Population Kinetics Across the Indo-Pacific Region

Aydar Aliev

Submitted in fulfillment of a Masters in Philosophy

Massey University, New Zealand

19 August 2014

Table of Contents

Table of Figures

Table of Tables

1. Outline

1.1 Research Questions

The Pacific region provides a natural system to study complex admixture. From a broad perspective, there were two waves of settlement; the first 45,000 years ago (Melanesian), and the second, approximately 5,000 years ago (Asian) [1]. According to recent research, Asian ancestry does not decline gradually across Island Southeast Asia, but instead dramatically decreases, forming a cline [2]. There are several hypotheses explaining why there is a drastic, but not gradual, change in genetic ancestry proportions (Asian to Melanesian) across the region. One of these is a steep change in environmental conditions in Eastern Indonesia, which complicates rice cultivation [3]. Another explanation can be the switch from matri- to patriarchal social systems [4]. The main goal of this project is to explore demographic factors, such as migration and selection, to see if they can explain the genetic ancestry distribution.

The main theoretical question that I will answer is: **what is the reason behind the steep change in genetic ancestry proportion across eastern Indonesia?** One of the reasons behind this could be cultural selection, although selection is just a hypothesis and the process might be selectively neutral.

Anthropological data from the region are quite sparse, and this leads to the second goal of the project: **to infer the history of modern Pacific populations using genetic data**.

1.2 Rationale and Importance of the Study

Many populations experienced admixture with similar levels of complexity to the Pacific region [5]. Complex admixture dominates many other natural and experimental systems as well [6]. Several works describing this admixture [2, 3, 7, 8] have been published, but as yet, none of them explained the cause of the steep cline in Asian ancestry across the region.

At the moment, there are several programs capable of describing population admixture proportions available [9, 10]. They utilize SNP data. Common limitations are that they do not allow us to infer underlying processes and parameters, such as effective population size and migration rates. Another limitation is that they do not provide confidence intervals. However, it is possible to identify said parameters using Agent Based Modelling (ABM) approaches, which in its turn will reduce the time required for fitting of the parameters in further simulations. The aim of the following simulations is to provide molecular diversity comparable to actual molecular diversity, which could be further analyzed using coalescent based approaches such as SPLATCHE2 [11].

Apart from reconstructing history, another benefit of the work is that it can make a contribution to a basis of future biomedical research, since there is an increasing demand for methods that allow researchers to differentiate between genetic patterns produced by demographic history and disease susceptible genotypes [12].

1.3 Description of Available Data

Glossary:

SNP chip – Consider short (usually 15-25 nucleotides) DNA sequences attached to a solid base (chip). When a fluorescently labeled genetic sample is applied to the chip, hybridization occurs. This allows us to scan the chip with a laser, and in our case, obtain a picture of SNPs presented in a genome.

Ancestry Informative Markers (AIMs) – are SNPs which are at high frequency for one population, thus reflecting the ancestry of individuals.

At the moment, two main datasets are available. First, the Cox group, Eijkman Institute and the University of Arizona genotyped 500,000 autosomal SNPs in 250 individuals from across the Indonesian archipelago using the Affymetrix 550k SNP chip (Fig. 1.3.1; supplementary materials, Table S1). Second, the same NZ-US-Indonesian consortium genotyped 37 AIMs in 1,430 individuals from 60 populations spanning mainland Asia to Melanesia.

Similar datasets are also available for future work: the HUGO Pan-Asian SNP Consortium has screened 50,000 autosomal SNPs in 1,928 individuals from 73 Asian populations using the Affymetrix 50k SNP chip, and the 1000 Genomes Project has released full genome sequences for several individuals from Asian parental populations.

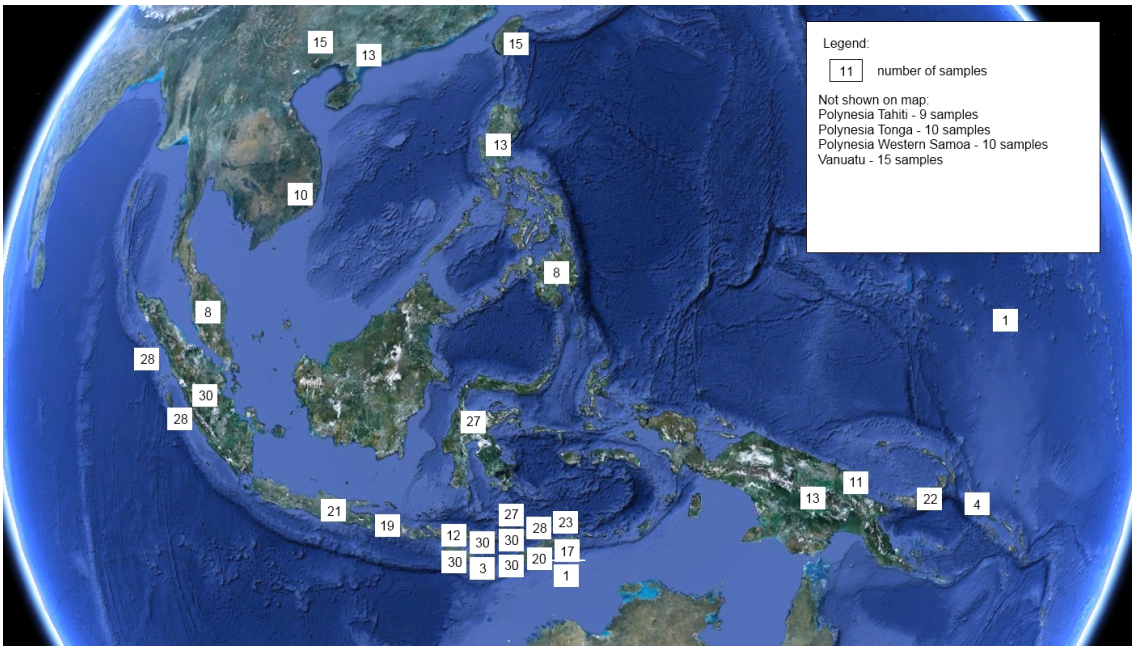

Figure 1.3.1: Samples map. The Affymetrix 550k SNP chip was used to assay individuals. Each square corresponds to one local population. A number in the square shows the number of assayed individuals.

2. Literature Review

2.1 History of Human Pacific Populations

In 1890 Wallace, after travelling through Indonesia, described intergradation of the human population. He noted that phenotypic change occurs in Eastern Indonesia [15]. The described phenotypic change line denoted change from Asian to Melanesian phenotype.

Later in 1924 Bais and Verhoef contributed to the research question by reporting differences in frequencies of blood groups between populations of Java and New Guinea [16]. Two-thirds of 20th century research was summarized by Hudson in 1974 [17]. Researchers mostly utilized serological markers, various anthropomorphic parameters and linguistic methods [18] to differentiate between populations. Work describing and comparing anthropomorphic characteristics of populations within the region were published until the mid-nineties. For example, Hanihara [19] and Pietrusewsky [20] described differences in craniofacial features between Asian and Melanesian populations. Interestingly, Hanihara noted that Australian aboriginals showed more similarities to African rather than to Melanesian populations. Also Melanesians can be characterized by dark skin color [21, 22]. These facts suggest that Melanesian populations can be distinguished into one cluster. However, since the discovery of PCR in 1983, the attention of the scientific community has shifted to comparing genomes. Thus, genome comparison replaced anthropometrics, but not linguistic methods.

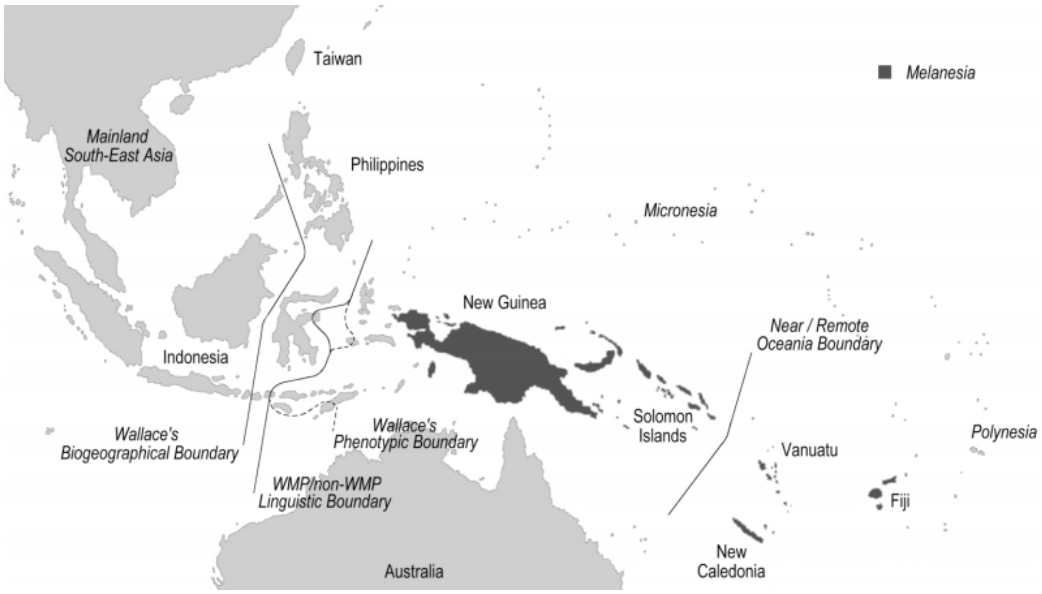

Figure 2.1.1: Population boundaries in the Pacific region [3].

