few markers is the R1b1a2 data of Balaresque[3]. Our method (this time with 7 useable markers) gave SD > 30%. Now Balaresque used the Bayesian method BATWING[30] to suggest a Neolithic origin in Anatolia. With the same Cinnioglu[7] data our method gives for Turkish R1b1a2 ($n = 75$) a TMRCA = 5300BC, SD = 3100, i.e. anytime from the Ice Age to the Iron Age as seen in

R1b1a2	*n*	*TMRCA*	*[SD]*
Eire	75	$1750BC$	[1250]
England	74	$1844BC$	[1250]
Spain	207	$4600BC$	[1900]
France	62	$4300BC$	[2400]
Germany	147	$5650BC$	[2300]
Turkey	69	$5300BC$	[3100]

Fortunately, once again, we find good data from FTDNA: the Armenian DNA project, see below. By tradition the Armenians entered Anatolia from the Balkans c1000BC so they might not seem a good example of ancient Anatolian DNA. But some 100 generations of genetic diffusion has resulted in an Armenian distribution of Haplotypes J, G, R1b1a2 closely matching that of all Anatolians, therefore representive of typical Anatolian DNA. We see that Anatolian R1b1a2 arrived after c3300BC, ruling out the Neolithic expansion c6000BC. When dealing with regional haplotypes, e.g. R1b1a2 in Anatolia, the *TMRCA* is only a upper bound for the arrival times, for the genetic spread may be carried by movements of whole peoples from some other region. This means one has to be careful interpreting regional data, e.g. the TMRCA for the R1b1a2(USA) is c3700BC but nobody thinks it arrived then.

Armenian	*n*	*TMRCA*	*[SD]*
R1b1a2	99	$3300BC$	[800]
G2a2b	46	$9300BC$	[2000]
J2	97	$12100BC$	[2200]

Observe that our *TMRCA* for Armenian G2a2b (formerly G2a3) and J2 show them to be the first Neolithic farmers from Anatolia, i.e. older than $7000BC$. From Table 1 we see J2, G2a2b for all of Western Europe (non-Armenian data). Our dates show J2 was expanding at the end of the Ice Age. Modern J2 is still concentrated in the fertile crescent, but also in disconnected regions across the Mediterranean. The old genetic model predicted a continuous wave of Neolithic farmers settling Europe [8]. But you cannot have a continuous maritime settlement: it must be *leap-frog*. Also repeated resettlement from the Eastern Mediterranean has mixed ancient J2 populations, and our method gives the oldest date. On the other hand G2a2b shows exactly the dates expected from a continuous wave of Neolithic farmers across Central Europe. Our dates are consistent with recent findings that the majority of early Neolithic skeletons found in Western Europe are G2a2, c 5000BC see[33], whereas the oldest R1b1a2 found so far is Bellbeaker c2300BC, [24], [25].

Discussion

Archeology, evolutionary biology, not to mention epidemiology, forensics and genealogy are just some of the applications of molecular clocks. Unfortunately current clocks have been found to give only "ballpark" estimates. Our method is the only one giving accurate time, at least for the human y-chromosome verified over the period $500 - 15,000ybp$. There should be many applications for this y-clock, not to mention generalizations to mitochondrial and allele clocks.

Some geneticists thought natural selection makes mutation rates too variable to be useful. The problem is confusion between the actual biochemistry giving mutations and superimposed processes like kin selection producing apparently greater rates. Notice that the SD for our mutation rates is on average 14% which is much smaller than the actual previous rates. We believe this proves the reality of neutral mutation rates.

Many applications to genetics, forensics, genealogy require the TMRCA between just two individuals, or between two species, a classic method was given by Walsh[28]. While we are accurate for "big data", for this "two -body problem" one cannot determine what singular lineages the branching has been through. Just using our new asymmetric mutation rates will not work. So it would be important to find an accurate method.

Pääbo et al[24], [25] observed all 6 skeletons from Yamnaya sites, c 3300BC by ^{14}C dating, are either R1a1b1 and R1a1a. This and other work [33] involve very difficult genetic analysis of specimens which may not always be available. Also such analysis cannot date the origin of R1a1b1 and R1a1a. Our *TMRCA* shows both these haplotypes expanding at essentially the same time c3700BC. This and our later date for Anatolia, combined with Pääbo et al, implies that R1b1a2 and R1a1a must have originated in the Yamnaya Culture.

In checking accuracy we ran into the question of the origins of PIE. Although there are genes for language there is certainly none for any Indo-European language. Thus inferences have to be indirect. Marija Gimbutas saw patterns in symbolism and burial rituals suggesting the Yamnaya Culture was the cradle of Proto Indo-European. Also their physiology was robustly Europeanoid unlike the gracile skeletons of Neolithic Europe, but this could be nutrition and not genetic. From the above we conclude that the spread of this robust type into Western Europe in the late Neolithic marked an influx of Steppe nomads. Now if R1b1a2 had been shown to spread from Anatolia c6000BC it would have been taken as strong evidence for "out of Anatolia" because of the association of R1b1a2, R1a1 with Indo-European languages. But our accuracy check showed that it was G2a, J2 that spread with the Neolithic Expansion from Anatolia. Now these have been associated with Caucasian languages or Semitic, but never with Indo-European.

Materials and Methods

This work is biomathematical theory validified by data from publshed sources, see Supplementary Information SI for full mathematical development, data, algorithms and detailed MATHEMATICA worksheets. To verify the theory and compute mutation rates we use diverse data, from FTDNA y-haplotype projects for G2a2b, R1b1a2, R1a, I1, L21, U106, J2, P312. Also we used regional projects for Germany, Scandinavia, Poland and Russia for their R1b1a2, R1a1a data. The Armenian DNA project was important for its R1b1a2, J2 and G2a2b data. We also used DNA projects M222 (O'Niall), Macdonald (Group A which is R1a1a), Hamilton (group A which is I1). This was compared with non FTDNA data from Balaresque, Underhill and Rootsi.

1. D.W. Anthony, (2007)The Horse, the Wheel and Language, Princeton Univ. Press, Princeton
2. F. J. Ayala et al (2001)Erratic overdispersion of three molecular clocks: GPDH, SOD, and XDH, Proc Natl Acad Sci USA vol. 98 no. 20 pp 1140511410
3. P. Balaresque et al (2010) A predominantly Neolithic Origin for European paternal Lineages, PLoS Biol ; 8: e1000285.
4. C. Burgarella et al (2011) Mutation rate estimates for 110 Y-chromosome STRs combining population and father-son pair data, European Journal of Human Genetics 19, pp.70- 75; doi:10.1038/ejhg.2010.154
5. G. Busby et al The peopling of Europe and the cautionary tale of Y chromosome lineage R-M269, Philos Trans R Soc Lond. Biol Sci. 2012 Mar 7; 279(1730):884-92. doi: 10.1098/rspb.2011.1044
6. L.L. Cavalli-Sforza et al (2009)Y chromosome diversity, human expansion, drift, and cultural evolution, Proc Natl Acad Sci USA 106, pp. 20174-20179.
7. C. Cinnioglu et al. (2004) Excavating Y-chromosome haplotype strata in Anatolia. Hum Genet. 114, pp.127- 148.
8. C. Edmonds, A.Lillie, L.L.Cavalli-Sforza Mutations arising in the wave front of an expanding population, Proc Natl Acad Sci USA (2004); 10, pp 975-979.
9. N. Elango, J.W. Thomas , S. V. Yi (2006)Variable molecular clocks in hominoids Proc Natl Acad Sci USA vol. 103, no.5,pp13701375
10. C. Epstein, C. and R. Mazzeo(2013)Degenerate Diffusion Operators Arising from Population Biology, Princeton
11. J.N. Fenner, (2005) Cross-Cultural Estimation of the Human Generation Interval for Use in Genetics-Based Population Divergence Studies American Journal of Physical Anthropology128pp 415-428.
12. M. Gimbutas,(1956) The Prehistory of Eastern Europe, Part 1, Cambridge:American School of Prehistoric Research #20.
13. W.D. Hamilton (1964)The genetical evolution of social behaviour. I, II. Journal of Theoretical Biology 7 (1) pp 1- 52.
14. S.Y. Ho, M.J. Phillips, A. Cooper, A.J. Drummond (2005), Time dependency of molecular rate estimates and systematic overestimation of recent divergence times. Molecular Biology & Evolution 22 (7): pp 1561-1568.
15. J. Hunt, E. Bermingham, R.Ricklefs, (2001) Molecular systematics and biogeography of Antillean thrashers, tremblers, and mockingbirds Aves: Mimidae. Auk 118 (1)pp. 35 - 55 doi:10.1642/0004-8038
16. M. Lynch (2010), Rate, molecular spectrum, and consequences of human mutation, Proc Natl Acad Sci USA vol. 107 , no. 3 pp 961968
17. M. Kimura (1968) Evolutionary rate at the molecular level. Nature Vol. 217, pp 624 - 626 , doi:10.1038/217624a0
18. D.G. Kendall (1975)The Genealogy of Genealogy Branching Processes before (and after) 1873, Bulletin of the London Mathematical Society 7 (3) pp 225- 253.
19. Andrew H. Knoll et al, (2011) , Estimating the timing of early eukaryotic diversification with multigene molecular clocks, Proc Natl Acad Sci USA vol. 108, no. 33, pp 1362413629
20. P. Menozzi, A. Piazza, L.L. Cavalli-Sforza Synthetic maps of human gene frequencies in Europeans. Science(1978) 201:786- 792
21. N. Myre, N. et al A major Y-chromosome haplogroup R1b Holocene era founder effect in Central and Western Europe, European Journal of Human Genetics 19(1) pp 95101
22. Nielsen, R. (Editor)(2005) Statistical Methods in Molecular Evolution, Springer.
23. F. W. J. Olver (1964) Bessel functions of integer order pp. 355- 434 Handbook of Mathematical Functions edited by M. Abramowitz National Bureau of Standards, Washington.
24. S. Pääbo et al (2014),Ancient human genomes suggest three ancestral populations for present-day Europeans, Nature 513, pp 409-413.
25. D. Reich et al (2015) Massive migration from the steppe was a source for Indo-European languages in Europe, Nature 14317, doi:10.1038/nature1431
26. S. Rootsi S. et al (2012) Distinguishing the co-ancestries of haplogroup G Y-chromosomes in the populations of Europe and the Caucasus, Eur J Hum Genet. 20(12)pp 1275-82. doi: 10.1038/ejhg.2012.86.
27. Peter A Underhill et al, (2015)The phylogenetic and geographic structure of Y-chromosome haplogroup R1a , European Journal of Human Genetics 23, pp124-131; doi:10.1038/ejhg.2014.50
28. B. Walsh (2001) Estimating the Time to the Most Recent Common Ancestor for the Y chromosome or Mitochondrial DNA for a Pair of Individuals, Genetics 158, pp 897-912
29. C.F. Wehrhahn(1975) The evolution of selectively similar electrophoretically detectable alleles in finite natural populations, Genetics 80, pp 375- 394.
30. I. J. Wilson, M.E. Weale, D.J. Balding (2003) Inferences from DNA data: population histories, evolutionary processes and forensic match probabilities.J Roy Statist Soc A. (2003);166:1-33.
31. L.A. Zhivotovsky, L.A. (2000), Estimating Divergence Time with the Use of Microsatellite Genetic Distances, Molecular Biology and Evolution 18, No 5, pp 705-709
32. E. Zuckerkandl and L. Pauling (1965) Evolutionary divergence and convergence in proteins. In Bryson, V.and Vogel, H.J. (editors). Evolving Genes and Proteins. Academic Press, New York. pp. 97-166.
33. M. Lacana et al (2011)Ancient DNA suggests the leading role played by men in the Neolithic dissemination, Proc Natl Acad Sci USA vol. 108, no.45, pp1825518259
34. A.C. Renfrew (1987) Archaeology and Language:The Puzzle of Indo-European Origins, London:Pimlico.

ONLINE TEXT: Supplementary Information

Biomathematical theory

We emphasize the role of extraneous forces like kin-selection which operates on too big a scale and rarely enough with results that cannot be subsumed into the mutation rates. So we return to basic principles.

Fundamental Solutions: The Y-chromosome has DYS marked by $j = 1, ...N$, where one can count the STR number x_j. Consider the probability $P_{j,k}$ (at time t generations) that at marker j we have $x_j = k$. This satisfies the homogenous stochastic system

$$\frac{P_{j,k}(t)}{dt} = -\mu_j P_{j,k} + \sum_{m>0} \mu_{j,-m}\, P_{j,k-m} + \mu_{j,m}\, P_{j,k+m}$$

This homogenous system gives a uniform expansion from a single patriarch.

The system is essentially the model of Wehrhahn[29] who had $\mu_{j,-1} = \mu_{j,1}$. We introduce asymmetric mutations with total rate

$$\mu_j = \sum_{m>0} \mu_{j,-m} + \mu_{j,m}$$

About 50% of DYS markers show asymmetric mutations, i.e. $\mu_{j,-1} \neq \mu_{j,1}$.

The fundamental solution comes from the generator function

$$G(z,t) = \sum_{-\infty}^{\infty} P_{j,k} z^k \ ,$$

with complex variable z, and normalized initial condition $x_j = 0$ or $P_{j,0}(0) = 1$:

$$G(z,t) = Exp[-\mu_j t + t \sum_{m>0} \mu_{j,-m} z^m + \mu_{j,m} z^{-m}]$$

Then G can be expanded in powers of z to give $P_{j,k}(t)$. Now for the simplest asymmetric case, with only one step mutations, we have $G(z,t) = e^{-\mu_j t} e^{t\mu_{j,-1} z} e^{t\mu_{j,1}/z} =$

$$e^{-\mu_j t} \left\{ \sum_{m=0}^{\infty} \frac{\mu_{j,-1}^m}{m!} (z\, t)^m \right\} \left\{ \sum_{m=0}^{\infty} \frac{\mu_{j,-1}^m}{m!} (t/z)^m \right\}$$

so using the Hyperbolic Bessel Function of Order $k \geq 0$, see Olver [7]

$$I_k[u] = \sum_{m=0}^{\infty} \frac{u^{2m+k}}{2^{2m+k} m!(m+k)!} \ ,$$

we see that the homogenous system has fundamental solution

$$P_{j,k}(t) = e^{-\mu_j t} \left(\frac{\mu_{j,1}}{\mu_{j,-1}} \right)^{k/2} I_{|k|}[2t\sqrt{\mu_{j,-1}\, \mu_{j,1}}\,]$$

From this we obtain the second moment:

$$\sum_{k=-\infty}^{k=\infty} k^2 P_{j,k} = \{\frac{d}{dz} z \frac{d}{dz} G(z,t)\}|_{z=1} = t\mu_j + t^2(\mu_{j,1} - \mu_{j,-1})^2$$

Also from the fundamental solution we find, independently of time

$$\frac{P_{j,1}(t)}{P_{j,-1}(t)} = \frac{\mu_{j,1}}{\mu_{j,-1}} \ ,$$

which we call the *asymmetric ratio.* It will be repeatedly used.

Of course the actual initial value is not $x_j = 0$ but was usually taken to be the mode m_j which was assumed to be the value for original patriarch. Assuming symmetry, i.e. $\mu_{j,-1} = \mu_{j,1}$, the TMRCA is:

$$T = \frac{1}{n\mu} \sum_{j,i} (x_j(i) - m_j)^2 \ , \ \mu = \sum_j \mu_j .$$

From the present distribution of data we use the frequency

$$f(j,k) = \frac{\text{Count}(x_j(i) = k)}{n} \ .$$

One problem with the KAPZ formula is that higher frequencies $f(j,k), |k| = 2, 3...$ are overrepresented in the actual data. This is because the probability of a spontaneous two step mutation is much higher then the product of two one step mutations. So instead we use the frequency to solve the transcendental equation for the unknown t

$$f(j,0) \sim P_{j,0}(t) = e^{-\mu_j t} I_0[2t\sqrt{\mu_{j,-1}\, \mu_{j,1}}\,]$$

This nonlinear equation is easily solved via mathematical software such as MATHEMATICA (I used version 9 running on a boosted 2014 iMac which has accurate hyp erbolic Bessel func-tions. Earlier versions on older iMacs gave inaccuracies so one had to compile one's own functions). Using this formula resolves some other problems with the KAPZ method, e.g. $\mu_{j,-1} \neq \mu_{j,1}$ gives an extra quadratic term which if ignored causes large errors.

Heterogeneous diffusion equation : However the main problem is singularities in the stochastic process. For a uniform stochastic process, $1 - P_{j,0}(t) \sim 1 - f(j,0)$ is the probability of some mutation. So the expected variance is $f(j,0)(1 - f(j,0))$. Thus if the actual data variance $V_j >> f(j,0)(1 - f(j,0))$ we are not uniform. Now a sublineage of very high fertility increases variance, giving apparently greater *TMRCA* although it is unchanged. One finds similar results for Bayesian methods.

The correct approach to nonuniformity assumes at times t_i (generations ago) a certain proportion $0 \leq \rho_i \leq 1$ of the present population originated from a "virtual patriarch" with an initial STR value m_i. The resulting system :

$$\frac{p_{j,k}(t)}{dt} = -\mu_j p_{j,k} + \sum_{m>0} \mu_{j,-m}\, p_{j,k-m} + \mu_{j,m}\, p_{j,k+m} + d\rho$$

i.e. $d\rho$ are atoms of weight ρ_i with STR value m_i occurring at time t_i. As the system is linear and isotropic the solution is a combination of fundamental solutions P of the homogenous system. Thus the present distribution $f(j,k)$ is

$$f(j,k) = \sum_i \rho_i\ P_{j,k-m_i}(t_i)$$

This allows us to consider populations mixed by having singular lineages from overfertile patriarchs, or by actual immigration from the outside. The inverse problem seeks to find singularities from present data. Unfortunately inversion is ill posed for such systems like the heat equation . This instability produces poor accuracy. Furthermore there is no unique solution, e.g.the present distribution could have been created yesterday.

However we find that $\sim$ 50% of the DYS markers show no significant difference from the uniform expansion of a single patriarch, i.e. the data variance V_j is close to the expected variance $f(j,0)(1-f(j,0))$. The other markers show at most one significant side branch, i.e. there is an original branch starting at time $t_{j,0}$ with STR m_0 and a second one with STR $m_1 = m_0 \pm 1$ at time $t_{j,1} < t_{j,0}$ with significant $0 < \rho_1 < \rho_0$.

Reduction : We locate these singular lineages by looking for asymmetries in the distribution. For a uniform flow from a single patriarch the frequency of STR value k is given by $f(j,k) \sim P_{j,k}(t)$. The asymmetric ratio:

$$\frac{f(j,1)}{f(j,-1)} \sim \frac{P_{j,1}(t)}{P_{j,-1}(t)} = \frac{\mu_{j,1}}{\mu_{j,-1}} \ ,$$

is completely independent of time t. Therefore if say

$$\frac{f(j,1)}{f(j,-1)} >> \frac{\mu_{j,1}}{\mu_{j,-1}} \ ,$$

we have a singular lineage at $k = +1$. Thus the excess at $k = +1$ is

$$f(j,+1) - f(j,-1)\frac{\mu_{j,1}}{\mu_{j,-1}}$$

To first order approximation then frequency $f(j,+2)$ is due to this singularity at $j = +1$ which therefore gave a contribution

$$f(j,+2)\frac{\mu_{j,-1}}{\mu_{j,+1}}$$

to $k = 0$. Thus removing the effect of the singularity at $k = +1$ leads to new frequencies

$$\begin{aligned} f^*(j,-1) &= f(j,-1) \\ f^*(j,0) &= f(j,0) - f(j,+2)\tfrac{\mu_{j,-1}}{\mu_{j,+1}} \\ f^*(j,+1) &= f(j,+1) - f(j,-1)\tfrac{\mu_{j,1}}{\mu_{j,-1}} \end{aligned}$$

These of course are no longer normalized so we rescale to obtain the renormalized frequency $F(j,k)$, e.g.

$$F(j,0) = \frac{f^*(j,0)}{f^*(j,0) + f^*(j,-1) + f^*(j,+1)}$$

which will be used to compute the expansion time for marker j. There are similar formulae if the singularity was at $k = -1$.

However there is sampling error both in the frequencies and the $\mu_{j,1}, \mu_{j,-1}$. So we bootstrap taking into account these uncertainties, running the computation thousands of times. Generally we find the branch singularity is always one of $k = 0, +1, -1$ with no SD. In a few cases the singularity may seem to wander between $k = 0, +1, -1$. So in the case of a wandering singularity we obtain a distribution over $k = 0, +1, -1$ with a mean and SD. In these cases we find the singularity is relatively small and does not make much difference to the final result. However to have a stable method we do not throw out these wandering singularities but in the algorithm use the mean to average between $k = 0$ and $k = \pm 1$, e.g. if the mean is $k = 0$ then we use the original unreduced frequency.