Historically, there were two models explaining the settlement of the Pacific. Although new large scale (e.g., SNP chip data) datasets suggest that both models are only partially true, they were prevalent in Anthropology for several decades and are interesting from that perspective.

The model by Bellwood [5, 23] assumes that indigenous Melanesian populations, the modern descendants of which are Papuan, settled in the region approximately 50,000 years ago. Then about 5,000 years ago, wide adoption of agriculture in Asian communities near modern Taiwan and mainland China lead to expansion of Asian populations in an eastern direction [5, 24]. This hypothesis also has some linguistic support [25].

The model by Oppenheimer [26] conforms to the Bellwood model in the assumption that the Pacific region was settled about 50,000 years ago. However, contrary to Bellwood, Oppenheimer proposed that current populations are branches of one parental population, and not the result of the admixture and migration processes between Asian and Melanesian populations. In other words, modern human population diversity in the region evolved from one parental population. According to this model, eastern Indonesian populations expanded to remote Oceania approximately 10,000 years ago.

Most genetic research within the area has been done using the following types of data: mitochondrial DNA and Y-chromosome. However, more recently, the use of autosomal SNPs has gained popularity.

Hertzberg in 1989 [27], studying mtDNA from Polynesian, Australian and Melanesian aboriginals, found an Asian-associated 9-bp deletion in Polynesian mtDNA samples and the absence of this deletion in Papuan and Australian individuals. This finding was later confirmed and extended by Merriwether in 1999 [28]; the deletion was absent in a remote Papuan and almost all Australian samples, while it was present in eastern regions of Papua New Guinea, which are closer to Polynesia. It was also shown that Asian, Papuan and Australian populations each have their own characteristic haplogroups [29, 30].

There are similarities in the geographical distribution of Y-chromosome and mtDNA haplogroups. Numerous papers studying the molecular variance of Y-chromosome diversity within the region were published during the last two decades [1]. The most recent nomenclature system was published in 2008 [31]. A study by Kayser in 2001 [32] showed the presence of Asian haplotype M119C/M9G in coastal Papua New Guinea and its absence in both Highland and Australian samples. Additionally, the study did not reveal shared haplotypes between Australian and Papuan populations, which allows hypothesizing on the independent history of Papuan and Australian populations. In 2006 Kayser, comparing Polynesian Y chromosome and mtDNA data with potential parental populations from Asia, Australia and Melanesia, proposed that there is an admixture bias towards Melanesian men, as the Y-chromosome in Polynesian men predominantly comes from the Melanesian population. It is thought that the admixture event happened before the settlement of Polynesia. Another study by Mona in 2007 [34] showed that 97.5% of 162 samples obtained in the northwest region of Papua New Guinea originated in Melanesia, while the remaining 2.5% of samples contained the Asian-related O haplogroup. Another large scale study of 1,917 men published in 2010 by Karafet [35] revealed that Y-chromosome samples taken in eastern parts of the region had Melanesian haplogroups, while samples taken in the western parts carried Asian haplogroups. The border between the two clusters lies between Bali and Flores islands.

Research conducted up to 2010 was summarized by Kayser [36]. The earliest Pacific work utilizing autosomal SNP data was published in 2008 by Friedlaender [37]. The molecular diversity of the Pacific region was described by sampling 952 individuals from 41 populations. It is shown that isolated Papuan populations have decreased molecular diversity, which inflates the distances between Papuan populations. While there is a clear difference in genetic ancestry of Asian and Melanesian populations, genetic distances between Asian populations tend to be smaller.

A different approach was implemented by Cox in 2010 [2]. 37 autosomal Ancestry Informative Markers (AIMs) were identified, which are SNPs with significant F_{ST} values between Han Chinese and Papua New Guinea highlanders. These selected SNPs were assayed in 1,460 individuals across the region. Ancestry admixture proportions were calculated by defining a pseudo-parental population and measuring their SNP frequencies. Pseudo-parental populations are modern populations, which are presumed to be closest to the real ancestral population. In this particular case, Han Chinese and Papuan highlanders were selected as two pseudo-parental populations. An advantage of this approach is its ability to estimate ancestry admixture proportions, by comparing allele frequencies between pseudo-parental and presumably admixed populations (Fig. 2.1.2), because autosomal chromosomes come from maternal and paternal lineages and AIMs are independent from each other. Thus, this allows us to check if Bellwood's model migration assumptions hold.

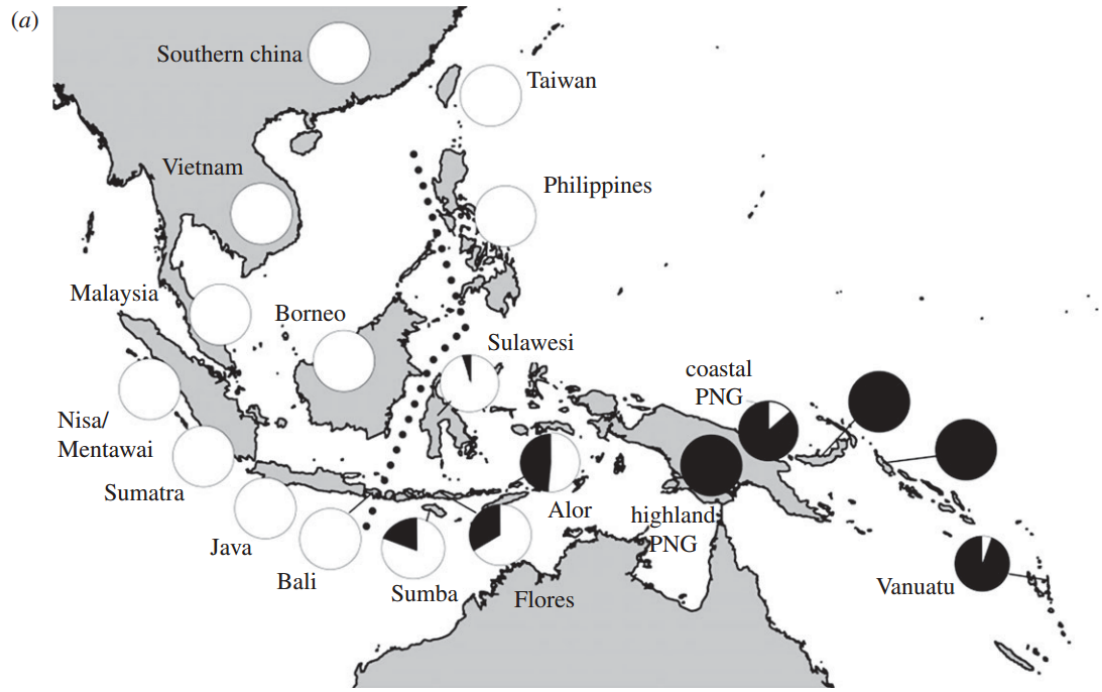

Figure 2.1.2 [2]: Admixture proportions from populations across the Pacific region. The dotted line is Wallace's biogeographic line. The Asian ancestry proportion is shown in white; Melanesian ancestry proportion is shown in black.

In 2010, Wollstein [8] also utilized autosomal SNPs to infer the history of Pacific demography. Two main groups from Polynesia and Papua New Guinea were assayed. Han Chinese, Japanese, African and Caucasian samples were also included in the

dataset. **Frappe** (see Section 2.2) was used to infer ancestry proportions. The researchers varied the number of assumed ancestral populations (*K*) from 2 to 6. When *K* was equal to two, individuals clustered into two groups: Africans versus non-Africans. When K was equal to four, Papuan, African, Caucasian and Asian/Polynesian samples appeared as different clusters. The authors utilized coalescent theory and Approximate Bayesian computation (see Sections 2.5 and 2.6) to propose that the split between Oceanian and Eurasian populations occurred approximately 27,000 years ago.

Adoption of autosomal SNP data allowed the description of admixture proportions. Potentially high-density SNP data allow not only to infer admixture proportions, but more importantly, to estimate when the admixture event occurred. Pugach in 2011 [38] developed a theory, also known as wavelet transform analysis. This estimation is done in two steps. In the first step, an algorithm specifies a sliding window along the chromosome and assigns it a score. The score is based on pseudo-parental populations allele frequencies. In the second step, these scores are transformed into wavelet signals. The wavelet frequencies are then used to infer ancestral block sizes. Knowledge of block size is then used to infer the time back to the admixture event. Utilizing this method, Pugach inferred the admixture time in a Polynesian population, which resulted from admixture between Asian and Melanesian populations. The admixture event happened 2,700 years ago.

Another study of population stratification using autosomal SNPs data was conducted by Xu [39] in 2012. In his study, he analyzed two different datasets. One contained 288 individuals from 13 Austronesian-speaking populations and two Papuan-speaking populations. Another dataset contained SNP data received from 36 individuals, from 7 populations in Indonesia and 25 individuals from Papua New Guinea. The Affymetrix 680,000 SNP chip was used. The authors utilized **Structure** and **Frappe** (Section 2.2) to perform population stratification (Fig. 2.1.3). Pugach's wavelet decomposition method was used to date the admixture event. The admixture time was estimated to have happened 121-204 generations or 3,026-5,109 years ago.

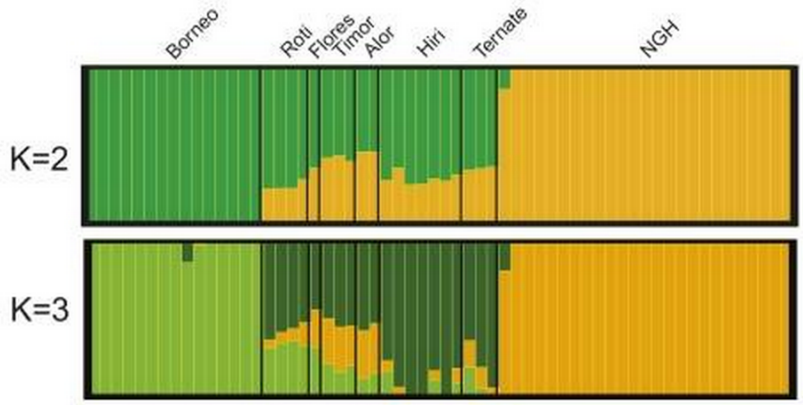

Figure 2.1.3 [39]: Population stratification of 680,000 SNPs from 36 individuals using Frappe. K reflects an assumed number of ancestral populations. Varying K is necessary because although the main hypothesis suggests two parental populations, there might have been more than two. For example, Australian aboriginals carry specific mtDNA lineages, which are not found in Indonesian and Papuan populations. Each color corresponds to an ancestry component.

On a broader scale, "biological" methods are not the only ones implemented within this research area. For example, in 2000, Gray [40] used lexical data to build a phylogenetic tree of Austronesian languages (which are languages of the Pacific region, excluding Papua New Guinea), thus showing that they had common ancestor. One of the landmarks of this Austronesian language family study was the creation of the Austronesian Basic Vocabulary Database in 2008 by Greenhill [41]. The database contains wordlists from more than 500 languages. This database was used by Gray in 2009 [42] to identify phylogenetic relationships between 400 Austronesian languages. The implementation of Bayesian phylogenetic methods allowed not only the description of relationships between the languages but also characterization of the ancestor language as originating in Taiwan about 5,000 years ago.

A recent study by Atkinson [43] focusing on identification of cognate words within the Austronesian language family was able to identify and successfully build the language tree reconstructing the descent of words from Proto-Austronesian (Fig. 2.1.4).

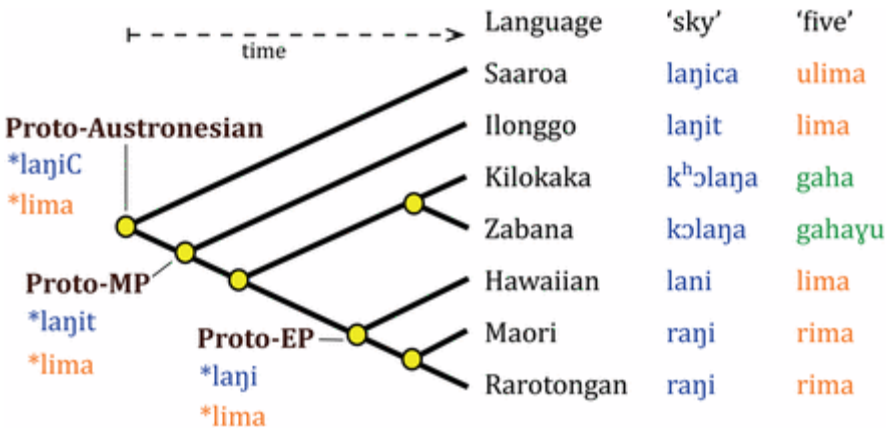

Figure 2.1.4 [43]: Reconstruction of words' descent on the language tree. The words' translation to English are sky and five. Cognate forms are coded blue, orange and green. Yellow nodes represent ancestral protolanguages. For two bigger cognate groups, word forms are shown for reconstructed ancestral languages.

Bellwood's model states that modern Pacific populations resulted from an admixture of Asian and Melanesian migration waves separated in time, whereas Oppenheimer assumed that the modern Pacific population resulted from one parental population. Overall, the research within the last two decades suggests that the Bellwood model is more consistent with real world data.

2.2 Ancestry Description Software

In this section, I will provide a summary of programs I used to infer individual admixture proportions.

Today, several software solutions allowing the description of ancestry exist. One of the most popular programs is **Structure** [44] by J. Pritchard, the original algorithm of

which was described in the year 2000. Several extensions of it were published later [45, 46]. **Structure** can run in four modes. No admixture mode, if there is an assumption that all samples come from different populations. The admixture mode, if there is an assumption that samples come from two or more admixed populations. The linkage mode can be utilized to take into account linkage disequilibrium between loci in admixed populations, thus inferring the admixed regions along a chromosome [45]. Finally, there is an option to include sampling location information to achieve better results with limited data [47].

Before setting up the run, it is necessary to have a hypothesis on the number of ancestral populations. However, **Structure** can calculate the most probable *K* out of many, if needed. **Structure** approach is based on MCMC (Markov Chain Monte Carlo). **Structure** assumes *K* populations. Each population is characterized by frequencies at each SNP position. Samples are then assigned to a population, or to two or more populations if there is an indication of admixture. The basic **Structure** algorithm assumes Hardy-Weinberg equilibrium and independent SNPs for subpopulations, except when the linkage disequilibrium option is activated. To summarize, it calculates the probability of an observed SNP set in a sample given allele frequencies in the ancestral populations. Then **Structure** clusters individuals based on that information [48].

Frappe [50] was released in 2005, by Tang et al. at Stanford University. The model of **Frappe** assumes that each person's genotype is a mixture of alleles coming from different populations. Because collecting samples from ancestral populations is usually impossible, individuals from populations closely related to ancestral ones are included into the dataset. For example, in our case, individuals from pseudo-ancestral populations are Han Chinese and Papuan samples, because those are assumed to be closest to ancestral Asian and Melanesian populations. Input data should contain samples from pseudo-ancestral and admixed individuals. Frappe requires specification of K (the number of ancestral populations) to run, and assumes Hardy-Weinberg equilibrium. Frappe utilizes a Maximum Likelihood approach to estimate genetic ancestry proportions. An individual's admixture proportion is set up by a vector $Q_i = (q_{i1} \dots q_{ik})$, where each q corresponds to the probability that an allele comes from population *K*. Initially, values along the vector are assigned randomly, and the value of the likelihood function is computed. The program then iterates until values maximizing the function are found. A disadvantage of **Frappe** is its inability to compute the most probable *K*.

Admixture [9, 52] is the latest program among those described above. As with **Frappe**, it focuses on Maximum Likelihood estimation, but maximizes the likelihood function utilizing a different algorithm. It is possible to specify several values of *K* (in our case from 2-6) and calculate error rates for each. It has a strong advantage over **Frappe** and **Structure** in terms of computational performance with big datasets. It is possible to obtain the same results within hours in comparison to days for Frappe and months for Structure.