Notice that we assume at most one side branch. In theory there could be many and solving for these produce even better approximations to the present data. In fact you could get perfect matching but find the atoms were created yesterday! The thing is that while many markers show significant deviation from a uniform flow from a single patriarch, after we have carried out reduction for one possible side branch we find no significant difference from a uniform flow, i.e. the difference is within the SD. This is of course an approximation, the next level beyond Zuckerkandl and Pauling, but given the noise in the data perhaps the best we can do. Later we further reduce the effect of outliers by using robust statistics.

Reducing the singular lineages increases the frequency $f(j,0)$ of the mode and decreases the computed $TMRCA$. But as the method of reducing singularities does not respect higher frequencies $f(j,k)$ it follows the KAPZ formula cannot be used and instead we use the probability of no mutations, i.e. solve

$$F(j,0) = e^{-\mu_j t} I_0[2t\sqrt{\mu_{j,-1}\ \mu_{j,1}}\,]$$

This is done for each DYS marker j , giving expansion times $t_1, ...t_N$ for each marker, with computed CI. (An extra fixed source of error is the uncertainty in the mutation rates which we deal with later). We find the reduction of singularities makes striking difference to the t_j of the effected markers, often a reduction of $\sim 50\%$ for $TMRCA$.

Now the existence of side branches implies that the main branch could itself have been the side branch for an earlier branch that did not survive. Thus we do not expect the expansion times $t_1, ...t_N$ for each marker to be essentially equal., i.e they are not within the SD of each other. Indeed we see that the distribution of the times t_j for different markers are almost certainly not randomly arranged about a single $TRMCA$ T but distributed from T to the present. This is seen whether you use reduction or not, or our mutation rates or not. (For a given population one could scale mutation rates to get equal t_j , but then applying these adhoc mutation rates to other populations does not yield the same values). The spread out distribution of surviving branches is another verification of our theory of many extinctions, few survivors. The distribution of the times t_j for different markers we call the branching distribution, which is now discussed.

The Branching Distribution : The times t_j for different markers are sorted from the youngest to the oldest, forming a sequence $t_1^*, ...t_N^*$. The generation of these branches is by an unknown probability distribution $d\tau_0$ over $[0, T]$. We model $d\tau_0$ by assuming a surviving lineage is generated at random with probability $\beta\Delta t$ in time period $[t, t + \Delta t]$, multiplied by the probability that the branching hasn't already occurred. The constant β averages fertility and extinction rates, the chance of a new lineage surviving. As $\beta \to \infty$ we get current theory where all lineages originate from a single patriarch at time T. Simulations with the data show that β varies in the range 1 to ∞. We make no a priori estimate of β, unlike Bayesian methods where an overall fertility rate is a predetermined parameter. Instead our stochastic simulation will find the most likely β, T in each case. Assuming independence, then the generation of branches follows the well known exponential distribution:

$$\tau_0[t] = Exp[\beta(t - T)]\ UnitStep[T - t]$$

Notice this implies a finite probability that some markers have essentially zero mutations. This is actually seen in examples. Both the Hamilton Gp A and Macdonald Gp A have number of individuals $n > 100$. For the time scale of > 700 years we do not expect there is more than one marker out of 33 which shows absolutely no mutations from the mode. In fact in both cases there are 8 markers where all n individuals have exactly the same STR value.

Estimating the parameter T for an exponential distribution is a well known problem of statistics. Kendall proved the best estimate for T would be $\max t_j$. Unfortunately there is also considerable error $\lambda_j\%$ for the mutation rates μ_j. Later we give a method for reducing this error, even so we find the SD in the range $10\% - 30\%$ which gives corresponding range in error for each t_j. We understand that the t_j are being generated by the distribution $d\tau_0$ but superimposed on this is a further uncertainty due to mutation rates etc. In particular the largest t_j may be wildly inaccurate. Also we found that simply taking the average consistently underestimates the *TMRCA* by a wide margin.

Assuming the mutation rates have normal distribution with mean μ_j and variance $\lambda_j^2\mu_j^2$, the t_j have SD $t_j\lambda_j$. Thus the actual data for t_j^* has probability density function for $s > 0$

$$d\tau(s) = \int_0^T \frac{e^{(t-T)/\beta}}{\beta}\,\frac{e^{\frac{-(t-s)^2}{2\nu}}}{\sqrt{2\pi\nu}}\,dt\ .$$

The variance ν depends on two sources. First from the uncertainty in mutation rates, for each marker we get varianceλ_j^2, giving total

$$\nu_1 = \frac{1}{N}\sum_j \lambda_j^2$$

However a small sample also has inherent error from sampling. We are measuring the probability that there is a mutation. This is binomial with probability $H_j = H_j(t) =$

$$1 - P_{j,0}(t) = 1 - e^{-\mu_j t}\left(\frac{\mu_{j,1}}{\mu_{j,-1}}\right)^{k/2} I_0[2t\sqrt{\mu_{j,-1}\,\mu_{j,1}}]$$

Hence for sample size n there is variance $H_j(1-H_j)/n$, so the variance in time due to this is scaled by the derivative giving:

$$\nu_2 = \frac{H_j(1-H_j)}{n(H_j')^2}$$

The function H_j' has actually to be computed as an inverse function depending on H_j. Therefore the total variance averaged over all N markers is $\nu = \nu_1 + \nu_2$. Although for large samples ($n > 1000$) the second term is insignificant it does effect the results once you get to $n = 100$. In our algorithm the branching distribution is used to generate large numbers of random branching times so as to bootstrap error estimates. In turns out much faster to compile the distribution function as a table which can be repeatedly called on.

Estimating TMRCA by Robust Statistics : Inaccurate large values of t_j^* are mitigated by using "robust" statistics with quintiles instead of means/variances. Using FTDNA data we began with 37 markers. However the 4 markers of DYS464 are unordered and cannot be used. Also we find that markers DYS 19/394, 385b, 459b, CDYb have errors $> 33\%$ in mutation rates so are not used. (These are some of the most popular ones in the literature!). So usually we have $N = 29$ markers and take "quintiles" $\theta^* = (t_9^*,\ t_{12}^*,\ t_{15}^*,\ t_{18}^*,\ t_{21}^*)$. This means that tail end data is not discarded but kept as the information there are 8 values of $t_j^* > t_{21}^*$, which effectively deals with outliers. Bootstrap methods give the confidence interval CI for each quintile.

Thus we wish to find the best estimate of T given θ^* (and CI). This well known statistical problem was investigated by Stochastic Simulations (SS). We also tried Maximum Likehood Methods which gave similar results but with larger CI. Monte-Carlo Methods are used to produce very large numbers ($\sim 10^7$) of T, β with corresponding Distribution. These randomly generate ordered times $(s_1 ... s_{29})$ for which we take the quintiles $\theta = (s_9,\ s_{12},\ s_{15}, s_{18},\ s_{21})$. We filter by requiring that θ close to the data θ^*, i.e. $||\theta^* - \theta|| < 2SD$. This gives a stochastic neighborhood $\mathcal{U}$ of θ^* typically containing $> 10^5$ sets of data but with T is known for each $\theta \in \mathcal{U}$. Thus we can construct a quasilinear estimator:

$$QL(s_9,\ s_{12},\ s_{15},\ s_{18},\ s_{21}) = q_1 s_9 + q_2 s_{12} + q_3 s_{15} + q_4 s_{18} + q_5 s_{21}\ ,$$

and use least squares over $\mathcal{U}$ to find constants $(q_1,\ q_2,\ q_3,\ q_4,\ q_5)$ minimizing

$$||q_1 s_9 + q_2 s_{12} + q_3 s_{15} + q_4 s_{18} + q_5 s_{21} - T||\ .$$

The $(q_1,\ q_2,\ q_3,\ q_4,\ q_5)$ are computed in MATHEMATICA . We test this by applying the QL to all of $\mathcal{U}$, unsurprisingly

$$Mean_{\mathcal{U}}[q_1 s_9 + q_2 s_{12} + q_3 s_{15} + q_4 s_{18} + q_5 s_{21} - T] \sim 0$$

What is important is that we find the uncertainty in the SS itself. Actually this depends on the data and is calculated in each case but for our examples we find

$$SD_{\mathcal{U}}[q_1 s_9 + q_2 s_{12} + q_3 s_{15} + q_4 s_{18} + q_5 s_{21} - T] \sim .05\ T$$

Finally the quasilinear estimator is applied to the experimental data

$$(t_9^*,\ t_{12}^*,\ t_{15}^*,\ t_{18}^*,\ t_{21}^*)$$

to obtain our best estimate of T. Application of QL computes the SD for our data, giving part of the overall SD. This must be combined with the SD coming from the uncertainty in the SS. Overall we find that our method has SD $\sim 12\%$, this includes variances from our data, mutation rates and uncertainty in the SS. We also tested with 15 and 7 markers. Here one must use "quintiles" $\tau = (t_5^*,\ t_8^*,\ t_{11}^*)$, $\tau = (t_3^*,\ t_5^*)$, respectively with all the loss of accuracy that implies. See Table 1 for comparisons using 29, 15, 7 markers on same data.

Accurate Mutation rates:

Any genetic clock depends on reasonably accurate mutation rates. The meiosos method looks for mutations in father-son studies. However typical rates of $\mu = .002$ would require nearly $50,000$ pairs to get an SD of 10%. Small samples have meant large errors. The phylogenetic approach studies large family groups with well developed DNA/genealogy data. So inverting the KAPZ formula would yield accurate rates. However, *singular lineages* makes this problematic. Genealogical data might give mutation rates much greater than the biochemical rates because kin selection etc tend to exaggerate the apparent mutation rate. An inspection of 10 different sources finds mutation rates claiming SD $\sim 10\%$ yet they differ from each other by up to 100%. We describe a new method.

To compute our rates we apply our theory to the large DNA projects for the SNP M222, L21, P312, U106, R1b1a2, I1, R1a1a. This avoids dealing with populations such as family DNA projects which are self selecting, i.e only those with the correct surname which neglects distant branches. Also we have very large samples, our average $n > 1000$. Greater accuracy should come from more generations and individuals. The problem is that we do not know their *TMRCA*.

Asymmetric Mutation: However before computing mutation rates we must consider asymmetric mutations, i.e. the left and right mutation rates $\mu_{j,-1} \neq \mu_{j,1}$. For a uniform stochastic process we again use the asymmetric ratio

$$\frac{p_{j,1}(t)}{p_{j,-1}(t)} = \frac{\mu_{j,1}}{\mu_{j,-1}} = \frac{A_j}{1-A_j}$$

to define the *asymmetric constant* $A_j \in [0,1]$ for marker j. For example $A_j = 0.5$ is complete symmetry. Of course singularities will effect this ratio, however these only occur $< 50\%$ of markers. Thus for each marker, SNP we compute this ratio. We find the SD for each SNP is relatively small while the difference between SNP can be large. However for each marker, using 8 SNP enables outliers to be easily removed leaving allowing us to use simple linear regression: i.e. average of the A_j over the remaining SNP groups. We see that asymmetry is a real effect: 50% of the A_j are more than two SD from symmetry $A_j = 0.5$.

Observe this is significant. The total second moment is

$$\sum_j \sum_{k=-\infty}^{k=\infty} k^2 P_{j,k} = t \sum_j \mu_j + t^2 \sum_j (\mu_{j,1} - \mu_{j,-1})^2$$

So using all our 33 DYS markers with our μ_j, we compute constants

$$\mu = \sum_j \mu_j = .12006, \tau = \sum_j (\mu_{j,1} - \mu_{j,-1})^2 = 0.000236$$

The KAPZ formula gives variance $V = \mu t$ compared to the corrected formula $\mu t + \tau t^2$. The uncorrected KAPZ gives an overestimate $> 400\%$ for > 200 generations. This effect can be nullified by using the mean instead of the mode, variance instead of the second moment, however failing to do so gives a large error. Furthermore other methods which assume symmetric mutations will also be inaccurate. Having estimates on the asymmetry is essential to our method because we find singular lineages by looking for asymmetry in the data. Any such anomaly needs to be significantly greater than the natural asymmetry.

Mutation Rates as a fixed Point: Next we compute mutation rates using 8 very large SNP groups. First, using the asymmetric constants we find singular lineages and reduce their effect. We take account of the error in the A_j by a bootstrap technique, which gives the variance for each frequency $f(j,0)$. For a given SNP k if markers j started their expansion at the same time TMRCA T_k we could calculate mutation rates μ_j via

$$(1) \qquad f(j,0) = e^{-\mu_j T_j} I_0[2T_j\sqrt{\mu_{j,-1}\ \mu_{j,1}}] \ ,$$

or rather average the 8 different μ_j we would obtain. However because of branching caused by extinction of lineages the different markers do not originate at the same time but at different times t_j. In this case we expect these t_j to be randomly distributed about the log mean over a middle set of times t_j. So, for each SNP group $k = 1,..8$ define mean time T_k, not the TMRCA but the mean log mean over a middle set of markers, which is less. We find that this is very stable. So for a fixed marker j the data $\tau_{k,j} = t_j - T_k$ should be randomly distributed about zero over the different SNP $k = 1,..,8$. However the wrong choose of μ_j would give a bias. In fact this is what we see if the mutation rates $\mu_j = .002$ were chosen. In appendix graphs show the $\tau_{k,j},\ k = 1,..8$ bunched around a nonzero point. Thus we try to find μ_j so that the $\tau_{k,j},\ k = 1,2,..8$ has mean zero. However the $\tau_{k,j},\ k = 1,2,..8$ depend nonlinearly on the rates μ_j, as does the mean $T_k, k = 1,..8$. We find this nonlinear regression problem is solved by an iterative scheme which starts with any reasonable set of DNA rates, finding any reasonable choice iterates to the same final answer. So choose $\mu_j = .002$ to begin. Suppose at some stage we have apparent mutation rates μ_j. Then, for each SNP, and each marker we solve equation (1) to obtain the apparent t_j. For each SNP $k = 1,..8$ we compute the mean log time T_k. At the next step we get new rates μ_j^* from

$$f(j,0) = e^{-\mu_j^* T_k} I_0[2T_k\sqrt{\mu_{j,-1}^*\ \mu_{j,1}^*}]$$

Averaging μ_j^* , $k = 1,..8$ we get our next set of μ_j of mutation rates. However this method would be effected by a marker showing a singular lineage. Fortunately these are few in number and by comparison between the different SNP we remove the outliers. We then repeat the process, computing T_k again with the new rates, and another set of mutation rates. So we have an iterative process.

One problem is that the iterates could tend to decrease to zero or increase to ∞, as we are only calculating relative rates. To prevent this we renormalize after each iteration so the total $\sum \mu_j$ is constant. We found the iterative scheme quickly converges to a fixed set of mutation rates, unique up to a constant factor. The CI is computed by bootstrap parametrized by the uncertainties in data and the asymmetric constants.

The generation factor γ : This method does not give absolute mutation rates but *relative* mutation rates $\mu_j\gamma$, where γ is universal time scale constant. To find γ we apply our method to compute the $T = TMRCA$ of three famous DNA projects and choose γ so the scaled T/γ best fits the historical record. We choose the DNA projects for the O'Niall(M222), Gp A of Macdonald (R1a1a) and Gp A of the Hamiltons (I1). These are large groups with characteristic DNA and fairly accurate times of origin. Of course finding one constant γ from three projects is inherently more accurate than using one project to find 33 different mutation rates. Actually assuming a generation of $27years$ these three projects yield $\gamma = 1$ with about 5% error, i.e. there is no actual need for this correction. This is a constant error (like uncalibrated ^{14}C dating).

Thus γ is related to the length of a generation. Most researchers use $25yrs$ for $t > 500ybp$ and $27yrs$ for $t < 500ybp$. Balaresque and al used $30yrs$ based on Fenner [11] who sees a $30yr$ generation for modern hunter-gatherers. Our theory allows any nominal generation as it really doesn't matter, being included in the γ factor which we compute in years not generations. However to give actual mutation rates we need an actual generation so we take 27 years. This appears in our worksheet computation. Notice that choosing a 30 year generation results in a 10% increase in the quoted mutation rate. As we find our mutation rates are close to the actual rates from meiosis this means the 27 year generation is also correct.

#	**DYS**	**Hamilton**[4]	**SD**[3]	**Burgella**[2]	**SD**[3]	**Chandler**[2]	**NIST**[2]	**FTDNA**[4]
1.	393	0.72	0.14	1.03	0.36	0.76	0.08	1.43
2.	390	2.52	0.18	2.12	0.22	3.11	2.4	5.32
3.	19/394[1]	1.3	0.52	2.19	0.21	1.51	2.38	1.45
4.	391	4.98	0.2	2.72	0.18	2.65	2.88	4.15
5.	385a	1.26	0.13				2.1	5.68
6.	385b[1]	3.13	0.34				2.1	5.68
7.	426	0.07	0.24			0.09		0.26
8.	388	0.22	0.22	0.42	2.31	0.22		0.25
9.	439	3.76	0.11	5.48	0.16	4.77		4.95
10.	389-I	1.93	0.1	2.53	0.21	1.86	1.88	2.23
11.	392	0.36	0.27	0.43	0.59	0.52	0.58	1.59
12.	389b	2.96	0.11	3.17	0.18	2.42	2.96	2.72
13.	458	7.99	0.08	6.88	0.16	8.14	10.8	6.3
14.	459a	0.39	0.18					
15.	459b[1]	2.98	0.47					
16.	455	0.16	0.21			0.16		0.46
17.	454	0.11	0.22			0.16		0.47
18.	447	3.8	0.15	4.56	0.96	2.64		4.
19.	437	0.99	0.18			0.99	1.5	2.15
20.	448	1.16	0.21			1.35	1.8	2.71
21.	449	11.7	0.14	18.97	0.52	8.38		7.84
22.	460	2.63	0.13	3.82	0.66	4.02		
23.	GATAH4	3.93	0.1	2.24	0.44	2.08	2.51	
24.	YCA IIa	0.32	0.23					
25.	YCA IIb	1.4	0.18					
26.	456	8.1	0.23	4.5	0.21	7.35		
27.	607	2.15	0.13			4.11		4.1
28.	576	10.65	0.11	16.22	0.44	10.22		10.2
29.	570	4.6	0.2	12.61	0.52	7.9		7.9
30.	CDYa	14.71	0.09					35.3
31.	CDYb[1]	13.4	2.					35.3
32.	442	2.9	0.11			3.24		
33.	438	0.43	0.14			0.55	0.7	
	Mean	3.6		5.3		3.2	2.5	6.4

Phylogenetic vs Predigree: MUTATION RATES($\times 10^{-3}$ /generation)

Notes:

1. Too inaccurate to use
2. meiosis: Burgella uses 80 sources, Chandler uses 20
3. $\times$100% one standard deviation, i.e $\times$2 for 95% CI
4. phylogenetic : Reduced Singularities(Hamilton) and not(FTDNA)

#	**DYS**	$\mathbf{A_j}$	**SD**	**SD to 0.5**
1.	393	0.675	0.087	2.
2.	390	0.463	0.093	0.4
3.	19/394	0.973	0.032	14.7
4.	391	0.029	0.008	62.8
5.	385a	0.699	0.096	2.1
6.	385b	0.82	0.085	3.8
7.	426	0.37	0.232	0.6
8.	388	0.91	0.072	5.7
9.	439	0.734	0.359	0.7
10.	389-I	0.779	0.105	2.7
11.	392	0.954	0.04	11.2
12.	389b	0.703	0.325	0.6
13.	458	0.512	0.137	0.1
14.	459a	0.139	0.125	2.9
15.	459b	0.003	0.001	353.
16.	455	0.277	0.168	1.3
17.	454	0.962	0.03	15.4
18.	447	0.154	0.025	13.6
19.	437	0.09	0.09	4.6
20.	448	0.216	0.172	1.6
21.	449	0.518	0.15	0.1
22.	460	0.107	0.05	7.9
23.	GATAH4	0.17	0.198	1.7
24.	YCAIIa	0.195	0.163	1.9
25.	YCAIIb	0.19	0.175	1.8
26.	456	0.671	0.416	0.4
27.	607	0.243	0.103	2.5
28.	576	0.387	0.157	0.7
29.	570	0.448	0.077	0.7
30.	CDYa	0.37	0.181	0.7
31.	CDYb	0.258	0.082	2.9
32.	442	0.603	0.17	0.6
33.	438	0.715	0.215	1.
Mean*	#1-33	0.26*	0.134	

Asymmetric Constants

Complete worked example for G2a3, R1b1a2, R1a1, I1, L21, U106, J2, P312.

We use   29 markers (standard method)   for G2a3, R1b1a2,R1a1, I1, L21, U106, J2, P312,
requires running compiled functions from 29ComFun and its data file W29ComFun
First we enter DNA file $\delta\delta$

```
δδ;
```

Each file has NN members

```
NN = Table[ Length[ δδ[[q1]]], {q1, 1, 8}]
```

```
{1221, 460, 1270, 2898, 1029, 1533, 1241, 971}
```

We use asymptotic rates α0, β0,LB shown

```
  0.674591   0.0869423      2.07306    0.821057      0.729026   0.39606
  0.463346   0.0925725      0.863399   0.321435     -0.146878   0.372291
  0.973243   0.0321081      36.3733    44.8473       3.59383    1.23297
  0.0290059  0.00750473     0.0298723  0.0079598    -3.51082    0.26646
  0.698661   0.0960793      2.31852    1.05808       0.840928   0.45636
  0.820451   0.085011       4.56951    2.63699       1.51941    0.577084
  0.369623   0.232259       0.586352   0.584483     -0.533836   0.996814
  0.909561   0.0716898      10.0572    8.76484       2.30828    0.871503
  0.734289   0.359214       2.76349    5.08787       1.0165     1.8411
  0.779328   0.105192       3.53162    2.16017       1.26176    0.611666
  0.954141   0.0404016      20.8057    19.2106       3.03523    0.92333
  0.702751   0.324864       2.36418    3.67672       0.860431   1.55518
  0.512368   0.1368         1.05072    0.57531       0.0494802  0.547536
  0.139428   0.125129       0.162018   0.16896      -1.82005    1.04285
  0.00301221 0.00140799     0.00302131 0.00141651   -5.80206    0.468839
{ 0.277241   0.168111   ,   0.383586   0.321817  ,  -0.958191   0.838969 }
  0.962441   0.0300025      25.6247    21.268        3.24356    0.829979
  0.153578   0.0254353      0.181444   0.0355028    -1.70681    0.195669
  0.0900312  0.0896803      0.0989388  0.108304     -2.31325    1.09465
  0.216224   0.17223        0.275874   0.280366     -1.28781    1.01628
  0.518123   0.149878       1.07522    0.645454      0.0725245  0.6003
  0.106981   0.049705       0.119797   0.0623274    -2.12196    0.520275
  0.169588   0.197965       0.204221   0.287079     -1.58855    1.40572
  0.194525   0.163447       0.241503   0.251925     -1.42087    1.04316
  0.189975   0.175366       0.23453    0.26727      -1.45017    1.1396
  0.670561   0.416154       2.03546    3.83445       0.710722   1.88382
  0.24349    0.103221       0.32186    0.18036      -1.13364    0.560367
  0.387314   0.157005       0.632157   0.418251     -0.458617   0.661624
  0.44752    0.0772017      0.81002    0.252926     -0.210696   0.312247
  0.369712   0.180559       0.586576   0.454507     -0.533454   0.774848
  0.258428   0.0818945      0.348486   0.148918     -1.05416    0.427329
  0.602829   0.170338       1.5178     1.07983       0.417265   0.711442
  0.714617   0.21496        2.50406    2.63938       0.917915   1.05404
```

```
 Table[1 , {j, 1, 33}]; BB[j_] := B[[j]]; BB0 = Table[1, {j, 1, 33}];
PP = Table[ { N[CDF[NormalDistribution[0, 1], -4 + 0.01*j]] , -4 + 0.01*j}, {j, 0, 800}];
P = Interpolation[PP];
```

We bootstrap with n02 cycles

```
n02 = 1000
```

```
500
```

The method of reduction is applied

```
ZZ = Flatten[Table[ {ClearAll[δ0, n01 , AA]; δ0 = δδ[[q1]]; n01 = Count[ Flatten[δ0] , _?Positive] / 33;
    AA = Flatten[Table[ {ClearAll[L1 , δ, m0, f1, m1, m2 , m3 , m4, m5, m6, m, f2, f3, mm ,
          δ1 , m10, m11, m12, m13, m14, m16, m17, m18, n, R, RR, β, μ, α, H, TS, TSS];
        n = IntegerPart[n01 * .5]; L1 = RandomSample[Range[n01], n]; δ = Table[ δ0[[L1[[j]] ]] , {j, 1, n}];
        m0 = Mean[δ]; m1 = Table[Commonest[ Table[ δ[[k, j]] , {k, 1, n}]][[1]], {j, 1, 33}];
        m2 = Table[ m1[[j]], {i, n}, {j, 33}]; m3 = δ - m2; m4 = Abs[m3]; f2[u_] := UnitStep[u - 1];
        m5 = f2[m4]; f3[j_] := N[ Sum[ m5[[i, j]], {i, 1, n}] / n]; m6 = Table[ f3[j] , { j, 1, 33}];
        m = Table[ Sum[m5[[i, j]] * BB0[j], {j, 33}], {i, n}] ; mm = N[ Sum[ m[[i]], {i, n}] / n];
         f1[k_] := 1 - UnitStep[ Abs[k] - 0.5] ; δ1 = Transpose[δ]; f2[ j_, k_] :=
          N[Sum[ f1[ δ1[[j, i]] - k] , {i, 1, n}] / n]; R = RandomReal[{.001, .999}, 33]; RR = P[R];
        β = Table[ Exp[LB[[j, 1]]  + LB[[j, 2]] * RR[[j]] ], {j, 1, 33}];
                        β[[j]]
        α = Table[ ————————————— , {j, 1, 33}];
                     1 + β[[j]]
        m10 = Parallelize[Table[ { Max[.001, f2[j, m1[[j]] - 2]], Max[.001, f2[j, m1[[j]] - 1]], Max[.001,
              f2[j, m1[[j]] ]], Max[.001, f2[j, m1[[j]] + 1]], Max[.001, f2[j, m1[[j]] + 2]]}, {j, 1, 33}]];
        m11 = Parallelize[Table[ { Min[.999, Max[.001, β[[j]] * m10[[j, 2]]]] ,
             Min[.999, Max[.001, m10[[j, 4]] / β[[j]]]]}, {j, 1, 33}]];
        m12 = Parallelize[Table[ { m10[[j, 4]] - m11[[j, 1]] , m10[[j, 2]] - m11[[j, 2]]}, {j, 1, 33}]];
        m13 = Parallelize[Table[ { UnitStep[m12[[j, 1]] - .001] , UnitStep[m12[[j, 2]] - .001]}, {j, 1, 33}]];
        m14 = Parallelize[Table[ m13[[j, 1]] - m13[[j, 2]], {j, 1, 33}]];
        m15 = Parallelize[
          Table[ { m10[[j, 2]] - m13[[j, 2]] * m11[[j, 2]],  m10[[j, 3]] - m13[[j, 1]] * m10[[j, 5]] / β[[j]] -
              m13[[j, 2]] * m10[[j, 1]] * β[[j]],  m10[[j, 4]] - m13[[j, 1]] * m11[[j, 1]]}, {j, 1, 33}]];
        m16 =  Parallelize[Table[ { Min[.999, Max[.001, m15[[j, 1]]]] ],  Min[1, Max[.001, m15[[j, 2]]]] ,
             Min[.999, Max[.001, m15[[j, 3]]]] }, {j, 1, 33}]]; m17 = Parallelize[
          Table[ (m16[[j, 1]] + m16[[j, 3]]) / (m16[[j, 2]] + m16[[j, 1]] + m16[[j, 3]]), {j, 1, 33}]];
        { Table[ {k, m17[[k]], 1 - m10[[k, 3]], m14[[k]] }, {k, 1, 33}]} }, {q2, 1, n02}] , 1] }, {q1, 1, 8}] ,
  1] ; MM[q_, j_] := Mean[1.0 * Table[ ZZ[[q, k, 1, j, 2]]  , {k , 1, n02}] ];
MM0[q_, j_] := Mean[1.0 * Table[ ZZ[[q, k, 1, j, 3]]  , {k , 1, n02}] ];
MM1[q_, j_] := Mean[1.0 * Table[ ZZ[[q, k, 1, j, 4]]  , {k , 1, n02}] ];
SS[q_, j_] := (Variance[1.0 * Table[ ZZ[[q, k, 1, j, 2]]  , {k , 1, n02}] ]) ^ (.5);
SS0[q_, j_] := (Variance[1.0 * Table[ ZZ[[q, k, 1, j, 3]]  , {k , 1, n02}] ]) ^ (.5);
SS1[q_, j_] := (Variance[1.0 * Table[ ZZ[[q, k, 1, j, 4]]  , {k , 1, n02}] ]) ^ (.5);
ZZ = Table[ {q, j,  MM[q, j], MM0[q, j], MM1[q, j],  SS[q, j], SS0[q, j], SS1[q, j]},  {q, 1, 8}, {j, 1, 33} ] ;
```

The output is for each file (q), marker (j), reduced frequency f0, unreduced frequency f0, mean ± then SD for each

```
MatrixForm[Transpose[ZZ]]
```

```
     1           2           3           4           5           6           7           8
     1           1           1           1           1           1           1           1
  0.240568    0.378642    0.0374759   0.0677803   0.0499991   0.0539756   0.0991811   0.0718497
  0.269364    0.40747     0.0610047   0.118237    0.0742763   0.0832689   0.113655    0.10812
   -1.         1.          0.636       0.496      -0.044      -0.698       1.         -0.428
  0.0169099   0.0262447   0.0114024   0.0185232   0.0130415   0.00951583  0.00963654  0.0149407
  0.012628    0.0236874   0.00673492  0.00584534  0.00822063  0.00717443  0.0086332   0.00993305
   0.         0.          0.743377    0.855247    0.971572    0.70127      0.         0.886786

     1           2           3           4           5           6           7           8
     2           2           2           2           2           2           2           2
  0.107158    0.218969    0.186691    0.365616    0.175806    0.416697    0.346812    0.190096
  0.166213    0.326591    0.283934    0.375769    0.257494    0.445901    0.4492      0.278878
   0.718       0.25       -0.874       1.          0.824       1.          0.978      -0.12
  0.0169463   0.0305233   0.0323346   0.0100439   0.0248054   0.0352365   0.0549321   0.0256537
  0.0111266   0.0214128   0.0125019   0.00858139  0.0133938   0.0128488   0.0145731   0.0148428
  0.686599    0.964033    0.484348     0.         0.563613     0.         0.203959    0.985669

     1           2           3           4           5           6           7           8
     3           3           3           3           3           3           3           3
  0.131797    0.263675    0.801196    0.186832    0.0994786   0.0888585   0.411807    0.180623
  0.1302      0.133226    0.546872    0.212102    0.110642    0.0876527   0.455342    0.145204
  -0.962      -0.938       -1.        -0.474      -0.306      -0.632      -0.228      -0.734
  0.054478    0.283173    0.179535    0.0906656   0.0643143   0.0656185   0.153207    0.164194
  0.0098552   0.0156641   0.0132677   0.0076426   0.00948744  0.00738005  0.0144856   0.0109657
  0.269632    0.344083     0.         0.866502    0.899991    0.754795    0.958046    0.669433

     1           2           3           4           5           6           7           8
     4           4           4           4           4           4           4           4
  0.102029    0.418723    0.570449    0.095267    0.446937    0.379255    0.229267    0.415773
  0.103105    0.369365    0.498841    0.0929137   0.433887    0.351585    0.219494    0.342025
   1.          0.896       0.8         1.          1.          1.          1.          1.
```

```
0.00984282   0.101791     0.222405     0.00585526   0.0208061    0.0272052    0.0159411    0.0739681
0.00863333   0.0215263    0.010672     0.00532949   0.0157913    0.012578     0.011802     0.0152697
0.           0.430761     0.583679     0.           0.           0.           0.           0.

1            2            3            4            5            6            7            8
5            5            5            5            5            5            5            5
0.372932     0.156555     0.129462     0.343645     0.0722556    0.0668929    0.330999     0.105297
0.3452       0.23133      0.167666     0.424041     0.129498     0.103789     0.494113     0.174994
-1.          0.988        -0.978       1.           0.172        -0.118       0.07         0.262
0.156192     0.0320991    0.016276     0.0395333    0.0211701    0.0167502    0.0671302    0.0301616
0.0137091    0.0194653    0.0100024    0.00915119   0.0103202    0.00806894   0.0142692    0.0120112
0.           0.154609     0.203959     0.           0.969689     0.976721     0.995531     0.952455

1            2            3            4            5            6            7            8
6            6            6            6            6            6            6            6
0.303214     0.310065     0.195828     0.230686     0.228629     0.191401     0.881276     0.246809
0.451843     0.413896     0.255685     0.313384     0.334957     0.252157     0.697429     0.320685
0.89         -0.24        -0.694       -0.346       -0.108       -0.812       -0.932       -0.704
0.0775789    0.109212     0.0376006    0.0586784    0.0790634    0.0379511    0.1483       0.0524204
0.014807     0.0234281    0.0127067    0.0084827    0.0149095    0.0109063    0.0112583    0.0151201
0.440782     0.965538     0.716513     0.933824     0.985026     0.577341     0.357255     0.708087

1            2            3            4            5            6            7            8
7            7            7            7            7            7            7            7
0.0049278    0.138349     0.0139767    0.0035918    0.0171963    0.0126407    0.00848146   0.00717218
0.00581639   0.176626     0.0174929    0.00435611   0.0221946    0.0156606    0.011029     0.0121938
-0.372       0.286        0.306        -0.458       0.616        0.624        0.608        0.768
0.00161293   0.0258992    0.00357312   0.00103036   0.00490497   0.00326455   0.00254619   0.00252049
0.00202028   0.0177634    0.00353908   0.00127954   0.00480104   0.00323704   0.00293991   0.00350903
0.69179      0.951849     0.913254     0.555746     0.746766     0.740081     0.712286     0.553886

1            2            3            4            5            6            7            8
8            8            8            8            8            8            8            8
0.382711     0.0414717    0.121485     0.0523867    0.0113426    0.0117869    0.473713     0.0171688
0.377426     0.0454261    0.0683654    0.103204     0.0223035    0.0224752    0.368171     0.0349732
-1.          -0.888       -0.084       -0.536       0.59         0.324        -0.788       0.76
0.016235     0.0120193    0.279873     0.0144323    0.00471267   0.0046887    0.23988      0.00675708
0.0137072    0.00982407   0.00687747   0.00531027   0.00451239   0.00385478   0.0131675    0.00595632
0.           0.442547     0.82359      0.821007     0.492326     0.68991      0.609754     0.427511

1            2            3            4            5            6            7            8
9            9            9            9            9            9            9            9
0.353073     0.355654     0.304021     0.18749      0.439431     0.387134     0.523378     0.406941
0.376085     0.36793      0.312794     0.224167     0.415685     0.403997     0.54249      0.397353
0.448        -0.344       0.966        0.112        -0.578       -0.704       0.474        -0.59
0.187036     0.133936     0.143399     0.084994     0.184245     0.1141       0.188248     0.164548
0.0133885    0.0221195    0.0126467    0.00737327   0.0153443    0.0128111    0.0137912    0.0159068
0.892687     0.937775     0.211976     0.990666     0.815626     0.710911     0.87799      0.806971

1            2            3            4            5            6            7            8
10           10           10           10           10           10           10           10
0.0713628    0.170909     0.153777     0.077468     0.117344     0.118668     0.397436     0.226436
0.118957     0.207687     0.232696     0.101213     0.171879     0.154371     0.441635     0.270825
0.858        -0.876       0.698        0.99         0.254        -0.47        -0.978       -0.812
0.0211065    0.0276348    0.0465147    0.0146727    0.0376701    0.0299822    0.0307026    0.0299138
0.00911389   0.0188094    0.0121851    0.0058851    0.0118822    0.00946882   0.0138265    0.0140673
0.492261     0.482794     0.70127      0.118016     0.950416     0.873284     0.203959     0.580802

1            2            3            4            5            6            7            8
11           11           11           11           11           11           11           11
0.0328995    0.195605     0.019464     0.0276099    0.0913281    0.0545896    0.0428634    0.0458454
0.035023     0.226243     0.0274583    0.0344444    0.086249     0.0604282    0.0473323    0.0549237
-1.          -0.556       -0.19        -0.404       -0.622       -0.896       -0.682       -0.744
0.00559746   0.0547398    0.00671988   0.0102913    0.105592     0.0146441    0.0109744    0.0114476
0.00518393   0.0206381    0.00482019   0.0033436    0.008516     0.00593477   0.00584676   0.00723343
0.           0.822324     0.760959     0.811234     0.753825     0.435388     0.688348     0.631872

1            2            3            4            5            6            7            8
12           12           12           12           12           12           12           12
0.339109     0.319741     0.361585     0.14143      0.229618     0.201784     0.383215     0.172139
0.350852     0.361217     0.395468     0.181101     0.279381     0.243068     0.435403     0.210062
-0.44        -0.49        -0.532       0.468        0.588        -0.404       0.536        0.338
0.135645     0.0802929    0.0814271    0.0359787    0.0625503    0.037088     0.145978     0.0647567
0.0137165    0.0236567    0.0139621    0.00675153   0.0135278    0.0112598    0.0137106    0.0136602
0.898897     0.871446     0.847592     0.875505     0.807192     0.913484     0.837919     0.938892

1            2            3            4            5            6            7            8
13           13           13           13           13           13           13           13
0.476577     0.54595      0.360548     0.274225     0.365821     0.393367     0.707439     0.364524
0.57322      0.608843     0.465093     0.382658     0.485257     0.515504     0.718939     0.488829
0.9          0.784        0.928        0.75         0.288        0.186        -0.344       0.164
0.132784     0.119895     0.0780383    0.0472547    0.0524781    0.0591396    0.172361     0.0548065
0.0138668    0.0217439    0.0135952    0.00890805   0.0153576    0.0126115    0.0104027    0.0164223
0.431709     0.621383     0.367541     0.657544     0.952297     0.980473     0.935636     0.983381
```

```
     1            2            3            4            5            6            7            8
     14           14           14           14           14           14           14           14
 0.0624034    0.0534526    0.0383731    0.0595369    0.0322549    0.0202481    0.466316     0.0461582
 0.0778262    0.0680087    0.0481165    0.0677764    0.0441012    0.0310078    0.472319     0.0533072
 -0.992       0.7          0.428        0.896        0.89         0.862        -0.924       0.94
 0.0127965    0.0134504    0.0102178    0.00946347   0.00697174   0.00443081   0.0169745    0.00810478
 0.00769154   0.0118631    0.00582004   0.00491314   0.00655514   0.00447344   0.014574     0.00703451
 0.0891734    0.703555     0.882254     0.439967     0.449783     0.472655     0.382775     0.329571
     1            2            3            4            5            6            7            8
     15           15           15           15           15           15           15           15
 0.136723     0.386403     0.33634      0.0644196    0.277988     0.273151     0.136782     0.379884
 0.0411148    0.240574     0.193011     0.0325549    0.184961     0.158217     0.0761097    0.219023
 1.           0.85         1.           1.           0.668        0.894        1.           0.998
 0.20748      0.174364     0.150186     0.105402     0.144691     0.147113     0.0986715    0.158311
 0.00567544   0.0202167    0.0115791    0.00337602   0.0125126    0.00933531   0.00797001   0.0128227
 0.           0.41454      0.           0.           0.500276     0.308146     0.           0.0447214
     1            2            3            4            5            6            7            8
     16           16           16           16           16           16           16           16
 0.0193586    0.0188599    0.0193956    0.0096757    0.0250929    0.0184955    0.0257319    0.0169415
 0.0328492    0.0246696    0.0261701    0.0107578    0.031751     0.0240627    0.0453355    0.0203546
 0.48         -0.412       0.728        1.           0.756        0.636        -0.072       0.96
 0.00505397   0.00701672   0.00452255   0.00193965   0.00587055   0.00404637   0.00762892   0.00449533
 0.00525437   0.00727703   0.00441726   0.00197075   0.005484     0.00376786   0.00585702   0.00457037
 0.831253     0.878521     0.653353     0.           0.633456     0.743377     0.957418     0.273037
     1            2            3            4            5            6            7            8
     17           17           17           17           17           17           17           17
 0.0423687    0.0127185    0.0179726    0.0119004    0.00997297   0.0125568    0.0908602    0.0235482
 0.0564328    0.0129913    0.0187937    0.01249      0.0098716    0.0130574    0.129777     0.0245732
 0.144        -0.454       -0.38        -0.716       0.05         -0.634       -0.65        -0.724
 0.0170089    0.0052182    0.00472831   0.00235444   0.00322644   0.00309282   0.025281     0.00650297
 0.00663093   0.00530869   0.00393579   0.00196422   0.00315979   0.00302826   0.00993743   0.0050316
 0.841832     0.603832     0.603596     0.525258     0.218163     0.529722     0.729771     0.593742
     1            2            3            4            5            6            7            8
     18           18           18           18           18           18           18           18
 0.254561     0.293814     0.527293     0.320815     0.283906     0.483114     0.92366      0.317199
 0.277134     0.312704     0.536683     0.352878     0.303572     0.519436     0.587242     0.345402
 1.           1.           0.666        1.           1.           0.996        1.           1.
 0.0144934    0.028504     0.148493     0.0136537    0.017521     0.0245071    0.10748      0.0205768
 0.0132915    0.0206442    0.0138653    0.00947081   0.0138524    0.0130838    0.014034     0.0145364
 0.           0.           0.729045     0.           0.           0.0894427    0.           0.
     1            2            3            4            5            6            7            8
     19           19           19           19           19           19           19           19
 0.0841961    0.135851     0.0075069    0.0528791    0.102671     0.128198     0.458605     0.231307
 0.110885     0.142939     0.00783307   0.063234     0.111521     0.140689     0.463623     0.283225
 0.448        0.88         0.97         0.608        0.816        0.746        0.892        0.064
 0.0219754    0.0319458    0.00244726   0.0123995    0.0192299    0.0256947    0.0802071    0.0688537
 0.00848921   0.0158963    0.00251892   0.00435065   0.0102981    0.00847201   0.014728     0.0146715
 0.879114     0.471216     0.170758     0.774297     0.571662     0.662088     0.452488     0.988868
     1            2            3            4            5            6            7            8
     20           20           20           20           20           20           20           20
 0.301911     0.120197     0.119487     0.0981758    0.127037     0.0587458    0.730929     0.236121
 0.326203     0.1368       0.144346     0.119086     0.152926     0.0727415    0.639139     0.304416
 0.634        0.922        0.572        0.814        0.786        0.766        0.816        -0.106
 0.0911821    0.0214009    0.023147     0.0149159    0.0251037    0.0102312    0.176829     0.0596804
 0.0139184    0.0167504    0.0101576    0.00608641   0.0110577    0.00662191   0.012606     0.0151568
 0.772812     0.384984     0.816179     0.579721     0.613969     0.63249      0.578631     0.990316
     1            2            3            4            5            6            7            8
     21           21           21           21           21           21           21           21
 0.610493     0.60805      0.548683     0.524637     0.603721     0.48144      0.796038     0.429884
 0.703859     0.666983     0.624526     0.588661     0.632148     0.581554     0.743868     0.532091
 0.016        0.168        -0.434       0.984        -0.1         0.602        0.542        0.74
 0.19081      0.130523     0.111978     0.127548     0.139197     0.0969687    0.177159     0.106083
 0.0126124    0.0217684    0.0143029    0.00980425   0.0106437    0.0130562    0.0104979    0.017454
 0.994848     0.984741     0.898476     0.178347     0.995984     0.795526     0.840028     0.667301
     1            2            3            4            5            6            7            8
     22           22           22           22           22           22           22           22
 0.553821     0.282852     0.254349     0.272732     0.19985      0.239778     0.656476     0.251641
 0.397652     0.307348     0.265795     0.260934     0.228767     0.269483     0.468084     0.306132
 1.           0.978        1.           1.           0.772        0.794        1.           0.508
 0.142994     0.0290677    0.0162436    0.0172761    0.0423308    0.045151     0.153543     0.0750222
 0.014292     0.0213008    0.012798     0.00859559   0.0127702    0.0111487    0.0144687    0.0147008
 0.           0.203959     0.           0.           0.623533     0.596889     0.           0.855219
     1            2            3            4            5            6            7            8
     23           23           23           23           23           23           23           23
```

```
  0.328663     0.484439     0.613505     0.136682     0.262803     0.334175     0.517864     0.289208
  0.388374     0.413835     0.447087     0.151159     0.288868     0.367995     0.420974     0.345324
  -0.16        0.908        0.978        0.558        0.72         -0.136       0.976        0.064
  0.0788466    0.201053     0.234179     0.0614367    0.0465338    0.130912     0.211129     0.0651582
  0.014047     0.0233485    0.0139081    0.00663136   0.0143292    0.0126484    0.0134849    0.0142894
  0.984041     0.41939      0.203959     0.829465     0.691778     0.989678     0.217989     0.992913

  1            2            3            4            5            6            7            8
  24           24           24           24           24           24           24           24
  0.219455     0.0369204    0.0152541    0.0638566    0.0171383    0.0248498    0.0720283    0.0295387
  0.235295     0.069513     0.0378992    0.0598634    0.0415019    0.0486919    0.0996452    0.0468041
  -1.          0.244        0.678        0.752        0.79         0.578        0.87         0.986
  0.0155442    0.018623     0.00452604   0.172969     0.00447098   0.0424478    0.136715     0.00621069
  0.0120083    0.012282     0.00516938   0.00426184   0.00603961   0.00531804   0.00839916   0.00696065
  0.           0.945649     0.686483     0.635061     0.57842      0.780471     0.487417     0.160797

  1            2            3            4            5            6            7            8
  25           25           25           25           25           25           25           25
  0.131858     0.0628114    0.10563      0.0300541    0.168405     0.10801      0.218778     0.0688693
  0.124787     0.111183     0.239975     0.0543092    0.24521      0.159008     0.357068     0.11854
  1.           -0.214       0.44         0.636        0.964        0.48         0.694        0.178
  0.0736288    0.0214409    0.0490273    0.00599343   0.0265341    0.0206309    0.14743      0.0198778
  0.00950662   0.0144692    0.0117806    0.00425068   0.0141066    0.0088233    0.0127892    0.0102737
  0.           0.958144     0.894427     0.754083     0.266169     0.871274     0.719304     0.971702

  1            2            3            4            5            6            7            8
  26           26           26           26           26           26           26           26
  0.277956     0.566391     0.666728     0.258531     0.604548     0.634763     0.692447     0.551759
  0.297197     0.545261     0.550838     0.288937     0.566128     0.607274     0.513839     0.514961
  -0.016       0.604        -0.472       0.278        0.326        -0.386       -0.03        -0.724
  0.159479     0.225086     0.215994     0.120512     0.217965     0.162283     0.303584     0.178355
  0.0125865    0.0218017    0.0143382    0.00863337   0.0106792    0.0117671    0.013866     0.015124
  0.994848     0.795267     0.882481     0.958412     0.943135     0.922337     0.999549     0.690491

  1            2            3            4            5            6            7            8
  27           27           27           27           27           27           27           27
  0.286284     0.608988     0.28787      0.162825     0.1891       0.154815     0.368795     0.164774
  0.294561     0.482243     0.388139     0.18405      0.230424     0.207496     0.437671     0.208697
  1.           1.           0.062        0.998        0.842        0.478        0.808        0.778
  0.063518     0.159259     0.0677377    0.0162894    0.0290734    0.0333481    0.0664925    0.025518
  0.013628     0.0255695    0.0131939    0.00704146   0.0128253    0.010046     0.0137427    0.0126028
  0.           0.           0.994048     0.0447214    0.530659     0.871226     0.589773     0.620876

  1            2            3            4            5            6            7            8
  28           28           28           28           28           28           28           28
  0.83791      0.614269     0.546978     0.498689     0.514985     0.589598     0.797234     0.59548
  0.706593     0.636461     0.611723     0.490482     0.555416     0.562514     0.715077     0.601464
  0.556        0.118        0.074        0.984        0.626        0.974        0.276        0.386
  0.217444     0.166543     0.107976     0.153556     0.132832     0.167211     0.191571     0.169624
  0.0132721    0.0208401    0.0139947    0.00914276   0.0158623    0.0125803    0.0104357    0.0160432
  0.829603     0.990979     0.995241     0.178347     0.776744     0.222312     0.960035     0.922337

  1            2            3            4            5            6            7            8
  29           29           29           29           29           29           29           29
  0.438918     0.355845     0.469487     0.747729     0.29653      0.390903     0.589925     0.382068
  0.5892       0.454304     0.608214     0.705874     0.42423      0.475603     0.692116     0.486054
  -0.812       0.994        0.068        0.796        0.69         0.996        0.46         0.844
  0.0571279    0.0491678    0.055929     0.139441     0.0449352    0.0538673    0.0800382    0.0506122
  0.0140239    0.023837     0.0146114    0.00807017   0.0159087    0.012604     0.012702     0.0156915
  0.584242     0.0999198    0.994663     0.605903     0.717586     0.0894427    0.882018     0.529362

  1            2            3            4            5            6            7            8
  30           30           30           30           30           30           30           30
  0.697219     0.815209     0.701099     0.585507     0.743338     0.613878     0.882958     0.746186
  0.699226     0.714165     0.624233     0.591463     0.659735     0.664131     0.766516     0.663184
  0.25         0.498        0.916        0.592        0.878        -0.114       0.65         0.684
  0.169178     0.166093     0.206957     0.159319     0.196214     0.157615     0.151472     0.184962
  0.0120361    0.0189024    0.0140891    0.00926066   0.0148317    0.0106129    0.00964158   0.0159221
  0.968181     0.86689      0.396558     0.804257     0.477044     0.991448     0.759377     0.730213

  1            2            3            4            5            6            7            8
  31           31           31           31           31           31           31           31
  0.980267     0.820739     0.978582     0.858941     0.886215     0.846092     0.874355     0.807889
  0.762521     0.695287     0.729701     0.696047     0.68023      0.666995     0.760023     0.677703
  0.988        0.8          0.996        0.968        0.948        0.98         0.942        0.89
  0.0571064    0.180458     0.0618665    0.134941     0.144913     0.140732     0.131328     0.170598
  0.00922038   0.0167097    0.0122011    0.00900586   0.0143748    0.0118126    0.0121757    0.0152389
  0.154609     0.597255     0.0894427    0.251202     0.318589     0.199197     0.332953     0.454216

  1            2            3            4            5            6            7            8
  32           32           32           32           32           32           32           32
  0.152614     0.19472      0.699889     0.169919     0.171173     0.240949     0.452412     0.179728
  0.201269     0.257591     0.654076     0.231429     0.234556     0.331399     0.544745     0.246346
  0.922        -0.672       0.988        -0.35        -0.286       -0.088       0.918        -0.422
  0.922        -0.672       0.988       -0.33        -0.286       -0.088       0.918        -0.422
  0.0288794    0.039516     0.205955    0.0248729    0.0263694    0.0351946    0.0860289    0.0277084
  0.0123298    0.0209075    0.0133831   0.00799638   0.013087     0.0116574    0.0144251    0.0137949
  0.384984     0.738584     0.154609    0.93233      0.958144     0.99309      0.394445     0.904185

  1            2            3           4            5            6            7            8
  33           33           33          33           33           33           33           33
  0.0510223    0.063852     0.0412005   0.0246973    0.030431     0.0648421    0.0535717    0.0329089
  0.0710164    0.0863391    0.059537    0.032697     0.0424008    0.0777728    0.0801452    0.0482557
  0.8          -0.304       -0.078      -0.464       -0.354       -0.962       0.712        -0.272
  0.0145717    0.0165081    0.0105283   0.00485418   0.00755017   0.0116778    0.0409857    0.00864755
  0.0073346    0.0130492    0.00648817  0.00331646   0.00622569   0.00626831   0.00763957   0.007195
  0.590506     0.945183     0.980734    0.873057     0.917806     0.254333     0.685563     0.935825
```