2.3 Agent Based Modeling (ABM) in Biology

An agent is a self-sufficient object, which can represent anything from a molecule to an individual.

An agent-based model can be described as an environment where a discrete number of agents interact with each other. The rules of interaction between agents, the agent's evolution through time and environmental characteristics are specified by the researcher. Ultimately, this approach allows us to observe emergence. 'Emergence' is a property of a system, which emerges at a certain level of organization. One cannot observe the emergent property on an individual level. In our case, the emergent property of interest is a cline in the Asian ancestry proportion corresponding to geographical latitude. Usually, the ultimate goal of ABM is the re-creation or prediction of system behavior.

Agent based modeling traces its roots to cellular automata, a concept which was originally described in the late 1940s by von Neumann [53]. The purpose of his work was to explore machine self-replication; for example, a 3D printer printing its own copy. In the late 1960s, John Conway created his famous Game Of Life [54], in which each cell has two states, "alive" or "dead". The state of each cell changes according to neighboring cells' states. The reason behind the popularity of Conway's game was that, given simple rules and changing them, it was possible to observe changes in system behavior and evolution. But overall, agent based modeling was not widely used until the early 1990s.

The key work that popularized the ABM approach was published in 1987 by Reynolds [55]. This work models the behavior of flocking birds (known as the 'boids' model), but as the author notes, the model can also be applied to any other groups of animals. Reynolds specified simple behaviors (or rules) for each bird (or more generally speaking – an agent), which allowed him to simulate complex behavior. The rules were:

1. Clash preclusion
2. Speed synchronization (with close flock mates)
3. Centralization (agents aim was to stay close to flock mates)

The model allowed the simulation of realistic behavior of animal groups and was successfully implemented in several different fields. One of the earliest examples of this implementation is in the movie "Batman Returns", where the boids model was used to create an animated penguin army of the Penguin attacking Gotham City [56].

In the general case, an ABM can be described as an environment, which holds self-sufficient agents. The environment sets the background for agents to make their own decisions. Minimally, the environment should provide a time step for an agent. The time step within a model can correspond to any discrete time interval (a second, month, year). Other facilities that should be provided by the environment heavily depend on the modeled system. The agent relies on the variables provided by the environment (e.g., time, food gradient, space coordinates) to make its own decisions on whether to die, migrate, mate, bind to another agent, etc.

The main difference between ABM approaches and more widely used equation-based modeling is that equation-based modeling considers a system as a set of processes, which can be described by a system of equations, the variables of which depend on one another. From the population genetics point of view, a disadvantage of equation-based modeling is that it is not possible to track the state of an individual, as only variables describing the state of the whole population are possible.

Even though ABM is implemented in many research areas [57], in this section I will concentrate on ABM examples within the Life Sciences domain.

Ecologists were one of the earliest adopters of ABM approaches. In 1980, DeAngelis [58] created an ABM model to study populations of one year old largemouth bass and their switching to cannibalism in relation to food sources. The model of DeAngelis considered each fish as an independent variable, and took into account natural death, as well as death due to the cannibalism, food source and growth of individuals.

During the 1980s, several works discussing and implementing ABM within the Life Sciences were published [59, 60]. However, the ABM approach was not widely used until the 1990s.

During the 1990s, the number of ABM publications drastically increased. The overwhelming majority of Agent Based Models created within this decade aimed to study emergent properties of various population(s). A large proportion of models came from the field of fish population biology [61]. For example, in 1991, DeAngelis [62] using ABM studied changes of smallmouth bass population age structure after changes in population density. Nests and fishes were considered as agents. After swim-up from the nest, each fish encountered randomly distributed prey. Growth and mortality were dependent on the prey availability. Ultimately, the model describes dependencies between populations and larval density. In general, a lot of ABM work during this decade considers not only natural limitations of the environment, but also aspects of individual behaviour within a population and/or population social structure [63-66].

The first attempt of agent based modelling to take into account GIS data was undertaken in 1997 by Kohler, in which he attempted to describe the settlement formation of prehistoric North American populations. However, only a 1995 paper describing the early stages of the project was found [67]. Another work studying population dynamics dependent on a landscape was published by Henein in 1998 [68]. This explored the effects of landscape spatial structure, changes in environmental conditions and connectivity of landscape types (woods-fencerow-agricultural fields) on the population size and survival of eastern chipmunks and white-footed mice. A landscape was simulated as a 64x64 cell grid. Researchers varied the proportion of wood to agricultural field on parts of the grid. It was shown that mice surpass chipmunks on all forms of landscapes and the connectivity between types of landscape can be a predictor of chipmunk success.

One of the most influential works, which took into account genetic loci, was published in 1999 by Dieckmann and Doebeli [69]. These authors studied sympatric speciation in abstract populations (i.e., how species are formed from a parental population without geographical separation). Two models were created with and

without sexual reproduction. In each model, individuals varied a quantitative trait **x** (for example, bird beak size) that determed resource use. Competition between individuals depended on the size of the difference in trait x: when the difference was large, authors assumed no competition; conversely, when the difference was smaller, there was more competition. In the model, sexual reproduction quantitative traits were determined by additive diploid loci (+ and -) inherited from a father and mother, independently (free recombination). The trait was proportional to the number of + alleles.

To incorporate assortative mating into the model, the authors added additional parameters. In the first scenario, the additional mating probability was based on similarities in quantitative trait **x** between two individuals. In the second scenario, the mating probability was based on selectively neutral quantitative 'markers', also determined by a combination of + and – alleles. Individuals carrying intermediate value **s** of the 'marker' mated randomly, while individuals carrying mostly negative alleles preferred to mate with the opposite type, and individuals carrying mostly + alleles preferred to mate with their own ilk (Figure 2.3.1).

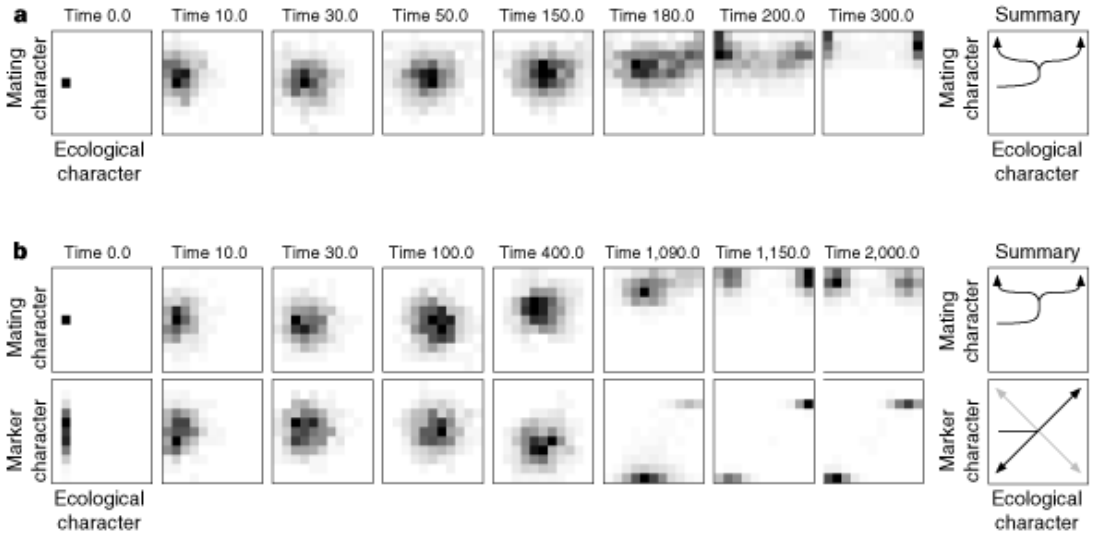

Figure 2.3.1 [69]: Speciation. a, Mating probabilities depend on the trait x. It takes 50 generations for x to reach an average value, and then speciation occurs. b, Mating probabilities depend on selectively neutral markers. Speciation still occurs due to genetic drift, but it takes longer in comparison to conditions implemented in a.

Interestingly in the second scenario, when mating depends on selectively neutral markers, in order for speciation to occur there has to be linkage disequilibria between ecological trait **x** and neutral markers. The opposing force in the model is free recombination. However, due to genetic drift, speciation still occurs.

Another example of incorporating genetic data into ABMs was performed by Pertoldi in 2004 [70] who used the ALMaSS [71] modelling system. A landscape and environmental conditions varied, in order to bring the population to a bottleneck (95% population mortality). Each individual carried 32 alleles in 16 pairs. Alleles were passed to the offspring randomly at each locus. The initial population consisted of

1,000 animals randomly placed on the map, and the length of one run was 2,000 years, discretized to one day steps. The values of census size, effective population size and several other coefficients were monitored during the last 500 years. Ultimately, the model was able to reflect decreases in effective population size, and expected and observed heterozygosity after bottlenecks.

One of the latest examples of an ABM implementation, where agents incorporated genetic information, was published in 2011 by Yamaguchi [72]. This study of food web stability described a food web on two trophic levels, embedding *S* species. At the beginning of the simulation, the number of prey and predators were equal. Predators were simulated as agents. Each predator contained a set of loci, which determined consumption of prey species. Generations of predators were discrete and mutations in predators caused them to switch from one prey species to another after mutation reached a certain frequency in the population. The more predators consumed, the more offspring they could leave. In parallel to selection, the allele frequency was also influenced by genetic drift. The number of species varied from 4 to 8. Dominant-loci, recessive-loci and no evolution models were tested. In the dominant loci model, individuals with the genotype 11 or 10 at G_j ate j. In the recessive-loci model, only individuals with 11 were allowed to eat prey j. Overall, the results demonstrated that the stability of (proportion of species that survive) food webs decreases with an increase in the complexity of the model. In the dominant loci model, evolution resulted in extinction. Also, increasing the mutation rate increased the stability of the system.

During the last decade, there were also many examples of ABM implementations within the Life Sciences domain that simulate movement and social interactions within populations. For example, Robbins implemented an ABM approach to simulate population dynamics and a social structure of mountain gorillas [73], Stroud used ABM to simulate the spread of influenza [74], Shwarzkopf predicted the movements of tropical toads in Australia during the wet season [75], Wood studied the evolution of movement of selfish herds [76], and Phoetke studied population dispersal [77].

Apart from population modelling, another trend in ABM arose after the year 2000, as researchers started to apply ABM to model the formation of biological tissues, cells, intracellular signalling and molecular pathways. The research papers which fell in this category up to 2009 were summarized by An [78].

In 2007, Folcik [79] developed a model of an immune system, which consisted of two main compartments (parenchyma, secondary lymphoid tissue), and blood/lymph circulation between them. Immune system cells were considered as agents.

Another example of an ABM modelling approach was published by Segovia-Juarez [80] focusing on cellular levels of tuberculosis granuloma formation. In his model, the environment represents a part of alveolar tissue, while agents were represented by macrophage and T cell agents. Chemokine and extracellular bacteria are represented by concentration variables. Chemokine sources are infected and/or activated macrophages. Secreted chemokine attracts other macrophages and T-cells leading to the granuloma formation. Necrosis was also implemented, by calculating how many times T-cells killed an infected macrophage and how many times macrophages

(assuming that reactive oxygen species released during both processes) burst within a particular section of the environment. By varying parameters of the model, researchers were able to reproduce three outcomes: clearance, small and growing solid granulomas, and large and necrotic granulomas.

One of the latest examples of cellular ABM modelling was published in 2012 by Macklin [81]. He used patient histopathology and tomography data to calibrate a model of ductal carcinoma *in situ* (an invasive form of breast cancer). The model was able to replicate patient tumour ductal growth and necrotic cell lysis.

In conclusion, it is possible to say that the popularity of ABM approaches has increased over time. I propose that the main reason behind this is that it is more intuitive and much easier to implement in comparison to equation-based modelling approaches. Another advantage, crucial for the Life Sciences, of ABM in comparison to the equation-based modelling is that it can account for individual variance. Even though ABM is a general modelling approach, the main disadvantage is its computational performance when large numbers of agents are modelled (for example, a concentration of matter in a mixture). However, in some cases, two approaches can be combined, and an environment variables of an AB model (e.g., nutritional medium) used by agents can be set up using anequation-based approach.

2.4 Agent Based Modeling Software Review

There are numerous options for frameworks and programs for Agent Based Modeling. In this section, I will provide a review of the most popular. In all cases, there were five key properties I was looking for: execution speed, ability to run simulations without a graphical user interface, the quality of documentation, a programming language used for model description, and the number of publications in which the program or framework was used. In my case, the option to detach the graphical interface is critical, because it not only allows running simulations on an external computational cluster, but also provides a way to automate the process. In its turn, automating the process is necessary because of the stochastic nature of ABM approach, which means that thousands of simulations are necessary to infer realistic values in complex processes. Also, running the simulation without graphical output can dramatically increase the execution speed.

One of the oldest and most well-established programs to perform ABM is **NetLogo** [82]. One of the main advantages of **NetLogo** is its simplicity of use, as among other applications, it is used to introduce scholars to programming. The Logo programming language dialect is used to describe a model. It is also best documented in comparison to other programs and frameworks. There is an extensive list of publications, in which NetLogo was applied to many different fields, including, but not limited to, Economics, Biology and Social Sciences. The first version of the program was published at 1999. The disadvantages of **NetLogo** are: its execution speed, unavailability of the source code, the non-detachable graphical interface and the program specific programming language. The user does not have any access to the graphical interface and is only allowed to describe the behavior of agents and the environment. Programming restrictions of **NetLogo** are that code has to be written in

one file and there is no IDE (integrated development environment) with Eclipse-like debugger available [83].

The second popular option is **Repast-Simphony**[84]. The advantages of this package are its extensive documentation, multiple language support (ReLogo, Groovy, Java, C++), a module for creation of high performance and distributed models [85]. It is more flexible than **NetLogo.** The main disadvantage of the package is that it is strongly attached to the Eclipse IDE, which leads to configuration problems on different platforms (Windows, Linux and Macintosh). Also, it tends to be a little bit slower than **MASON**, but faster than **NetLogo** [83].

The third popular option is **MASON** [86]. **MASON** was first published in 2003. It is basically a Java library containing prewritten classes, which set up a pattern for a model creation. In comparison to all software listed above, **MASON** focuses on execution speed. It replaces some of the built-in Java classes, which have poor performance with big data. The graphical interface is built on top of the model and can be easily detached (Fig. 2.3.1.). It also allows parallelization of a simulation run, using internal Java thread management tools. **MASON** is platform independent and the only requirement to run a simulation is presence of a Java Virtual Machine. It is actively used and well documented. There is a **GeoMason** module, which adds support for geographical data. The main disadvantage of **MASON** is that it requires knowledge of Java and Object Oriented Programming (OOP) principles, therefore in comparison to **NetLogo** and **Repast**, it can be hard to use for beginners.

After trying all software solution described above, I decided to use **MASON** to create my model. The reasons why I decided to choose **MASON** are the following:

1) Execution speed
2) Ability to run a model without the graphical interface
3) Reasonable amount of documentation, including tutorials
4) Numerous publications using the framework
5) Ability to incorporated 3rd party libraries
6) Potential for parallelization and complex visualization of the model
7) Integration with Eclipse IDE, allowing the use of a debug mode

A bit outdated, but still useful review comparing **MASON**, **Repast**, **NetLogo** and some others was published by Railsback in 2006 [83]. Summarizing the results of the comparison, it is possible to sort the software described above by the execution time in the following way: **Mason** < **Repast-Simphony** < **Netlogo**.

Several other options such as **Breve** [87], **FLAME** [88] and **SPARK** [89] were also considered.

Breve is a Python based ABM environment and was rejected due to a lack of support (latest version dated 2008) and the lack of publications in peer reviewed journals.

FLAME is a C++ framework for ABM. Even though it has numerous publications, good execution speed and very strong visualization options, it was rejected because of its complexity, as it would require significantly more time to create a model, as well as insufficiency of documentation.

SPARK is an ABM environment which allows the creation of models using the Logo-based SPARK-PL language. This software was rejected because of its sparse documentation, domain specificity (mostly used to model processes beyond the individual level, e.g., cell signaling), software specific language and lack of publications in comparison to mainstream ABM toolkits.