We conservatively weight the reduction by mean ±

```
ZZZ = Table[ {q1, k, Abs[ZZ[[q1, k, 5]]] * ZZ[[q1, k, 3]] + (1 - Abs[ZZ[[q1, k, 5]]]) * ZZ[[q1, k, 4]] },
   {q1, 1, 8} , {k, 1, 33}];
```

```
MatrixForm[Transpose[ZZZ]]
```

```
  1            2            3            4            5            6            7            8
  1            1            1            1            1            1            1            1
  0.240568     0.378642     0.0460404    0.0932107    0.0732081    0.0628222    0.0991811    0.0925961

  1            2            3            4            5            6            7            8
  2            2            2            2            2            2            2            2
  0.123812     0.299686     0.198944     0.365616     0.190183     0.416697     0.349065     0.268225

  1            2            3            4            5            6            7            8
  3            3            3            3            3            3            3            3
  0.131736     0.255587     0.801196     0.200124     0.107226     0.0884148    0.445416     0.171202

  1            2            3            4            5            6            7            8
  4            4            4            4            4            4            4            4
  0.102029     0.413589     0.556128     0.095267     0.446937     0.379255     0.229267     0.415773

  1            2            3            4            5            6            7            8
  5            5            5            5            5            5            5            5
  0.372932     0.157453     0.130303     0.343645     0.119652     0.0994348    0.482695     0.156733

  1            2            3            4            5            6            7            8
  6            6            6            6            6            6            6            6
  0.319564     0.388976     0.214144     0.284771     0.323474     0.202823     0.868774     0.268676

  1            2            3            4            5            6            7            8
  7            7            7            7            7            7            7            7
  0.00548584   0.165679     0.0164169    0.00400606   0.0191156    0.0137762    0.00948011   0.0083372

  1            2            3            4            5            6            7            8
  8            8            8            8            8            8            8            8
  0.382711     0.0419145    0.0728274    0.0759657    0.0158366    0.0190122    0.451338     0.0214419

  1            2            3            4            5            6            7            8
  9            9            9            9            9            9            9            9
  0.365776     0.363707     0.304319     0.220059     0.42941      0.392126     0.533431     0.40301

  1            2            3            4            5            6            7            8
  10           10           10           10           10           10           10           10
  0.0781212    0.17547      0.177611     0.0777054    0.158027     0.13759      0.398409     0.234781

  1            2            3            4            5            6            7            8
  11           11           11           11           11           11           11           11
  0.0328995    0.209208     0.0259394    0.0316833    0.0894082    0.0551968    0.0442845    0.0481694

  1            2            3            4            5            6            7            8
  12           12           12           12           12           12           12           12
  0.345685     0.340894     0.377442     0.162535     0.250121     0.226389     0.40743      0.197244

  1            2            3            4            5            6            7            8
  13           13           13           13           13           13           13           13
  0.486242     0.559535     0.368076     0.301333     0.450859     0.492786     0.714983     0.468443

  1            2            3            4            5            6            7            8
  14           14           14           14           14           14           14           14
  0.0625268    0.0578194    0.0439464    0.0603938    0.033558     0.0217329    0.466772     0.0465872

  1            2            3            4            5            6            7            8
  15           15           15           15           15           15           15           15
  0.136723     0.364529     0.33634      0.0644196    0.247103     0.260968     0.136782     0.379563

  1            2            3            4            5            6            7            8
  16           16           16           16           16           16           16           16
  0.0263737    0.022276     0.0212382    0.0096757    0.0267175    0.020522     0.043924     0.017078

  1            2            3            4            5            6            7            8
  17           17           17           17           17           17           17           17
  0.0544076    0.0128674    0.0184817    0.0120678    0.00987666   0.01274      0.104481     0.0238311

  1            2            3            4            5            6            7            8
  18           18           18           18           18           18           18           18
  0.254561     0.293814     0.53043      0.320815     0.283906     0.483259     0.92366      0.317199
```

```
    1          2          3           4          5          6          7          8
   19         19         19          19         19         19         19         19
0.0989285  0.136702  0.00751669  0.0569382  0.104299   0.131371   0.459147   0.279902

    1          2          3           4          5          6          7          8
   20         20         20          20         20         20         20         20
0.310802   0.121492  0.130127    0.102065   0.132577   0.0620208  0.714039   0.297177

    1          2          3           4          5          6          7          8
   21         21         21          21         21         21         21         21
0.702365   0.657082  0.59161     0.525662   0.629305   0.521285   0.772144   0.456457

    1          2          3           4          5          6          7          8
   22         22         22          22         22         22         22         22
0.553821   0.283391  0.254349    0.272732   0.206443   0.245897   0.656476   0.278451

    1          2          3           4          5          6          7          8
   23         23         23          23         23         23         23         23
0.37882    0.477944  0.609844    0.143081   0.270101   0.363395   0.515539   0.341732

    1          2          3           4          5          6          7          8
   24         24         24          24         24         24         24         24
0.219455   0.0615604 0.0225459   0.0628663  0.0222547  0.0349112  0.0756185  0.0297804

    1          2          3           4          5          6          7          8
   25         25         25          25         25         25         25         25
0.131858   0.100831  0.180863    0.0388829  0.17117    0.134529   0.261095   0.109699

    1          2          3           4          5          6          7          8
   26         26         26          26         26         26         26         26
0.296889   0.558024  0.605538    0.280484   0.578653   0.617885   0.519197   0.541603

    1          2          3           4          5          6          7          8
   27         27         27          27         27         27         27         27
0.286284   0.608988  0.381922    0.162868   0.195629   0.182315   0.382019   0.174525

    1          2          3           4          5          6          7          8
   28         28         28          28         28         28         28         28
0.779605   0.633842  0.606932    0.498558   0.530106   0.588894   0.737753   0.599154

    1          2          3           4          5          6          7          8
   29         29         29          29         29         29         29         29
0.467171   0.356435  0.598781    0.73919    0.336117   0.391241   0.645108   0.39829

    1          2          3           4          5          6          7          8
   30         30         30          30         30         30         30         30
0.698725   0.764485  0.694642    0.587937   0.733139   0.658402   0.842203   0.719957

    1          2          3           4          5          6          7          8
   31         31         31          31         31         31         31         31
0.977654   0.795649  0.977587    0.853728   0.875504   0.84251    0.867723   0.793569

    1          2          3           4          5          6          7          8
   32         32         32          32         32         32         32         32
0.156409   0.215342  0.699339    0.2099     0.216429   0.32344    0.459983   0.218233

    1          2          3           4          5          6          7          8
   33         33         33          33         33         33         33         33
0.0550211  0.079503  0.0581068   0.0289851  0.0381635  0.0653334  0.0612248  0.0440814
```

This has SD

```
ZZS = Table[ (((Abs[ZZ[[q1, k, 5]]] * (ZZ[[q1, k, 6]]^2 + ZZ[[q1, k, 3]]^2) +
        (1 - Abs[ZZ[[q1, k, 5]]]) * (ZZ[[q1, k, 7]]^2 + ZZ[[q1, k, 4]]^2) -
        (Abs[ZZ[[q1, k, 5]]] * ZZ[[q1, k, 3]] + (1 - Abs[ZZ[[q1, k, 5]]]) * ZZ[[q1, k, 4]] ) ^
         2)) ^ (.5)), {q1, 1, 8}, {k, 1, 33} ]
```

```
MatrixForm[Transpose[ZZS]]
```

```
( 0.0169099   0.0262447   0.0150785   0.0287026   0.00984279  0.0161131   0.00963654  0.0217723  )
  0.0307772   0.0524265   0.044439    0.0100439   0.0388115   0.0352365   0.0564037   0.0332447
  0.0534683   0.27608     0.179535    0.0639244   0.0368056   0.0523607   0.0764675   0.141652
  0.00984282  0.0977706   0.201033    0.00585526  0.0208061   0.0272052   0.0159411   0.0739681
  0.156192    0.0329973   0.017108    0.0395333   0.0251382   0.0152384   0.0472955   0.0358338
  0.0868517   0.0724301   0.0423258   0.0527821   0.0443006   0.0418971   0.150493    0.0560309
  0.00192754  0.0267641   0.00390191  0.00123232  0.00543883  0.00356787  0.00297937  0.00349696
  0.016235    0.0118596   0.0827048   0.0276943   0.00710732  0.00649531  0.21735     0.0100516
  0.126103    0.080783    0.140969    0.0314825   0.140918    0.0962967   0.130339    0.126889
  0.0258845   0.029322    0.053553    0.0148008   0.0320828   0.0280632   0.0311151   0.0326282
  0.00559746  0.0456823   0.00610206  0.00779095  0.0834782   0.0141062   0.00986613  0.0112513
  0.0907476   0.0622441   0.0624854   0.0319673   0.0545517   0.0322746   0.110393    0.0431591
  0.129337    0.109735    0.0799693   0.0624436   0.062339    0.0551232   0.10159     0.0532593
  0.0128375   0.0146067   0.00934349  0.00943822  0.00785646  0.00578414  0.0168792   0.00822179
  0.20748     0.16916     0.150186    0.105402    0.126318    0.143559    0.0986715   0.158316
  0.00848783  0.00772002  0.00541158  0.00193965  0.00644745  0.00477032  0.00785509  0.00454781
  0.0101823   0.00526955  0.004273    0.00226616  0.00316323  0.00307881  0.0281879   0.00614934
  0.0144934   0.028504    0.121529    0.0136537   0.017521    0.0245791   0.10748     0.0205768
  0.0207913   0.0305564   0.00245007  0.0112451   0.0182489   0.0232447   0.0759226   0.0258139
  0.0740204   0.0215403   0.0224035   0.0159434   0.0251841   0.0112051   0.163736    0.0320145
  0.0296024   0.0611702   0.0834911   0.126785    0.0459594   0.0901655   0.13318     0.102062
  0.142994    0.0291417   0.0162436   0.0172761   0.0395943   0.0422917   0.153543    0.0608903
  0.0404923   0.192795    0.232881    0.0466613   0.0418753   0.0510235   0.209117    0.025524
  0.0155442   0.0198649   0.0115951   0.15002     0.0110421   0.0345259   0.127893    0.00654414
  0.0736288   0.0256221   0.0747159   0.0128693   0.0298429   0.0298989   0.138549    0.0227615
  0.0238464   0.175773    0.159612    0.0653973   0.126052    0.102126    0.0622873   0.152855
  0.063518    0.159259    0.0321319   0.0163037   0.0310627   0.0357315   0.0659135   0.0295817
  0.174997    0.0608871   0.0364877   0.15233     0.107341    0.165092    0.107502    0.106173
  0.0783244   0.0496406   0.0403901   0.125599    0.0704257   0.0540305   0.0750201   0.0602003
  0.0852334   0.128335    0.199261    0.12276     0.185953    0.0564526   0.134278    0.158017
  0.0615236   0.169192    0.0637144   0.135835    0.148358    0.141565    0.130267    0.166094
  0.0308394   0.045431    0.204782    0.0334488   0.0337864   0.0298252   0.0863302   0.0389421
( 0.0156393   0.01756     0.00846399  0.00572231  0.00883082  0.0117811   0.0368462   0.0102306  )
```

To computes times for each marker need mutation rates

```
MatrixForm[ μ00]
```

```
⎛ 0.000722271  0.137882 ⎞
   0.00252192  0.175606
   0.00130357  0.516471
   0.00497715  0.199051
   0.00126812   0.12551
   0.00312581    0.3394
 0.0000647366  0.244154
   0.00021529  0.215784
   0.00376011  0.113175
   0.00193695  0.102042
  0.000362231  0.273074
   0.00295976  0.110029
   0.00799224 0.0848847
  0.000398105   0.17669
   0.00297971  0.468587
  0.000163263  0.205268
  0.000114375   0.21543
   0.00380365  0.151532
  0.000995294  0.179893
   0.00116395  0.214196
    0.0117063  0.139137
   0.00263369  0.133619
   0.00393157 0.0996985
   0.00032059  0.233166
   0.00140393   0.17576
   0.00810337  0.228217
   0.00214539  0.130452
    0.0106568   0.11133
   0.00460282  0.204858
      0.01471 0.0948324
    0.0134021   2.00472
    0.0029028  0.112088
⎝ 0.000433081  0.141076 ⎠
```

Also need hyperbolic Bessel functions

```
nn = 60; τ = .01;
```

$$\mathbf{I0}[t\_] := \mathbf{N}\left[1 + \sum_{k=1}^{nn}\left(\frac{(t/2)\wedge(2*k)}{k!*(k!)}\right), nn\right]; \mathbf{I1}[t\_] := \mathbf{N}\left[\frac{t}{2} + \sum_{k=1}^{nn}\left(\frac{(t/2)\wedge(2*k+1)}{k!*((k+1)!)}\right), nn\right];$$

$$\mathbf{I2}[t\_] := \mathbf{N}\left[\frac{t\wedge 2}{8} + \sum_{k=1}^{nn}\left(\frac{(t/2)\wedge(2*k+2)}{k!*((k+2)!)}\right), nn\right]; \mathbf{EE}[t\_] := \mathbf{N}\left[1 + \sum_{k=1}^{nn}\left(\frac{t\wedge k}{k!}\right), nn\right];$$

```
SGN[x_] := UnitStep[x] + (-1) * (1 - UnitStep[x]); μ0 = Table[ μ00[[j, 1]] , {j, 1, 33}];
α = Table[ α0[[j, 1]] , {j, 1, 33}];
μ = Table[ {μ0[[j]] * α[[j]], μ0[[j]] * (1 - α[[j]])}, {j, 1, 33}]; H[t_, j_] :=
```

$$\mathbf{EE}[-t*\mu 0[[j]]]*\Bigg(\mathbf{I0}\Big[2\,t*\sqrt{\mu[[j,1]]*\mu[[j,2]]}\Big] + \left(\sqrt{\frac{\mu[[j,1]]}{\mu[[j,2]]}}\right)*\mathbf{I1}\Big[2*\tau*t*\sqrt{\mu[[j,1]]*\mu[[j,2]]}\Big]*$$
$$\left(\frac{\mu[[j,2]]}{\mu[[j,1]]}\right)*\mathbf{I2}\Big[2*(1-\tau)*t*\sqrt{\mu[[j,1]]*\mu[[j,2]]}\Big]$$
$$+\left(\sqrt{\frac{\mu[[j,2]]}{\mu[[j,1]]}}\right)*\mathbf{I1}\Big[2\,t*\tau*\sqrt{\mu[[j,1]]*\mu[[j,2]]}\Big]*\left(\frac{\mu[[j,1]]}{\mu[[j,2]]}\right)*$$
$$\mathbf{I2}\Big[2*(1-\tau)*t*\sqrt{\mu[[j,1]]*\mu[[j,2]]}\Big]\Bigg);$$

The times T for each marker are obtained by

```
TS[k_] := Normal[InverseSeries[Series[ (1 - H[x, k]), {x, 0, 60}]]]; TSS[t_, k_] := TS[k] /. x → t ;
```

which is applied to give expansion times for each marker

```
ZZZ01 = Parallelize[ Table[ TSS[ ZZZ[[ q1, k , 3]] , k] ,  {q1, 1, 8}, {k, 1, 33} ] ];

MatrixForm[ Transpose[ZZZ01]]
```

```
⎛407.203  745.888  65.9476  138.51   107.078  91.1482  148.092  137.529⎞
 54.2553  156.492  93.4188  206.936   88.553  253.043  193.422  135.232
 108.765  228.193  1295.65  172.306  87.2671  71.1848  459.416  144.761
 21.6883  108.901   167.1   20.1721  121.055  97.1282  52.7115  109.674
 413.047  140.358  113.532  367.827  103.345  84.4952  619.416  139.631
  131.05  170.949  80.0411  113.094  133.147  75.1141  1023.86  105.184
 85.0834  2927.15  256.695  62.0648  299.495   214.98  147.468   129.58
 2337.26  199.593   353.44  369.393  74.2456  89.3013  2940.57  100.859
 134.181  133.109  104.465    69.65  170.101  148.332  245.394  154.463
 42.5993  103.153  104.595  42.3597  91.5971  78.4726  290.096  145.157
 92.4878  654.785  72.6387  89.0081  259.634  157.138  125.293  136.585
 158.752  155.705   179.89  62.3289  103.904  91.9382  201.529  77.9842
  104.23   138.46  66.0026  49.7747  91.1088  106.861  289.141  97.4104
 163.463  150.689  113.503  157.665  86.0952  55.3393  1719.21   120.53
 49.3623  152.367  137.763  22.3517  95.3347  101.584   49.385  160.425
 164.596  138.614  132.058  59.6696  166.783  127.539  277.648  105.875
 490.118  113.285   163.21  106.199  86.8137  112.156  968.708  211.067
 80.4307  96.0044  222.838  107.391  91.9683  191.456  1102.76  105.832
 105.572  149.514  7.58545  59.1864  111.687  143.177  651.989  339.302
 342.875  113.841  122.743  94.2453  125.291   55.619  1452.75   323.54
 183.088  144.329  107.693  82.7402  126.769  81.3532  294.972  63.5303
  334.13  130.825  114.751  124.835  89.8265  110.214  456.773  128.035
 130.467  183.747  281.819  40.1682  83.9784  123.225   207.79  113.447
 805.373  200.201   71.387  204.634  70.4513  111.467  248.366  94.7549
  103.01  76.9857  146.754  28.4232  137.827  105.308  226.595  84.2999
 47.4786  129.094  154.545  44.0827  139.443  162.204  111.949  121.505
 168.361  551.528  248.401  85.7678  105.897  97.5775  248.491  92.7994
 304.143  137.153  122.494   80.592  90.7147  113.876  230.738  118.672
 167.831  109.133  280.642  575.497  100.333  125.428  343.952  128.937
 132.225  192.023  129.565  81.4689  158.809  109.295  352.956  147.646
 1301.11  198.221  1298.72  281.979  329.967  261.634  311.339  196.013
 61.1945  89.0373   690.95  86.3282  89.5821  150.128  256.813  90.4892
⎝132.219   194.63  139.957  68.3293   90.571  158.222  147.813  105.072⎠
```

The SD for T are

```
ZZZ001 = Parallelize[ Table[ TSS[ (ZZZ[[ q1, k , 3]] + ZZS[[ q1, k]]), k] ,  {q1, 1, 8}, {k, 1, 33} ] ];
ZZZ002 = Parallelize[ Table[ TSS[ (ZZZ[[ q1, k , 3]] - ZZS[[ q1, k]]), k] ,  {q1, 1, 8}, {k, 1, 33} ] ];
ZZZ02 = 0.5 * (ZZZ001 - ZZZ002)
```

```
MatrixForm[Transpose[ZZZ02]]
```

```
35.3745  76.0796 22.3553 45.8682 15.2219 24.5178 15.5416 34.7482
14.9582  36.9794 24.9631 8.39574 21.4063 34.5354 45.3361 21.6812
47.6569  304.723 1227.52  62.177 31.8336 44.3298 109.938 133.837
2.21588  34.9097 103.677 1.30772 7.82788 9.05847 4.21865 26.3997
263.798  33.4154 16.5164 59.0017 23.8446 13.9762  105.69 36.2485
46.8203  45.2437 18.6277 26.4156 23.9038 18.0879 2560.86  27.214
30.0166  543.801 61.7586 19.1484 86.4349 56.2488 46.6714 54.6867
 133.15  57.9098 421.035 141.114 33.6333  30.853 2212.34  47.885
 67.509  42.2603 65.4358 11.9637 90.9453 54.5133 119.857 75.1234
14.9263  19.7269 36.2246 8.52698 20.9671 17.7369 32.9826 24.3854
16.0256  163.068 17.3345 22.2754 256.008 41.4276  28.614 32.7767
 58.926  39.6193 43.6599 13.9852 28.2798 15.9117 85.4523  20.136
55.8616  66.9808 21.6297 14.0044 21.9645 22.7935 243.517 19.9594
34.9462  39.5156 24.8191 25.6189 20.5899 14.9312  94.774 21.9146
82.3561  91.8126 77.4888 37.9868 56.9458 66.1593 38.5657 87.8415
53.9809  48.8078 34.1615 12.0437 41.0241 30.0823  51.257 28.5378
 94.535  46.7174 38.1147 20.0731 27.9527 27.2913 277.531 55.1744
5.55259  11.7277 89.7761 5.90899 7.07462  15.345 2301.62 8.84551
23.5948  36.4711 2.48338  12.098 20.8539 27.5382 158.992 38.1436
107.317  22.0632 23.2613 15.8465 26.2526 10.4944 1238.65 44.8597
31.7232  45.7595 40.2954 45.1135 26.9981 29.7441 618.985 25.7489
153.104  16.5379  8.7791 9.62489 19.8634 22.5746 240.952 34.3546
19.3797  127.581 278.668 14.5173 16.1092 23.6461  159.39 11.2717
67.5829  67.4005 37.2723 517.156 35.4806 112.933 447.353 21.2421
63.4768  21.0046 69.6877 9.65704  27.288 25.7963  151.07 18.9054
5.03345  102.392 119.015 13.3773  73.854 71.6164 25.8455 77.6566
 48.036  362.647 30.0507 9.73911 19.6574 22.0927 62.0412 18.0373
  2617.  37.5859 18.5573 51.4442 39.3137 97.1415 200.008 56.7127
51.0714  22.2327 48.2617 800.298  29.996 26.8321 130.217 30.6184
64.4397  259.952  285.63 44.6256 463.411 28.7709 5540.92 219.195
7642.67  429.009 8521.82 864.266 3696.23 719.411 1181.31 381.004
13.7756  22.8054 1966.69  16.584 16.9775 19.1291 85.4561  19.651
39.1352  45.6275 21.2765  13.774 21.5456 29.9444 93.1496 25.1813
```

We are only using 29 markers

```
B = Table[ j , {j, 1, 33}]; UU[x_] := 1 - UnitStep[-x];
B = UU[{1, 2, 0, 4, 5, 0, 7, 8, 9, 10, 11, 12, 13, 14, 0, 16, 17,
   18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 0, 32, 33}]; BB[j_] := B[[j]];
```

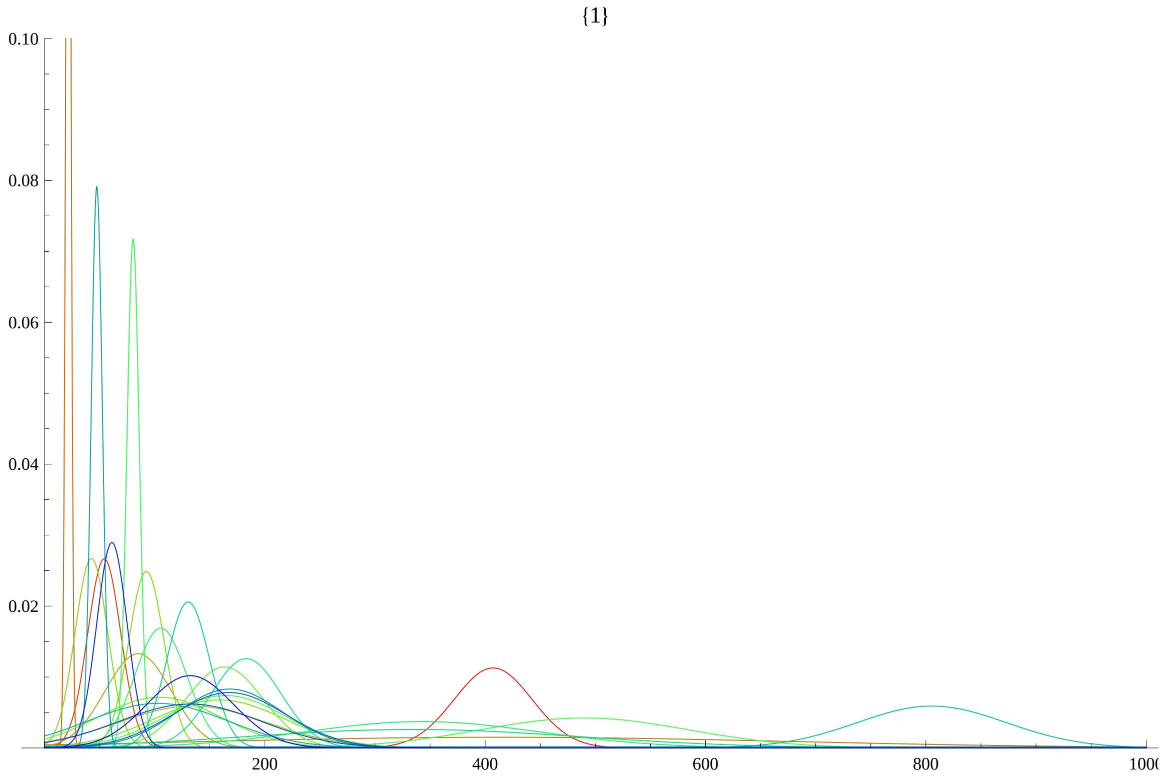
We first sketch the spectrum of times

$$\texttt{QQ[x\_, q3\_, q4\_] := N}\left[\frac{1}{q4\sqrt{2*\pi}}*\texttt{Exp}\left[-\frac{(x-q3)\,\hat{}\,2}{2*(q4)\,\hat{}\,2}\right]\right];$$

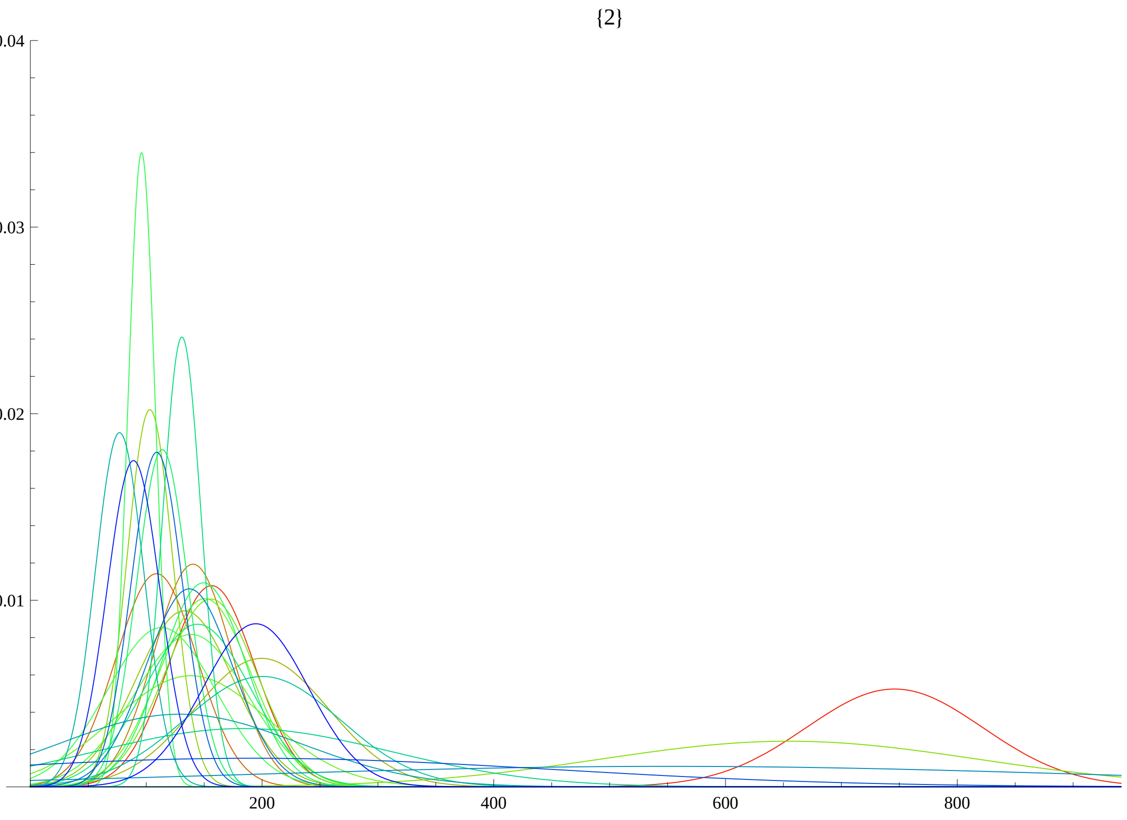
```
Q[x_, q1_, j_] := BB[j] * QQ[x, ZZZ01[[ q1, j]] + .01, ZZZ02[[ q1, j]] + .01]; Pic1[q_, a_, b_] :=
 Plot[Evaluate[Table[Q[ x, q, j], {j, 1, 33}]], {x, 0, a}, PlotRange → {0, b}, PlotStyle → Table[
    RGBColor[UnitStep[ 22 - j] * (22 - j) / 22, Sin[ j * π / 33], UnitStep[ j - 11] * (j - 11) / 22 ], {j, 1, 33}],
  PlotLabel →
   {q}]
```