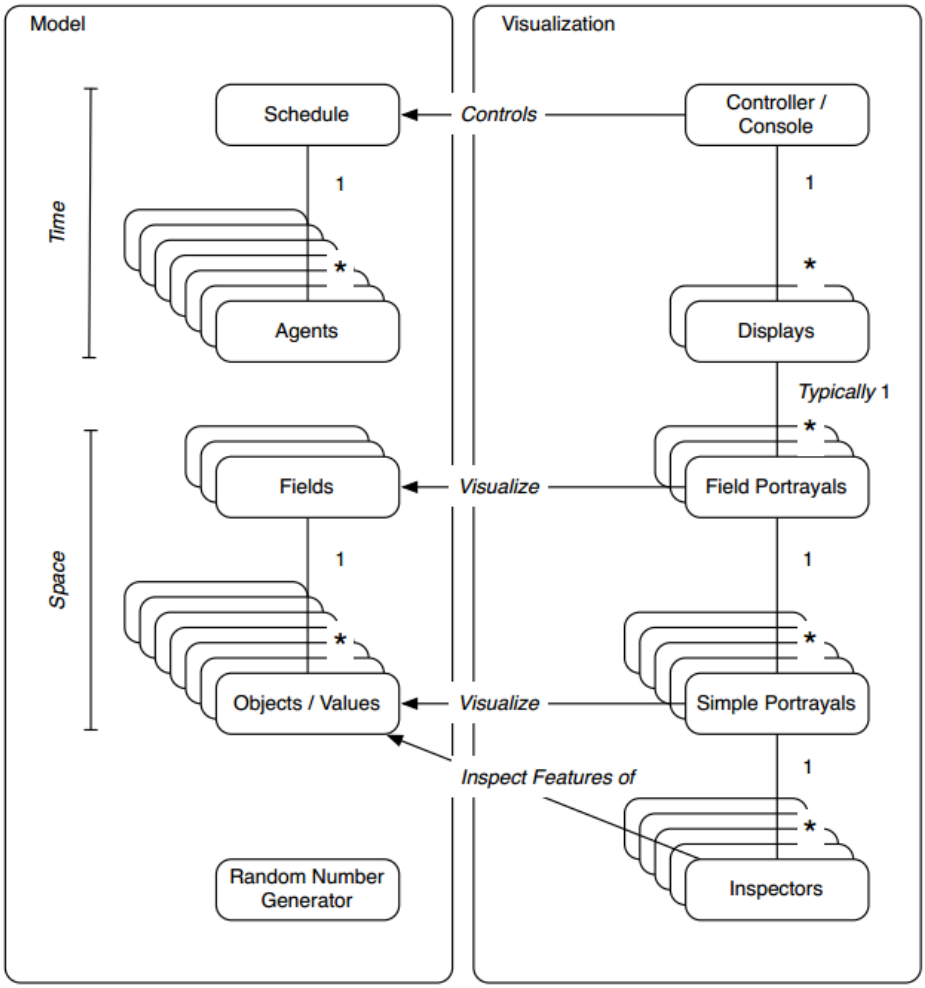

Figure 2.3.2: MASON architectural layout. Modeling and visualization are separated, which allows the model to run on external computational clusters. Scheme taken from MASON manual.

2.5 Spatial Coalescent Modeling

After determining parameter sets and their values that allow ABM to reproduce the cline in Asian ancestry observed in the data, it would be possible to generate molecular diversity more accurately utilizing coalescent theory. This is in order to check and tune parameters and their values, both of which are from the agent-based model. Comparison of simulated data with actual genetic data gathered across the region is highly beneficial. It enables us to infer past demographic processes and tune their values. Generation of genetic markers matching real-life data is the strength of

coalescent theory. Statistical inference techniques could then be used to compare with the data (see Section 2.6).

Coalescent theory was developed by Kingman in 1982 [90] and many others [91]. Kingman followed the assumptions of the Wright-Fisher population model. Under this model, mating is random. This assumption allows the construction of a stochastic genealogy tree that extends back to the Most Recent Common Ancestor (MRCA) or, in other words, until the lineages coalesce. Once the genealogical tree is constructed, mutations may be placed on it, immediately generating the molecular variation of the final population.

There are several programs currently available to generate SNP data under various assumptions of demographic conditions [92]. However, **SPLATCHE2** [11] is the only coalescent simulator to date that allows the incorporation of the actual landscape. In accordance with coalescent theory, **SPLATCHE2** performs two steps to generate diversity. On the first step, also called the backwards step, it builds a phylogenetic tree, taking into account the geographical distribution of demes, migration, deme population size and admixture between populations. A deme's population capacity can be changed in every step to reflect environmental fluctuations. On the second step, also called forward step, using the obtained tree, **SPLATCHE2** outputs genetic markers for each individual in **ARLEQUIN** [93] format. **ARLEQUIN** is a widely used format for genetic data and can be easily converted to match our dataset. **PGDSpider** [94] can be used for the conversion.

2.6 Statistical Inference

Modeling results obtained from the coalescent simulation could be compared with real-life data. Derivation of the likelihood function, describing the relationships between probability distribution parameters, does not seem achievable. This is because of the complexity and stochastic nature of the system. Therefore, I plan to utilize the Approximate Bayesian Computation technique for comparison between artificially generated and real-life data.

Approximate Bayesian Computation (ABC) is a widely used statistical inference technique within the population genetics field. The first publication describing the method was published in 1997 by Tavare [95] and extended by others [96]. The main principle behind ABC is running a sufficiently large number of simulations varying underlying model parameters and comparing them to real-world data. Measuring the level of difference between simulated and real-world data usually requires a reduction of the dataset. The reduction of the dataset is done by calculation of summary statistics. Thus, if the difference between simulated and real-world data is too large, the parameter value is rejected. Several different models can be compared in the same way. Basic principles and pitfalls of the ABC approach were recently reviewed [97]. The main problem of using summary statistics is loss of information. Several reduction techniques are available at the moment [98]. An incorrect choice of the reduction technique can dramatically affect inferred results. Using several different summaries can increase the precision (for example, F_{ST} and average genetic ancestry admixture proportions within a population). However, an increase in the

dimensionality of summary statistics causes a decrease in acceptance probability of model parameters [99].

Several libraries for Python and R [100-102] are available at the moment. More detailed reviews and performance comparisons will be needed.

3. Ancestry of Modern Pacific Populations

3.1 Frappe and Admixture Results

The procedure for finding differences in allele frequencies between subpopulations due to differences in ancestry and/or geographical isolation is known as population stratification. **STRUCTURE** is the most widely used software to perform population stratification analysis. It utilizes the Bayesian approach. The main disadvantage of the program is its unacceptable computational speed: our dataset (576 samples and 500,000 SNPs) took more than two months to run on the IMBS1 server. **STRUCTURE** requires its own input data format. Conversion of the original dataset was done using the **PGDSpider2** program. It required 10 hours of computational time and utilization of Massey University computational facilities.

Due to poor runtime, I instead turned to alternative software with better runtime properties. I chose **FRAPPE** and **ADMIXTURE**. They are based on Maximum Likelihood estimation, but utilize different algorithms. One **FRAPPE** run took 7-8 hours, whereas **ADMIXTURE** performed faster by roughly five times. Another advantage of both programs is that they do not require specific input formats and accept standard **PLINK** *.ped files [103]. Plink is most widely used program for analyzing SNP data and provides data management, summary statistics and many other tools. The advantage of *.ped files is that they are lightweight. Our dataset in a plain text format used 1.5 Gb of storage space, while the dataset in *.ped format used about 150 Mb.

Parameter K reflects our assumptions on a number of ancestral populations for an individual. After the choice of program was made, I conducted several runs varying K from 2 to 6. Obtaining results for different numbers of ancestral population was needed to test if our hypothesis of two ancestral populations in the region holds.

After the runs were completed, I assigned geographical coordinates into output for each sample. This reflects the location where samples were collected. Thus, the columns of the resulting table, in the case of $K = 2$, were: unique sample name, Asian ancestry proportion, Melanesian ancestry proportion and coordinates of an individual's population. **R** was used to plot the results (Fig. 3.1.1). All samples were sorted by latitude, because latitude is a good proxy for the Asian-Melanesian admixture cline. Latitude values were rescaled using a Python script in order to put Polynesian samples at the end of the x-axis. The results are consistent with previous findings, as it is possible to see the admixed origins of modern Pacific populations. However, this use of high-density SNP chips allowed us to generate improved estimates of admixture proportions for some Pacific populations previously studied by Cox in 2010 [2].

FRAPPE and **ADMIXTURE** results were consistent with the 2010 paper [2] by Cox. Utilization of high-density SNP data allowed the adjustment of admixture proportions for some populations, but for several populations, admixture proportions matched almost exactly (Fig. 3.1.1).

Further, it was possible to replicate the steep cline in Asian ancestry proportion in dependence to geographical latitude. Asian ancestry proportions were averaged, and plotted according to geographical latitude. To estimate the relationship between Asian ancestry proportion and latitude, I utilized a local regression method. The local regression is based on a weighted least squares method. A polynomial function was fitted locally by assigning more weight to neighboring variable values (Fig. 3.1.2).