This plots the time distributions for each marker

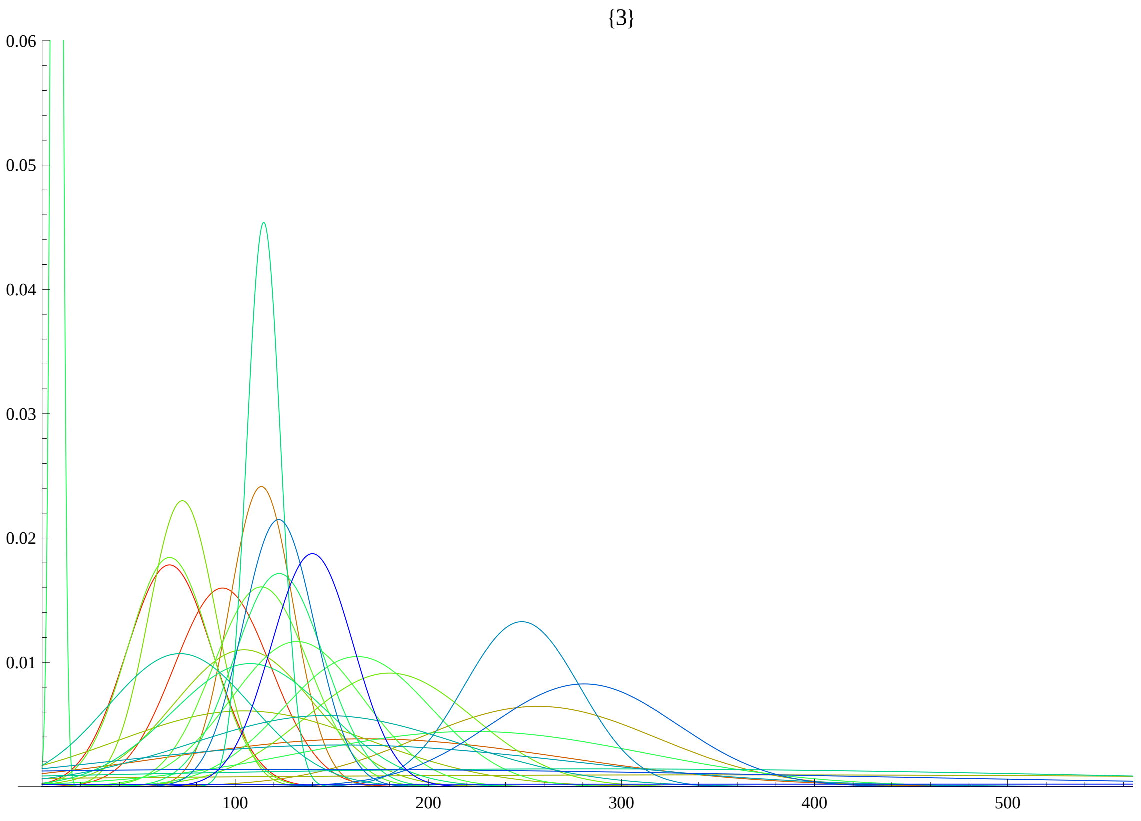
```
Pic1[1, 1000, 0.1]
```



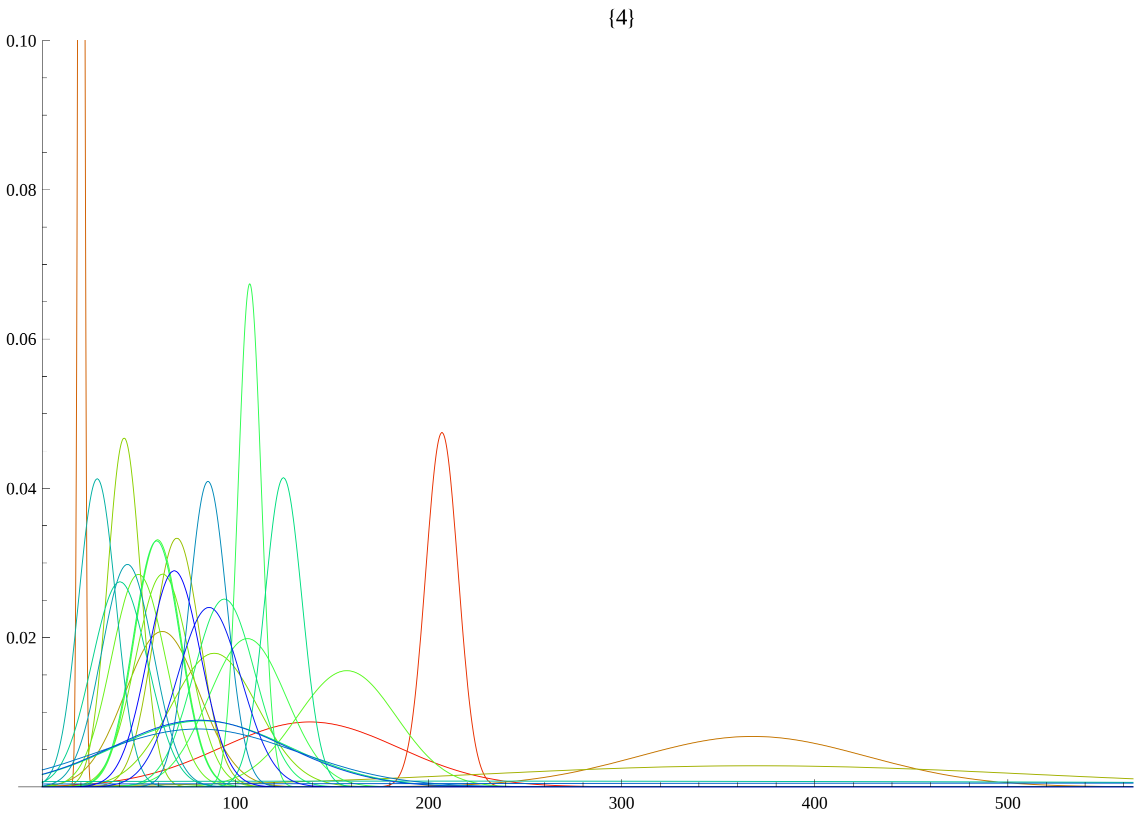
```
Pic1[2, 1000, 0.04]
```



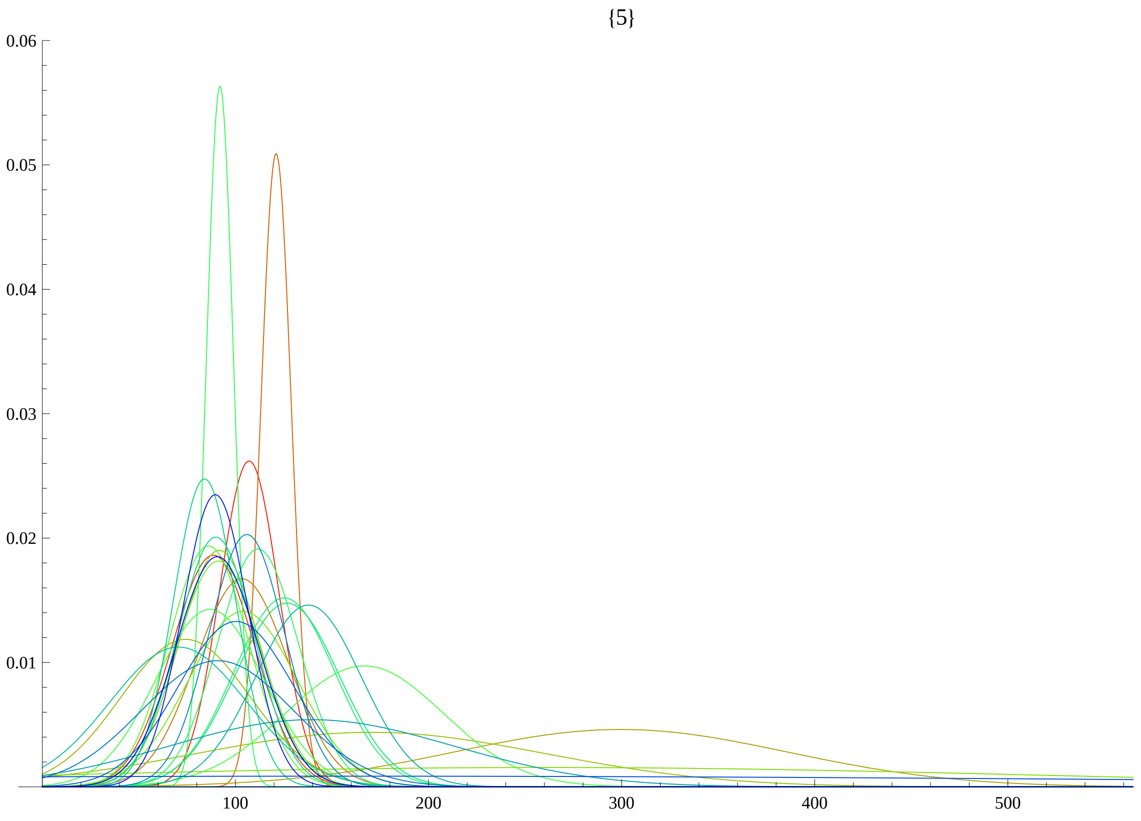
```
Pic1[3, 600, 0.06]
```



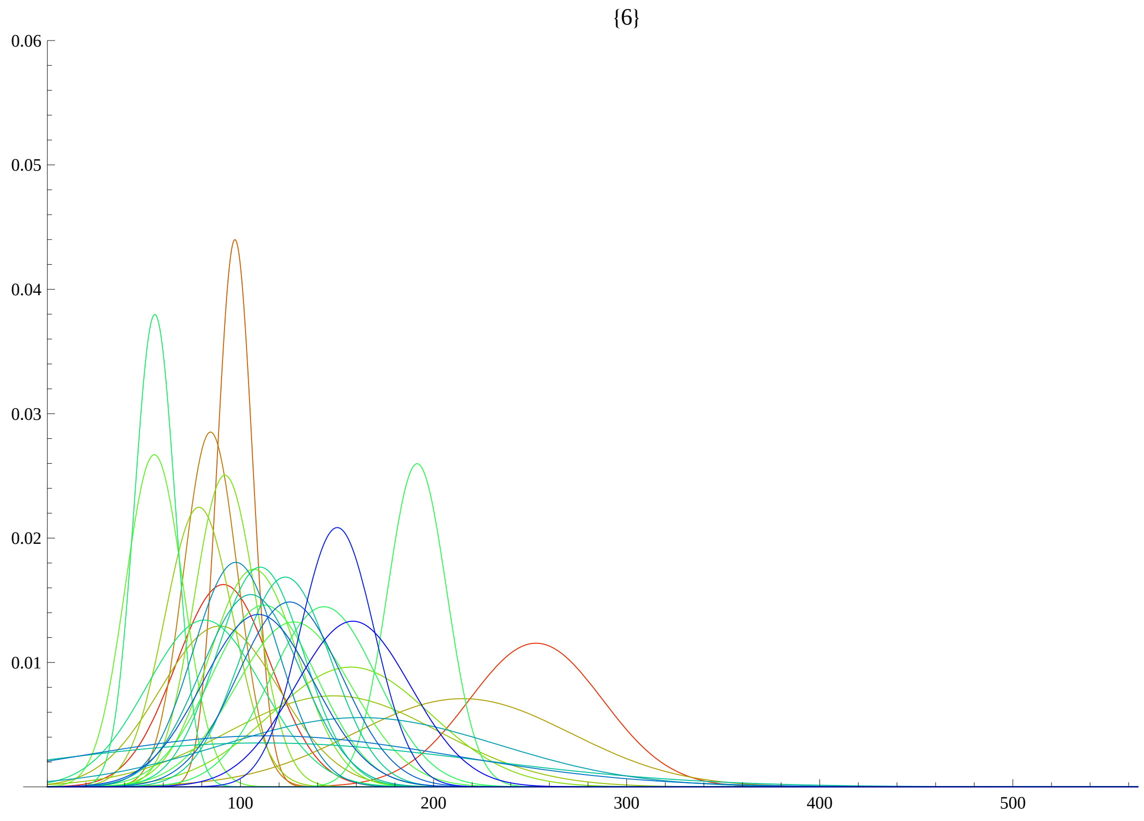
```
Pic1[4, 600, 0.1]
```



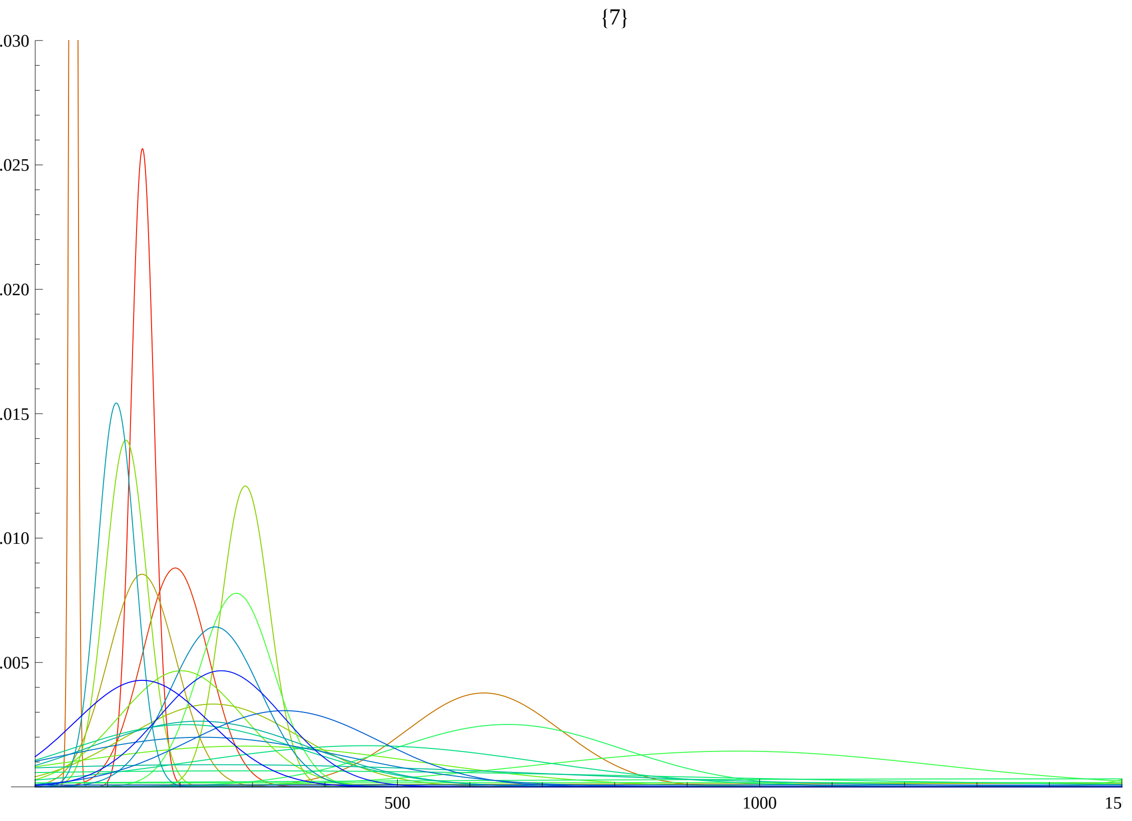
```
Pic1[5, 600, 0.06]
```



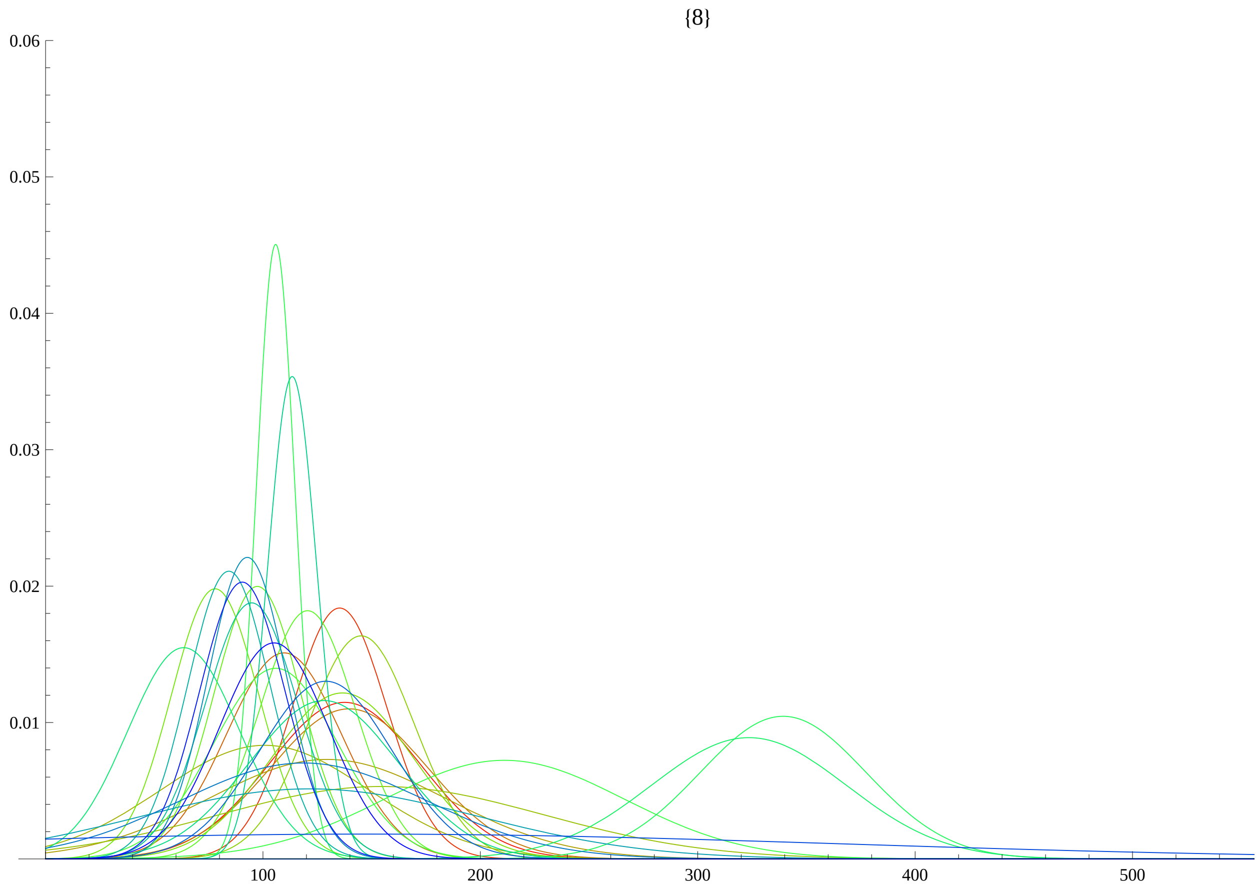
```
Pic1[6, 600, 0.06]
```



```
Pic1[7, 1600, 0.03]
```



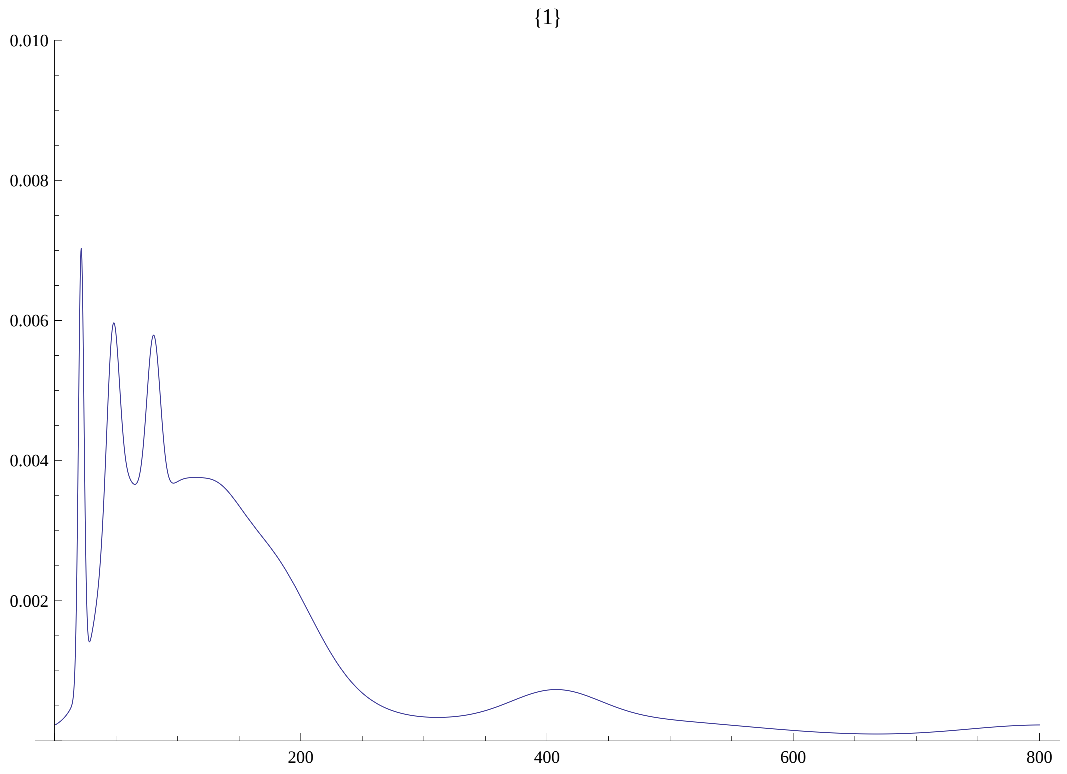
```
Pic1[8, 600, 0.06]
```



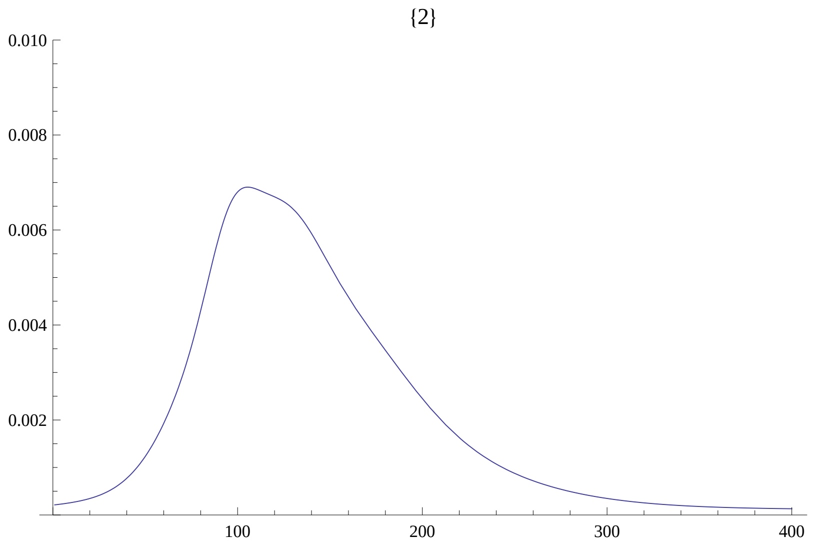
The individual branches may be combined to obtain average times

```
Q0[x_, q_] := Sum[Q[x, q, j] / 29, {j, 1, 33}];
Pic2[q_, a_, b_] := Plot[ Q0[x, q] , {x, 1, a}, PlotRange → {0, b},
  PlotLabel → {q}]
```

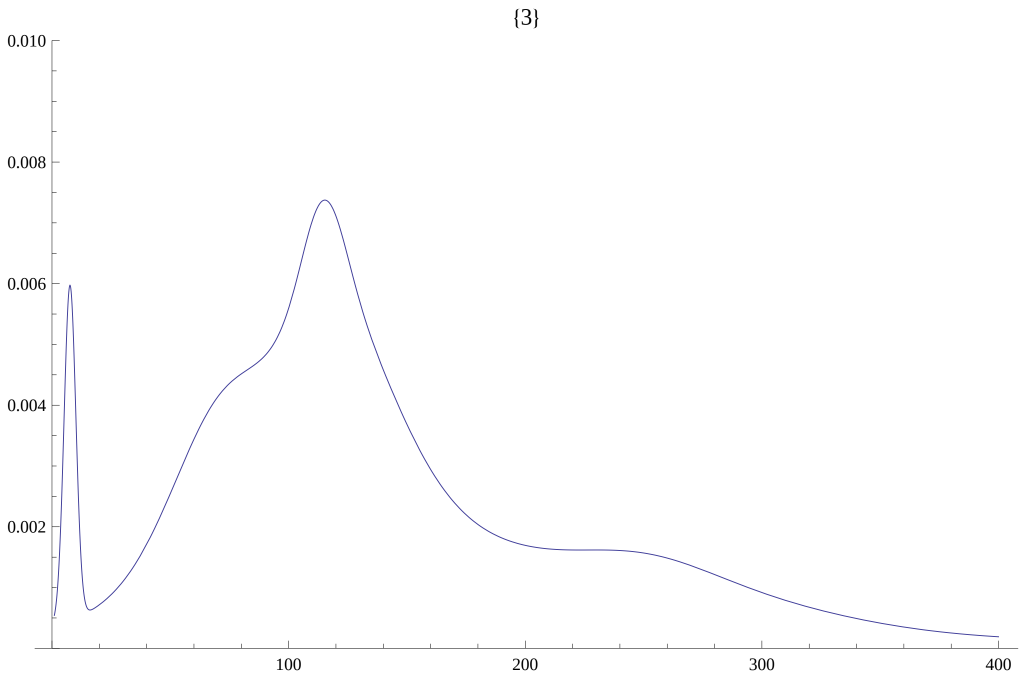
```
Pic2[1, 800, 0.01]
```



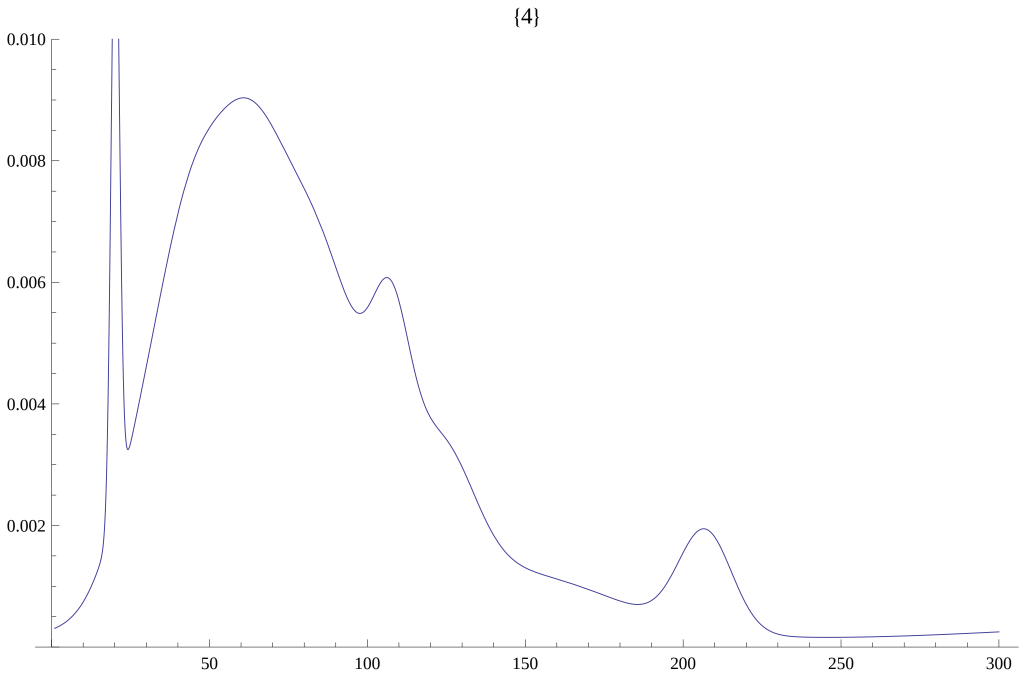
```
Pic2[2, 400, 0.01]
```



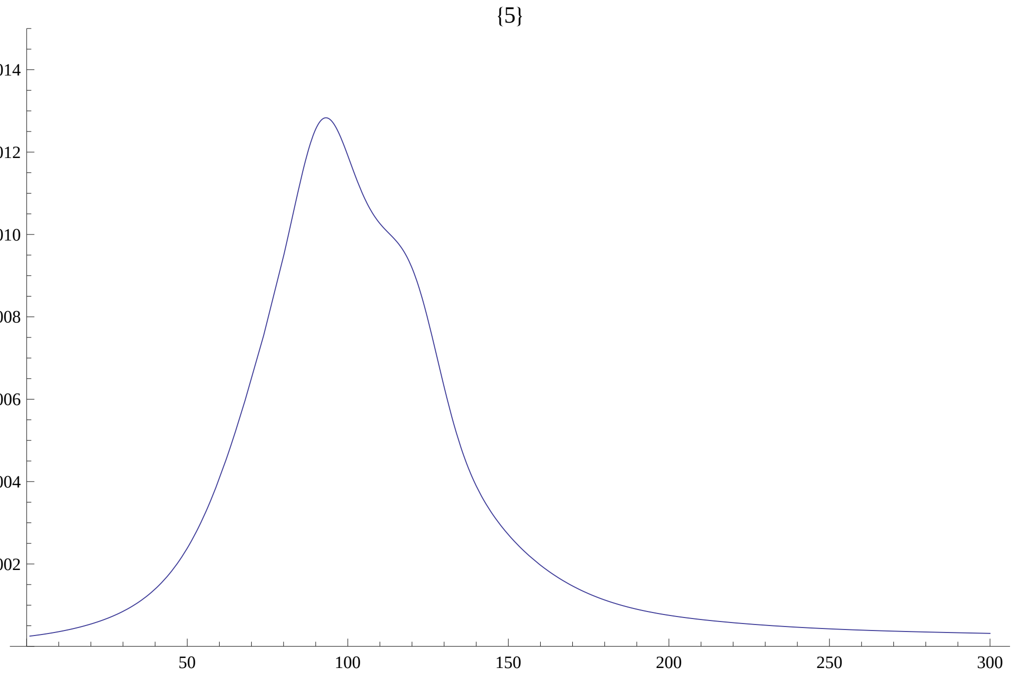
```
Pic2[3, 400, .01]
```



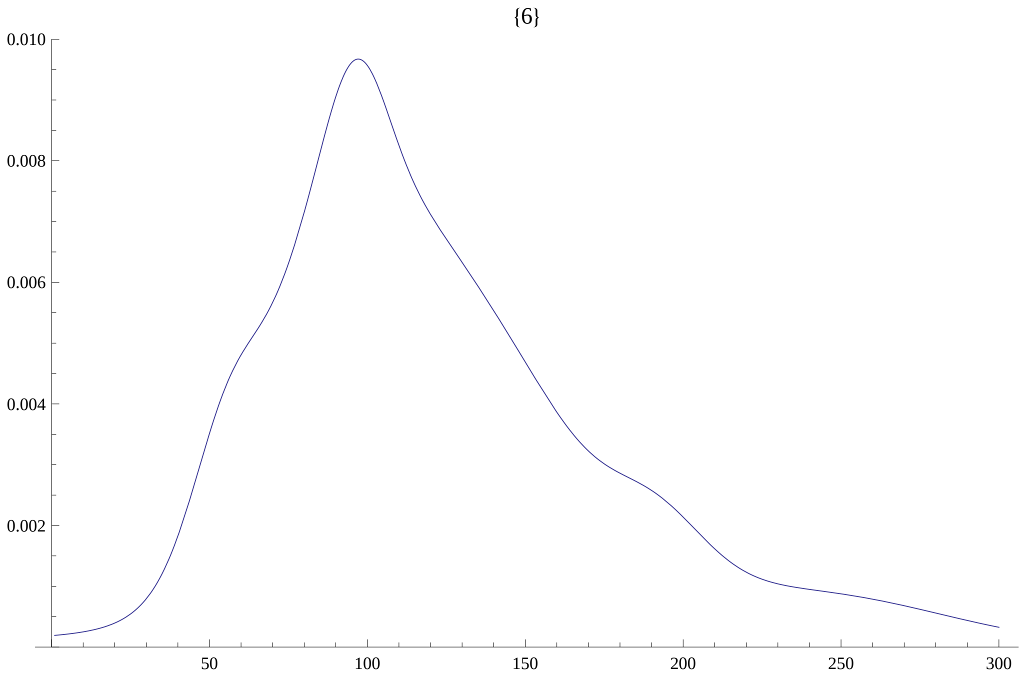
```
Pic2[4, 300, .01]
```



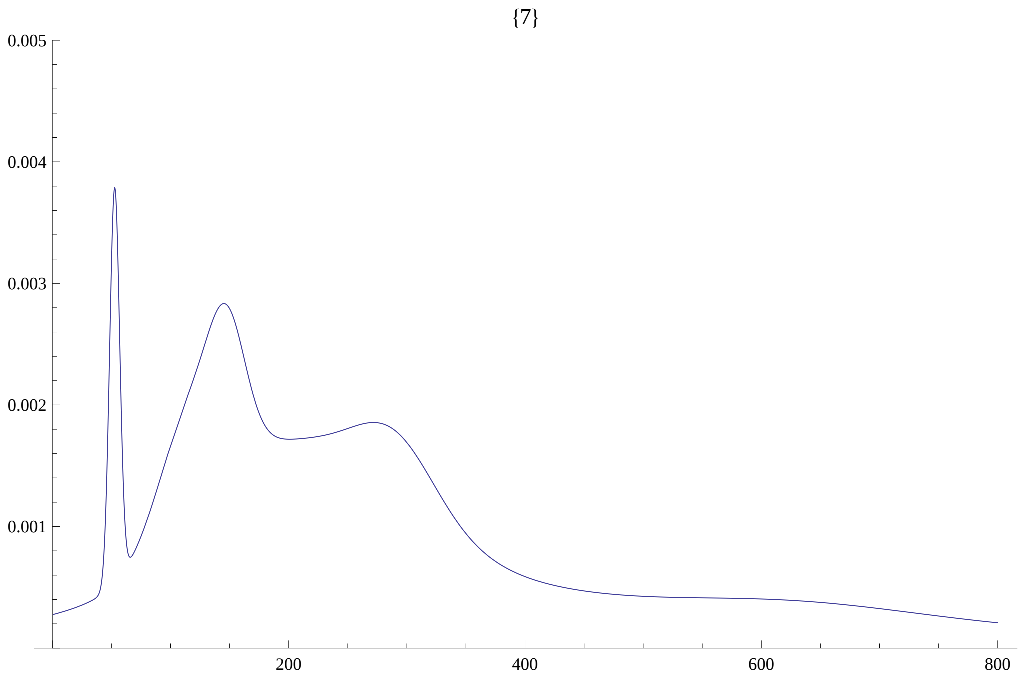
```
Pic2[5, 300, .015]
```



```
Pic2[6, 300, .01]
```

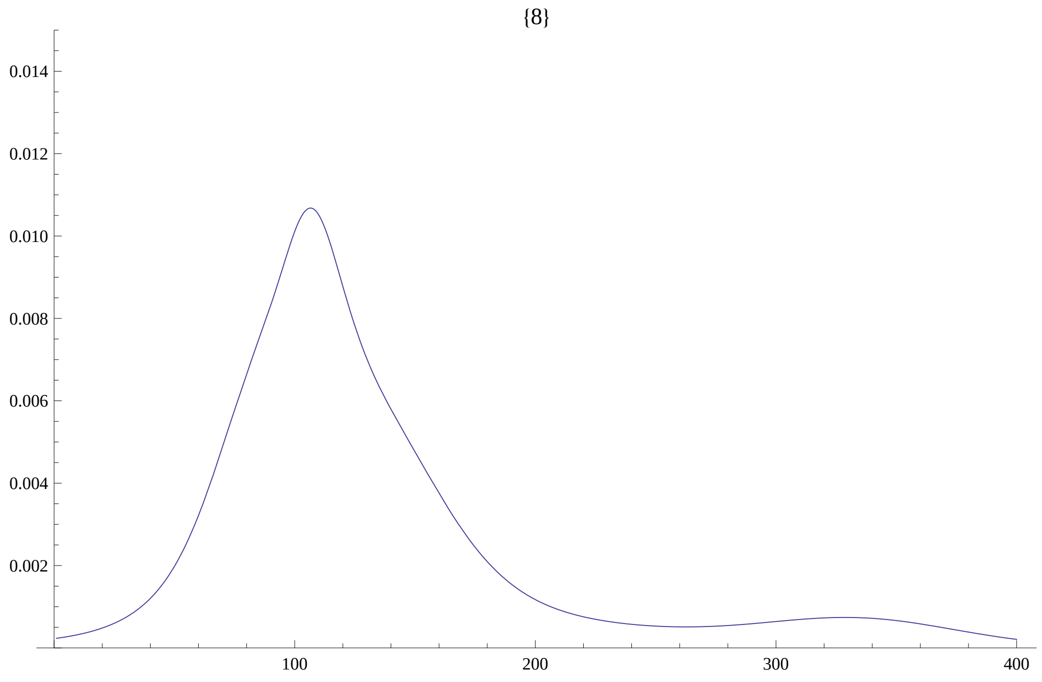


```
Pic2[7, 800, .005]
```



J2: Notice the spike at c50 generations ago, i.e. 650AD, the time of the Arab conquests.

```
Pic2[8, 400, .015]
```



Some idea of the TMRCA is obtained by averaging, either the individual times or the overall, notice the discrepancy

```
m1 = Table[ Sum [BB[j] * ZZZ01[[ q,  j]] , {j, 1, 33}] / 29.0, {q, 1, 8}]
```

```
{267.84, 288.422, 166.504, 122.94, 119.412, 122.29, 501.842, 133.084}
```

```
27 * m1
```

```
{7231.69, 7787.39, 4495.61, 3319.37, 3224.13, 3301.84, 13549.7, 3593.27}
```

```
P0 = Table[ NIntegrate[ Q0[x, q], {x, 1, 800}], {q, 1, 8}]
```

```
{0.906858, 0.92433, 0.931425, 0.957112, 0.974528, 0.989365, 0.756562, 0.986581}
```

```
m2 = Table[ NIntegrate[ x * Q0[x, q] / P0[[q]], {x, 1, 800}], {q, 1, 8}]
```

```
{177.328, 181.523, 153.992, 112.943, 123.952, 124.166, 273.327, 136.124}
```

```
27 * m2
```

```
{4787.86, 4901.12, 4157.79, 3049.47, 3346.71, 3352.48, 7379.82, 3675.35}
```

As the averages are not stable we use robust statistics, and stochastic simultations to get the MLE for the TMRCA

For later use we generate random quintiles at roughly 30,40,50,60,70%

```
n01 = 10000000; W0 = Flatten[Parallelize[Table[{Clear[ R, Z]; R = RandomReal[{.0005, .9995}, 29]; Z = Sort[R];
      {Z[[9]], Z[[12]], Z[[15]], Z[[18]], Z[[21]]}}, {k, 1, n01}]], 1]; {Mean[W0], √Variance[W0] }
```

```
{{0.300155, 0.40008, 0.499977, 0.599857, 0.699769}, {0.0822195, 0.0879131, 0.0897336, 0.0878987, 0.0822163}}
```

Using the distribution of times ZZZ01 we use bootstrap to generate the same quintiles for our data, inc SD

```
{n01, n00} = {100000, 33}; PPP[x_] := N[InverseCDF[NormalDistribution[0, 1.0], x]];
DD = Flatten[Table[{Clear[ LM, WW0, Z]; LM = Table[
        Log[ 1 + (.001 + ZZZ02[[q1, j]]) / (.1 + ZZZ01[[q1, j]]) ], {j, 1, 33}]; WW0 = Flatten[Parallelize[Table[
          {Clear[RR, R0, R1, RR, Z]; R0 = RandomReal[{.01, .99}, n00]; R1 = Table[ PPP[R0[[j]]], {j, 1, n00}];
           RR = Table[ BB[j] * ZZZ01[[q1, j]] * Exp[ LM[[j]] * R1[[j]]] , {j, 1, n00}]; Z = Sort[RR];
           {Z[[13]], Z[[16]], Z[[19]], Z[[22]], Z[[25]]}}, {k, 1, n01}]], 1];
     {NN[[q1]], Mean[WW0], √Variance[WW0] }}, {q1, 1, 8}], 1];
```

Thus for each file q our data used is the qunitile and its SD

```
MatrixForm[DD]
```

```
1221  {90.8499, 115.439, 140.954, 170.332, 217.298}   {10.1373, 12.621, 14.2566, 17.661, 28.7741}
 460  {114.848, 128.624, 143.707, 162.414, 189.27}     {8.4479, 9.2868, 10.872, 13.6341, 18.6516}
1270  {100.683, 114.985, 129.318, 150.008, 186.164}   {9.06927, 8.13395, 10.0857, 14.84, 24.0563}
2898  {59.2039, 69.987, 82.3342, 97.1986, 116.388}    {5.27506, 5.92387, 6.67114, 7.98144, 9.91024}
1029  {89.0055, 96.5624, 104.842, 114.359, 125.886}   {5.31427, 5.52968, 6.16197, 6.63901, 8.09177}
1533  {92.5846, 102.663, 114.04, 127.738, 144.532}    {6.10225, 6.59259, 7.74065, 9.00197, 10.6095}
1241  {176.613, 223.176, 272.816, 340.009, 500.669}   {24.0803, 29.1931, 31.6585, 53.483, 99.4678}
 971  {98.4029, 107.6, 117.107, 128.584, 143.855}     {6.21668, 6.16753, 7.00661, 8.59908, 11.1291}
```

This is now applied to each case, begiining with

We do G2a3 using 5 quintiles

```
KK = 1; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0 / D0[[1]],
  D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5 * D0[[2, 5]] }
```

```
{1221, 0.000819001, 90.8499, 115.439, 140.954, 170.332, 217.298, 115.439, 325.946}
```

The stochastic simulation uses the branching distribution τo to generate random quintiles. These are filtered by requiring they are within two SD of the data.   To speed the process we compiled the branching distribution in files 29ComFun together with interpolation file W29ComFun which must be loaded.   This information is contained in the function F4 used. This are the slowest part of the routine,   taking about an hour.

```
n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a,  J,  Z, Z0, Z1]; Y = RandomReal[{L1, L2}] ;
      a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
      Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
      {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]]  < D0[[2, 1]] + r*D0[[3, 1]] &&
     #[[1]]  > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
     #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]]  < D0[[2, 3]] + r*D0[[3, 3]] &&
     #[[3]]  > D0[[2, 3]] - r*D0[[3, 3]]  &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
   #[[4]]  < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]]  > D0[[2, 4]] - r*D0[[3, 4]] &&
     #[[5]]  < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]]  > D0[[2, 5]] - r*D0[[3, 5]]  &]; n05 = Length[W20]
```

```
253 250
```

Thus by filtering we now have 253,250 random quintiles within 2 SD of the experimental data.   The mean for these is

```
{Mean[ Table[ W20[[j, 6]] , {j, 1, n05}] ], √Variance[ Table[ W20[[j, 6]] , {j, 1, n05}] ] }
```

```
{235.85, 38.2347}
```

We now use least squares to find a quasilinear estimator which gives the best estimate of the TMRCA for all the random quintiles.