Admixture proportions for *K* from 3 to 6 are provided in supplementary materials. I only discuss $K = 2$ results here because all hypothesis so far suggest two ancestral populations (Asian and Melanesian). **ADMIXTURE** provides a method for best *K* selection by calculating the error rate for each number of *K*. I calculated error rates for each *K* from 2 to 6. Error rates were almost identical. For $K = 2$, the error rate was higher than for $K = 3$ to 6, but a difference between rates differed on the order of 0.001 (supplementary materials, Figure S1). I connect this with a presence of the Philippines Aeta samples in our dataset. The modern Aeta population is assumed to be a descendent from the Negrito population, which is why it appears to be a distinct cluster for *K* greater than two. Our results are additional evidence supporting the Bellwood model. Under assumptions of this model, there were two migration waves. First, the region was settled by Melanesians around 50,000 years ago. Then about 5,000 years ago, mainland China and Taiwan Asian populations started to expand into the region. Thus, as I can see from previous work and our study that individuals are admixed. Another popular settlement model was suggested by Oppenheimer. The Oppenheimer model suggests that modern population resulted because of the geographical isolation of one parental population within Indonesia. If the Oppenheimer model was true, I would not be able to identify as many admixed individuals. In other words, different populations would appear as different clusters on the bar plots. Also, error rates for $K=2$ would be significantly higher than for bigger numbers of *K*. However, both models fail to explain the origin of Australian aboriginals, as some studies of non-autosomal DNA indicated that Australian aboriginals do not share mtDNA lineages with Asian or Melanesian populations.

Averaging Asian ancestry proportion per population and fitting the polynomial function allowed us to replicate the steep cline in Asian ancestry across the Pacific region described in the 2010 Cox [2] paper. Ancestry proportion is consistent with existing theories. The Bellwood model suggests the admixed origin of modern Pacific populations, which can be clearly seen on Figures 3.1.1 and 3.1.2. An increase in the Asian ancestry proportion in Polynesia is also consistent with the results of Wollstein in 2010 [8]. Polynesian samples show a high Asian ancestry component and Papuan samples appeared as a distinct cluster, with low levels of Asian ancestry.

3.2 MASON Results

Reasons for selecting **MASON** as a framework of choice are stated in Section 2.4. Briefly, I selected **MASON** as software of choice because of its computational performance, flexibility, in terms of incorporating third party libraries and complete separation of the computational part from a graphical user interface.

I have performed several iterations of the model logic. All prototypes can be summarized into three groups:

1) Prototypes with individuals as agents
2) Prototypes with demes as agents
3) Mixed prototypes

In an early prototype, the model consisted of individuals, which moved randomly across the map. In order to provide a background for mating and migration, I tried to add an interaction network. Networks can be described as undirected graphs. The vertices of the graph are individuals, while edges are relationships between individuals. A social interaction graph was initially instantiated at the beginning of the run. I planned to add an additional vertex to the graph for each newborn, and delete it when individual reached its endpoint. However, due to problems described below, I did not code it. Each edge of the graph had its weight, which reflected relationships between the two vertices it connected. Most edges specified neutral relationships. However, some specified negative and positive relationships. The negative relationship caused individuals to move away from each other, while positive relationships caused the exact opposite. Overall, the movement vector resulted from a force to the closest attraction point, location of "enemies", location of "friends" and a small random movement component. At the beginning, the idea was to mate close individuals if the friendship weight exceeded some certain threshold. With such a movement specification, some unlucky individuals were forced to the edge of the map (Fig. 3.2.1). Then I added attraction points onto the map. Attraction points were coordinate points on the map. The addition of these caused individuals to crowd around the attraction points. Crowding around the attraction points was supposed to replicate the deme's population. The attraction force increased gradually with distance from an individual to the point. Each individual chose a point closest to its location as its attraction point.

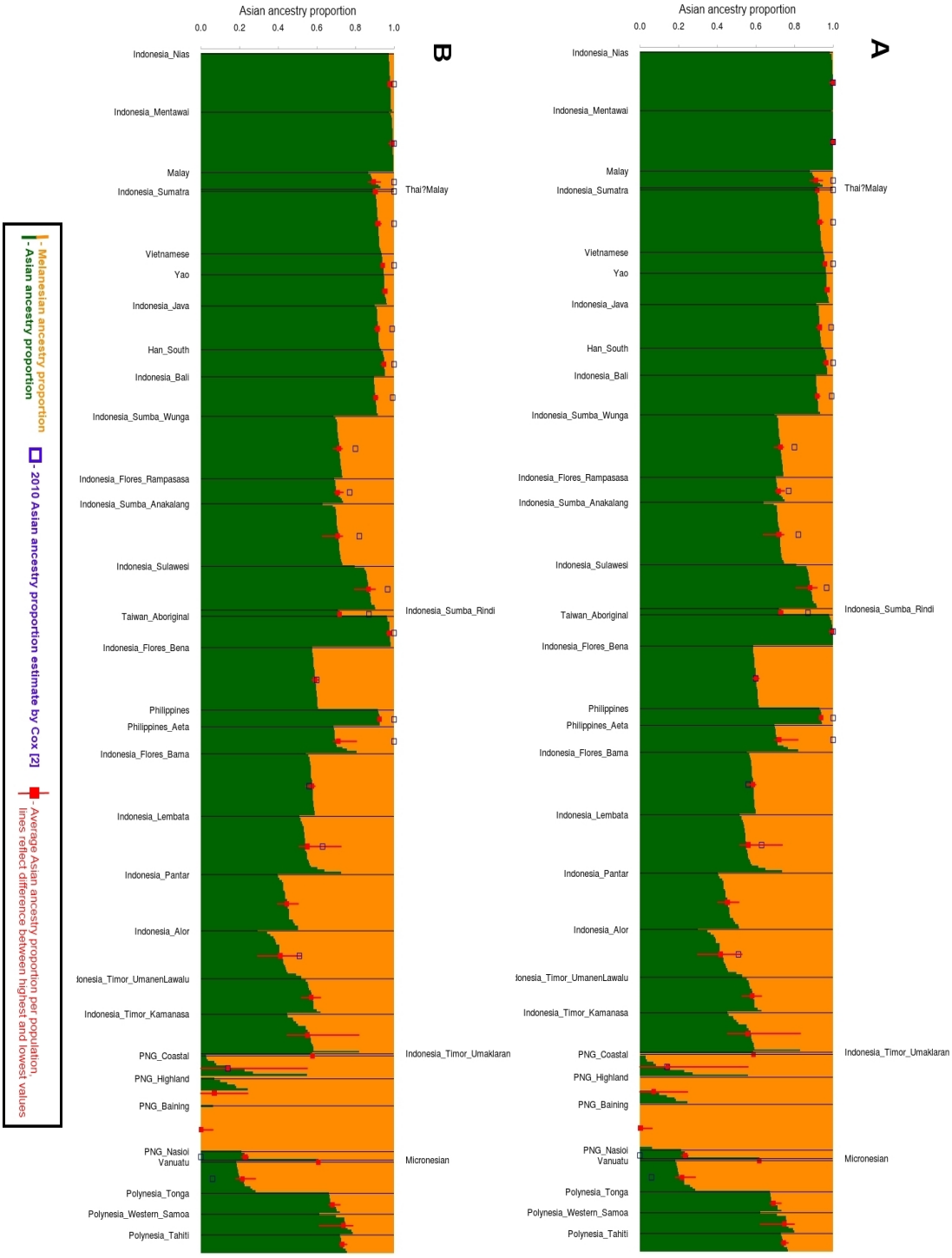

Figure 3.1.1: Ancestry admixture proportions for modern populations in the Pacific region. Abbreviations: PNG – Papua New Guinea. The assumed number of ancestral populations is equal to two (K = 2). Populations are sorted by geographical latitude, from west to east. For each population, samples are sorted by the increase in Asian ancestry fraction. A – ADMIXTURE results; B – FRAPPE results.