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1  +  WW3[[2]] * B2 +  WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW  = Table[
   BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 , {j, 1, n05}];
D1 = { n01, n05, Mean[BBW], √Variance[BBW]   }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];

BBW  = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 ,
   {j, 1, n05}]; D3 = { n01, n05, Mean[BBW], √Variance[BBW]   }
```

```
{5 000 000, 253 250, -4.46895×10^-16, 0.102957}
```

We see the QL estimates the TMRCA with average accuracy -4.46895×10^{-16} and overall SD 0.102957.   Thus we estimate the TMRCA from our experimental data in generations

```
g1 = BB1[D0[[2]] ]
```

```
272.582
```

Also we estimate the SD given the variance in the experimental data

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1) ^ j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √ Variance[WS] }
```

```
{272.582, 18.3486}
```

```
t1 = D4[[1]]
```

```
272.582
```

Thus our estimate for the TMRCA in ybp is

```
27*D4
```

```
{7359.7, 495.411}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.067314
```

One should not forget that the overall variance is the sum of the variance from sample error and the intrinsic error of the stochastic simulation

```
r0 = √(D3[[4]]^2 + D5^2)
```

```
0.12301
```

This % error gives SD in years:

```
.125*7300
```

```
912.5
```

ie for G2a3 we have 5359BC ± 912(1824 at 95% CI)

This is now repeated for each file, but this time without explanation

Next we do R1b1a2 using 5 quintiles

```
KK = 2; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0/D0[[1]],
  D0[[2, 1]],  D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5*D0[[2, 5]]  }
```

```
{460, 0.00217391, 114.848, 128.624, 143.707, 162.414, 189.27, 128.624, 283.904}
```

```
n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a,  J,  Z,  Z0, Z1]; Y = RandomReal[{L1, L2}] ;
      a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
      Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
      {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]]  < D0[[2, 1]] + r*D0[[3, 1]]  &&
     #[[1]]  > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
     #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]]  < D0[[2, 3]] + r*D0[[3, 3]] &&
     #[[3]]  > D0[[2, 3]] - r*D0[[3, 3]]  &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
   #[[4]]  < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]]  > D0[[2, 4]] - r*D0[[3, 4]] &&
     #[[5]]  < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]]  > D0[[2, 5]] - r*D0[[3, 5]]  &]; n05 = Length[W20]
```

```
320 405
```

```
{Mean[ Table[ W20[[j, 6]] , {j, 1, n05}] ],  √(Variance[ Table[ W20[[j, 6]] , {j, 1, n05}] ]) }
```

```
{198.638, 31.3521}
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1 + WW3[[2]] * B2 + WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW = Table[
  BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 , {j, 1, n05}];
D1 = { n01, n05, Mean[BBW], √Variance[BBW] }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
BBW = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 ,
  {j, 1, n05}]; D3 = { n01, n05, Mean[BBW], √Variance[BBW] }
```

```
{5 000 000, 320 405, 1.64125×10^-16, 0.0987823}
```

```
g1 = BB1[D0[[2]] ]
```

```
210.822
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1) ^ j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √Variance[WS]}
```

```
{210.822, 11.2174}
```

```
t1 = D4[[1]]
```

```
210.822
```

```
27 * D4
```

```
{5692.2, 302.869}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.0532077
```

```
r0 = √(D3[[4]] ^2 + D5 ^2)
```

```
0.112201
```

ie for R1b1a2 we have 3700BC ± 625(1250 at 95% CI)

Next we do R1a1 using 5 quintiles

```
KK = 3; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0 / D0[[1]],
  D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5 * D0[[2, 5]] }
```

```
{1270, 0.000787402, 100.683, 114.985, 129.318, 150.008, 186.164, 114.985, 279.245}
```

```
n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a, J, Z, Z0, Z1]; Y = RandomReal[{L1, L2}] ;
      a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
      Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
      {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]] < D0[[2, 1]] + r*D0[[3, 1]] &&
     #[[1]] > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
     #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]] < D0[[2, 3]] + r*D0[[3, 3]] &&
     #[[3]] > D0[[2, 3]] - r*D0[[3, 3]] &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z = {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
   #[[4]] < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]] > D0[[2, 4]] - r*D0[[3, 4]] &&
     #[[5]] < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]] > D0[[2, 5]] - r*D0[[3, 5]] &]; n05 = Length[W20]
```

```
315 232
```

$\{$Mean[Table[W20[[j, 6]] , {j, 1, n05}]], $\sqrt{\text{Variance[Table[W20[[j, 6]] , \{j, 1, n05\}]]}}\}$

```
{187.278, 32.116}
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] *B1 + WW3[[2]] *B2 + WW3[[3]] *B3 + WW3[[4]] *B4 + WW3[[5]] *B5
 + WW3[[6]]; BBW = Table[
  BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 , {j, 1, n05}];
```

D1 = $\{$n01, n05, Mean[BBW], $\sqrt{\text{Variance[BBW]}}\}$; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};

```
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] *BB[{B1, B2, B3, B4, B5}];
BBW = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 ,
```

{j, 1, n05}]; D3 = $\{$n01, n05, Mean[BBW], $\sqrt{\text{Variance[BBW]}}\}$

{5 000 000, 315 232, -7.57454×10^{-16}, 0.0944614}

```
g1 = BB1[D0[[2]] ]
```

```
218.451
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1) ^j * (D0[[3]].ee[[i]]) ee[[i]];
```

{BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = $\{$Mean[WS], $\sqrt{\text{Variance[WS]}}\}$

```
{218.451, 15.2288}
```

```
t1 = D4[[1]]
```

```
218.451
```

```
27 * D4
```

```
{5898.17, 411.177}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.0697126
```

```
r0 = √D3[[4]]^2 + D5^2
0.1174
```

```
.12*5900
708.
```

ie for R1a1 we have 4000BC ± 700(1400 at 95% CI)

Next we do I1 using 5 quintiles

```
KK = 4; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0/D0[[1]],
  D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5*D0[[2, 5]] }
{2898, 0.000345066, 59.2039, 69.987, 82.3342, 97.1986, 116.388, 69.987, 174.581}
```

```
n01 = 5000000; Clear[W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a, J, Z, Z0, Z1]; Y = RandomReal[{L1, L2}] ;
     a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10000000}]; Z0 = W0[[J]];
     Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
     {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]] < D0[[2, 1]] + r*D0[[3, 1]] &&
    #[[1]] > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
    #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]] < D0[[2, 3]] + r*D0[[3, 3]] &&
    #[[3]] > D0[[2, 3]] - r*D0[[3, 3]] &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
  #[[4]] < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]] > D0[[2, 4]] - r*D0[[3, 4]] &&
    #[[5]] < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]] > D0[[2, 5]] - r*D0[[3, 5]] &]; n05 = Length[W20]
184201
```

```
{Mean[ Table[ W20[[j, 6]] , {j, 1, n05}] ], √Variance[ Table[ W20[[j, 6]] , {j, 1, n05}] ] }
{131.605, 17.7687}
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
  {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1 + WW3[[2]] * B2 + WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW = Table[
  BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]] - 1 , {j, 1, n05}];
D1 = { n01, n05, Mean[BBW], √Variance[BBW] }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
BBW = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]] - 1 ,
  {j, 1, n05}]; D3 = { n01, n05, Mean[BBW], √Variance[BBW] }
{5000000, 184201, 2.00021×10^-15, 0.0955571}
```

```
x1 = BB1[D0[[2]] ]
141.407
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[{Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1)^j*(D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √Variance[WS]}
```

```
{141.407, 6.59365}
```

```
x1 = D4[[1]]
```

```
141.407
```

```
27*D4
```

```
{3817.99, 178.028}
```

```
D5 = D4[[2]]/D4[[1]]
```

```
0.0466289
```

```
r0 = √(D3[[4]]^2 + D5^2)
```

```
0.106327
```

```
.11*3800
```

```
418.
```

ie for I1 we have 1800BC ± 400(800 at 95% CI)

Next we do L21 using 5 quintiles

```
KK = 5; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0/D0[[1]],
  D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5*D0[[2, 5]] }
```

```
{1029, 0.000971817, 89.0055, 96.5624, 104.842, 114.359, 125.886, 96.5624, 188.83}
```

```
n01 = 5000000; Clear[W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a, J, Z, Z0, Z1]; Y = RandomReal[{L1, L2}] ;
     a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10000000}]; Z0 = W0[[J]];
     Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
     {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]] < D0[[2, 1]] + r*D0[[3, 1]] &&
    #[[1]] > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
    #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]] < D0[[2, 3]] + r*D0[[3, 3]] &&
    #[[3]] > D0[[2, 3]] - r*D0[[3, 3]] &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
  #[[4]] < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]] > D0[[2, 4]] - r*D0[[3, 4]] &&
    #[[5]] < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]] > D0[[2, 5]] - r*D0[[3, 5]] &]; n05 = Length[W20]
```

```
238308
```

```
{Mean[ Table[ W20[[j, 6]] , {j, 1, n05}] ], √Variance[ Table[ W20[[j, 6]] , {j, 1, n05}] ] }
```

```
{135.73, 16.8171}
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1 + WW3[[2]] * B2 + WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW = Table[ BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]] -
    1 , {j, 1, n05}]; D1 = { n01, n05, Mean[BBW], √Variance[BBW] }
```

```
{5 000 000, 238 308, -0.0112503, 0.0866072}
```

```
WW3
```

```
{-0.610508, 0.0474205, 0.123362, 0.202087, 1.17647, 0.486372}
```

```
BB[D0[[2]] ]
```

```
134.385
```

```
D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])}
```

```
{0.98875, 1.01138}
```

```
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1 + WW3[[2]] * B2 + WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW = Table[
  BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]] - 1 , {j, 1, n05}];
D1 = { n01, n05, Mean[BBW], √Variance[BBW] }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
BBW = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]] - 1 ,
   {j, 1, n05}]; D3 = { n01, n05, Mean[BBW], √Variance[BBW] }
```

{5 000 000, 238 308, 6.78057×10^{-17}, 0.0875926}

```
g1 = BB1[D0[[2]] ]
```

```
135.914
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1) ^ j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √ Variance[WS]}
```

```
{135.914, 4.85268}
```

```
t1 = D4[[1]]
```

```
135.914
```

```
27 * D4
```

```
{3669.69, 131.022}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.035704
```

```
r0 = √(D3[[4]]^2 + D5^2)
0.0945899

.09*3600
324.
```

ie for L21 we have 1600BC ± 325(650 at 95% CI)

Next we do U106 using 5 quintiles

```
KK = 6; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0/D0[[1]],
  D0[[2, 1]],  D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5*D0[[2, 5]]  }
{1533, 0.000652316, 92.5846, 102.663, 114.04, 127.738, 144.532, 102.663, 216.797}

n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a,  J,  Z,  Z0, Z1]; Y = RandomReal[{L1, L2}] ;
     a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
     Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
     {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]]  < D0[[2, 1]] + r*D0[[3, 1]]  &&
    #[[1]]  > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z,  Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z =  {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
    #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z,  Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z =  {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]]  < D0[[2, 3]] + r*D0[[3, 3]] &&
    #[[3]]  > D0[[2, 3]] - r*D0[[3, 3]]  &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a,   Z,  Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z =  {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
  #[[4]]  < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]]  > D0[[2, 4]] - r*D0[[3, 4]] &&
    #[[5]]  < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]]  > D0[[2, 5]] - r*D0[[3, 5]]  &]; n05 = Length[W20]
251 586

D0
{1533, {92.5846, 102.663, 114.04, 127.738, 144.532}, {6.10225, 6.59259, 7.74065, 9.00197, 10.6095}}

W20[[1]]
{87.2573, 110.743, 114.902, 142.764, 157.358, 179.199, 0.483426}

{Mean[ Table[ W20[[j, 6]] , {j, 1,  n05}] ],  √(Variance[ Table[ W20[[j, 6]] , {j, 1,  n05}] ] )}
{156.545, 21.4071}

WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]])^(-1),
  {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]]*B1  +  WW3[[2]]*B2 +  WW3[[3]]*B3 + WW3[[4]]*B4 + WW3[[5]]*B5
 + WW3[[6]]; BBW  = Table[
  BB[ {W20[[j, 1]],  W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 , {j, 1, n05}];
D1 = {n01, n05, Mean[BBW], √(Variance[BBW])  }; D2 = {1 + D1[[3]], 1/(1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]]*BB[{B1, B2, B3, B4, B5}];
BBW  = Table[ BB1[ {W20[[j, 1]],  W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 ,
  {j, 1, n05}]; D3 = {n01, n05, Mean[BBW], √(Variance[BBW])  }
{5 000 000, 251 586, -1.53004×10^-16, 0.0893821}
```

```
g1 = BB1[D0[[2]] ]
```

```
162.787
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1)^j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √Variance[WS] }
```

```
{162.787, 6.86725}
```

```
t1 = D4[[1]]
```

```
162.787
```

```
27 * D4
```

```
{4395.25, 185.416}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.0421855
```

```
r0 = √D3[[4]]^2 + D5^2
```

```
0.0988371
```

```
.10 * 4400
```

```
440.
```

ie for U106 we have 2400BC ± 440(880 at 95% CI)

Next we do J2 using 5 quintiles

```
KK = 7; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0 / D0[[1]],
  D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]] , 1.5 * D0[[2, 5]] }
```

```
{1241, 0.000805802, 176.613, 223.176, 272.816, 340.009, 500.669, 223.176, 751.003}
```

```
n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a, J, Z, Z0, Z1]; Y = RandomReal[{L1, L2}] ;
     a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
     Z1 = {Y * F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
     {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]] < D0[[2, 1]] + r * D0[[3, 1]] &&
    #[[1]] > D0[[2, 1]] - r * D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Y * F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r * D0[[3, 2]] &&
    #[[2]] > D0[[2, 2]] - r * D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Y * F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]] < D0[[2, 3]] + r * D0[[3, 3]] &&
    #[[3]] > D0[[2, 3]] - r * D0[[3, 3]] &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a, Z, Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
     Z = {Z0[[1]], Z0[[2]], Z0[[3]], Y * F4[Z0[[4]], Y, a, m1], Y * F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
     {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
  #[[4]] < D0[[2, 4]] + r * D0[[3, 4]] && #[[4]] > D0[[2, 4]] - r * D0[[3, 4]] &&
    #[[5]] < D0[[2, 5]] + r * D0[[3, 5]] && #[[5]] > D0[[2, 5]] - r * D0[[3, 5]] &]; n05 = Length[W20]
```

```
338 773
```

```
{Mean[Table[W20[[j, 6]], {j, 1, n05}]], √Variance[Table[W20[[j, 6]], {j, 1, n05}]]}
```

```
{482.675, 106.163}
```

```
WW1 = Table[{W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * (W20[[j, 6]])^(-1),
   {j, 1, n05}]; WW2 = Table[1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1 + WW3[[2]] * B2 + WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW = Table[
   BB[{W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]}] / W20[[j, 6]] - 1, {j, 1, n05}];
D1 = {n01, n05, Mean[BBW], √Variance[BBW] }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
BBW = Table[BB1[{W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]}] / W20[[j, 6]] - 1,
   {j, 1, n05}]; D3 = {n01, n05, Mean[BBW], √Variance[BBW] }
```

$\{5\,000\,000,\ 338\,773,\ -9.58356\times10^{-16},\ 0.107998\}$

```
t1 = BB1[D0[[2]]]
```

```
650.798
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[{Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1)^j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √Variance[WS]}
```

```
{650.798, 66.5992}
```

```
t1 = D4[[1]]
```

```
650.798
```

```
27 * D4
```

```
{17571.6, 1798.18}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.102335
```

```
r0 = √D3[[4]]^2 + D5^2
```

```
0.148781
```

```
.15 * 17500
```

```
2625.
```

ie for J2 we have 15500BC ± 2600(5200 at 95% CI), this is definitely Paleolithic.

Next we do P312 using 5 quintiles

```
KK = 8; D0 = DD[[KK]]; {n0, m1, b1, b2, b3, b4, b5, L1, L2} = {D0[[1]], 1.0 / D0[[1]],
   D0[[2, 1]], D0[[2, 2]], D0[[2, 3]], D0[[2, 4]], D0[[2, 5]], D0[[2, 2]], 1.5 * D0[[2, 5]] }
```

```
{971, 0.00102987, 98.4029, 107.6, 117.107, 128.584, 143.855, 107.6, 215.782}
```

```
n01 = 5 000 000; Clear[ W10]; r = 2;
W10 = Flatten[Parallelize[Table[{Clear[ Y, a,  J,  Z,  Z0, Z1]; Y = RandomReal[{L1, L2}] ;
      a = RandomReal[{.05, 1.3}]; J = RandomInteger[{1, 10 000 000}]; Z0 = W0[[J]];
      Z1 = {Y*F4[Z0[[1]], Y, a, m1], Z0[[2]], Z0[[3]], Z0[[4]], Z0[[5]]} ; Z = Join[Z1, {Y, a}];
      {Z}}, {k, 1, n01}]], 2]; W11 = Select[W10, #[[1]]  < D0[[2, 1]] + r*D0[[3, 1]]  &&
     #[[1]]  > D0[[2, 1]] - r*D0[[3, 1]] &]; n02 = Length[W11];
W12 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z,  Z0]; Z0 = W11[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Y*F4[Z0[[2]], Y, a, m1], Z0[[3]], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n02}]], 2]; W13 = Select[W12, #[[2]] < D0[[2, 2]] + r*D0[[3, 2]] &&
     #[[2]] > D0[[2, 2]] - r*D0[[3, 2]] &]; n03 = Length[W13];
W14 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z,  Z0]; Z0 = W13[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Z0[[2]], Y*F4[Z0[[3]], Y, a, m1], Z0[[4]], Z0[[5]], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n03}]], 2]; W15 = Select[W14, #[[3]]  < D0[[2, 3]] + r*D0[[3, 3]] &&
     #[[3]]  > D0[[2, 3]] - r*D0[[3, 3]]  &]; n04 = Length[W15];
W16 = Flatten[Parallelize[Table[{Clear[ Y, a,  Z,  Z0]; Z0 = W15[[k]]; Y = Z0[[6]] ; a = Z0[[7]];
      Z =  {Z0[[1]], Z0[[2]], Z0[[3]], Y*F4[Z0[[4]], Y, a, m1], Y*F4[Z0[[5]], Y, a, m1], Z0[[6]], Z0[[7]]} ;
      {Z}}, {k, 1, n04}]], 2]; Clear[ W20]; W20 = Select[W16,
   #[[4]]  < D0[[2, 4]] + r*D0[[3, 4]] && #[[4]]  > D0[[2, 4]] - r*D0[[3, 4]] &&
     #[[5]]  < D0[[2, 5]] + r*D0[[3, 5]] && #[[5]]  > D0[[2, 5]] - r*D0[[3, 5]]  &]; n05 = Length[W20]
```

```
268 262
```

```
{Mean[ Table[ W20[[j, 6]] , {j, 1, n05}] ], √Variance[ Table[ W20[[j, 6]] , {j, 1, n05}] ] }
```

```
{153.844, 20.707}
```

```
WW1 = Table[ { W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]], 1} * ( W20[[j, 6]]) ^ (-1),
   {j, 1, n05}]; WW2 = Table[ 1, {j, 1, n05}];
WW3 = LeastSquares[WW1, WW2]; BB[{B1_, B2_, B3_, B4_, B5_}] :=
 WW3[[1]] * B1  +  WW3[[2]] * B2 +  WW3[[3]] * B3 + WW3[[4]] * B4 + WW3[[5]] * B5
 + WW3[[6]]; BBW  = Table[
   BB[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 , {j, 1, n05}];
D1 = {n01, n05, Mean[BBW], √Variance[BBW] }; D2 = {1 + D1[[3]], 1 / (1 + D1[[3]])};
BB1[{B1_, B2_, B3_, B4_, B5_}] = D2[[2]] * BB[{B1, B2, B3, B4, B5}];
BBW  = Table[ BB1[ {W20[[j, 1]], W20[[j, 2]], W20[[j, 3]], W20[[j, 4]], W20[[j, 5]]} ] / W20[[j, 6]]  - 1 ,
   {j, 1, n05}]; D3 = {n01, n05, Mean[BBW], √Variance[BBW] }
```

```
{5 000 000, 268 262, -3.42538×10^-17, 0.0886386}
```

```
t1 = BB1[D0[[2]] ]
```

```
156.838
```

```
ee = {{1, 0, 0, 0, 0}, {0, 1, 0, 0, 0}, {0, 0, 1, 0, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}}; WS =
 Flatten[Table[ {Clear[b1, b2, b3, b4, b5]; {b1, b2, b3, b4, b5} = D0[[2]] + (-1) ^ j * (D0[[3]].ee[[i]]) ee[[i]];
    {BB1[{b1, b2, b3, b4, b5}]}}, {i, 1, 5}, {j, 0, 1}]]; D4 = {Mean[WS], √ Variance[WS] }
```

```
{156.838, 6.66823}
```

```
t1 = D4[[1]]
```

```
156.838
```

```
27 * D4
```

```
{4234.63, 180.042}
```

```
D5 = D4[[2]] / D4[[1]]
```

```
0.0425166
```

```
r0 = √D3[[4]]^2 + D5^2
0.098308
```

ie for P312 we have 2240BC ± 420(820 at 95% CI)

This information can be summarized by following showing the two means compared with our calc TMRCA and SD

SNP	mean1	mean2	TMRCA	SD
G2a3	5231 BC	4747 BC	5359 BC	912
R1b1a2	5787 BC	2901 BC	3700 BC	625
R1a1	2495 BC	2157 BC	3800 BC	700
I1	1319 BC	1049 BC	1800 BC	400
L21	1224 BC	1346 BC	1600 BC	325
U106	1301 BC	1352 BC	2400 BC	440
J2	11 549 BC	5379 BC	15 500 BC	2600
P312	1593 BC	1675 BC	2240 BC	420

The means only give the right ballpark estimate, usually more than a SD less than the true TMRCA.