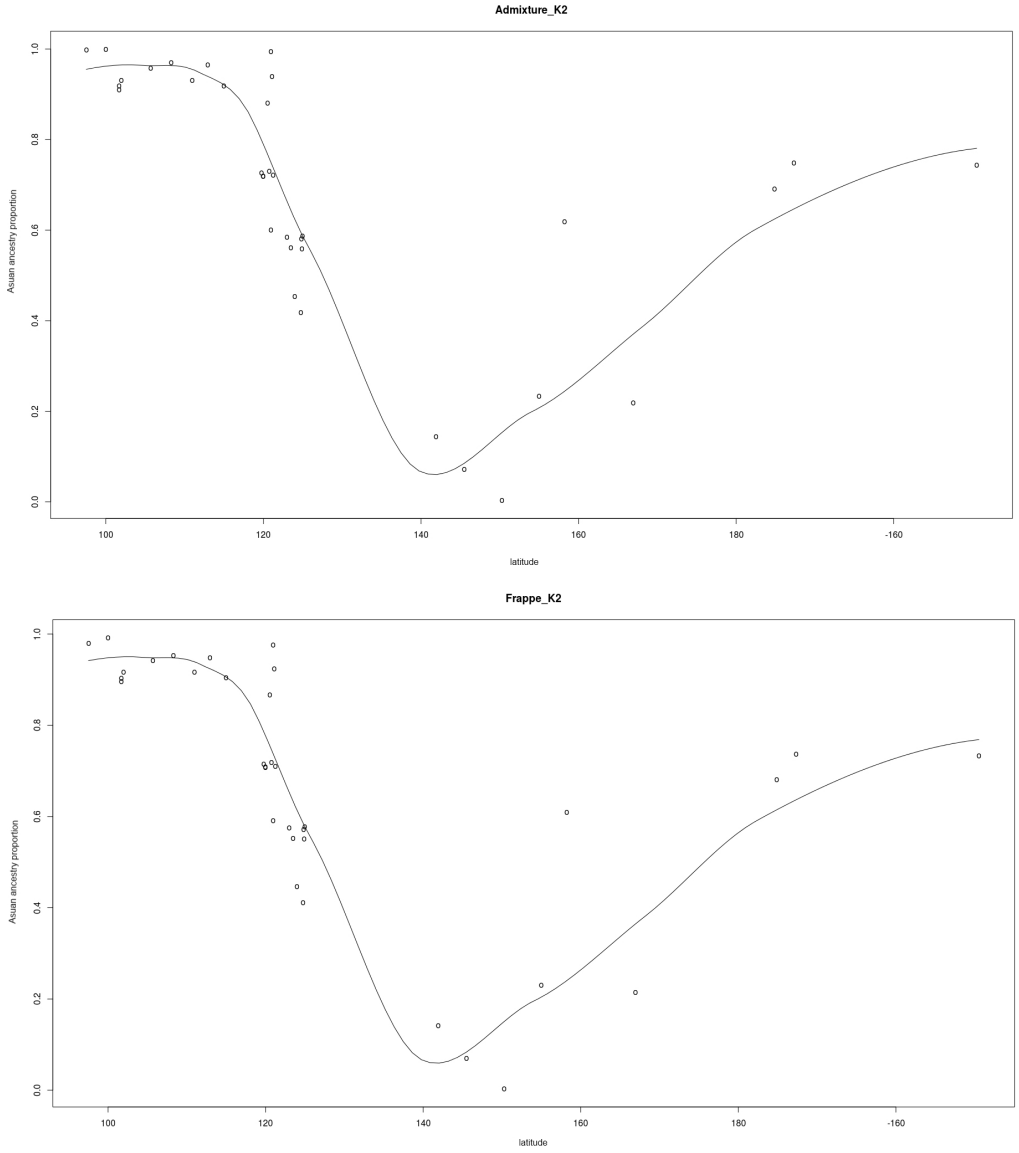

Figure 3.1.2: Average Asian ancestry proportion per population in relation to latitude. It is possible to see that the Asian ancestry proportion falls rapidly near 130 degree of latitude, or in other words, near the Indonesian island of Alor. The Asian ancestry proportion goes up at latitude corresponding to Polynesia because this region was settled during the last Asian migration wave. The migration process started approximately five thousand years ago, while the Polynesian region was settled from around 1,500 BC.

The next prototype did not include a social network. Instead I tried to implement migration between demes. I did this by allowing individuals to switch between the attraction points. Just as previously, individuals chose the closest attraction point, but could randomly switch between them. In addition, I added new individuals to the map periodically. This migration pattern caused an overwhelming majority of the "world" population to migrate constantly and in an unmanageable manner (Fig. 3.2.2). I could not locate the source of the problem and decided to try different design.

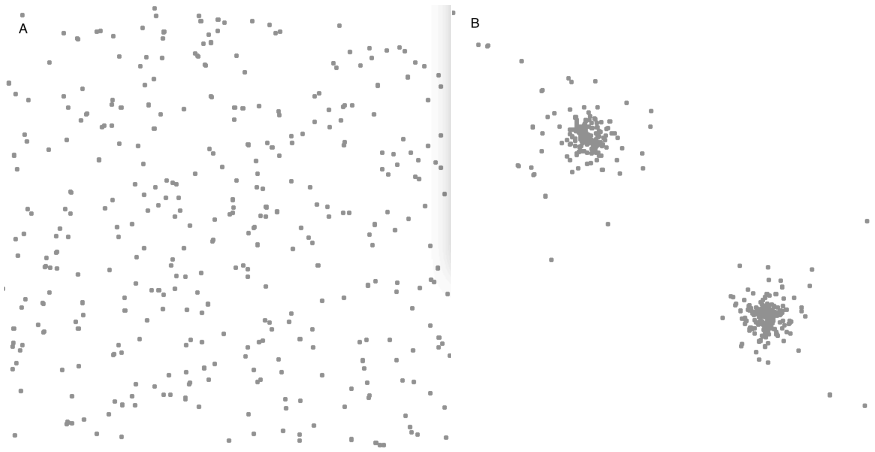

Figure 3.2.1: Visual representation of the model prototype with a social network. A – beginning of the simulation. Individuals placed randomly on the map. B – individuals chose their attraction points and clustered around them. It is possible to see outliers for which the attraction point vector is canceled by negative relationships with individuals surrounding the point.

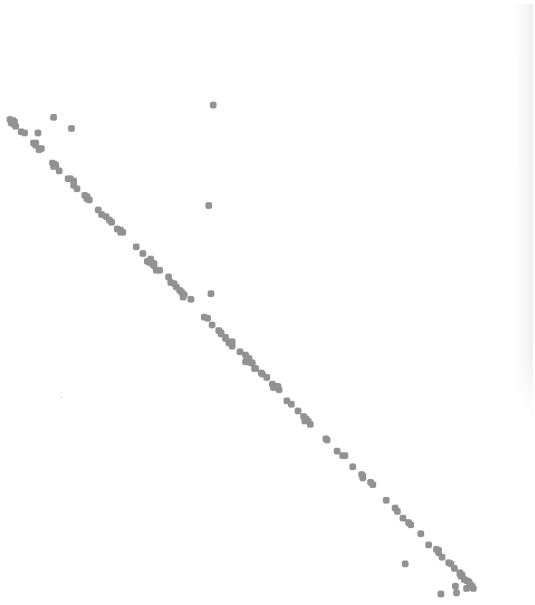

Figure 3.2.2: Visual representation of the prototype with migration. The overwhelming majority of the "world" population is in the migration process. Outlier individuals are newly added agents, which are moving to the closest attraction point. Under this model design, I faced difficulties in migration management and therefore decided to try a different design.

Eventually it turned out that the specification of complicated movement patterns leads to problems in parameter value selection. It was impossible to select an appropriate movement speed corresponding to the time step of the model because, at any given time, a significant proportion of the map population was in the migration process. Also, it was unclear what time (day, month, year) to assign to each time step of the model. Facing this and similar difficulties, I decided to not include individuals and their movement into the next set of prototypes.

I then tried a model design where agents were represented as demes scattered across the map. Each deme contained a collection of arrays. Each collection represented the deme's population, including an individual haploid strand with SNPs. At each step, a certain percentage of the population mated and produced offspring. The mating function randomly picked two individuals from the population, walked along their SNP strand and formed an associated SNP strand of a child. The number of children per pair was random and varied from one to three. After producing children, parents were removed from the population. Mating a certain fixed percentage of the population caused consistent growth of the deme's population, leading to a crash of the simulation run, or population extinction. The reason behind such behavior turned out to be a rounding error. I solved this problem by fixing the number of mating pairs per deme per time step. I also found that varying the number of mating pairs per step changes the deme population size. For instance, fixing the number to eight mating pairs per year caused population size to fluctuate around five hundred individuals. I chose this number based on the study of Lansing 2008 [104].

On a deme level, I was able to stabilize population size and implement mating. However, I was unable to implement a migration processes between the demes. The problem was that in order to implement migration, I had to modify the contents of one deme from another. This operation contradicts basic Java and ABM principles. Java is an Object Oriented Programming (OOP) language. One of the basic principles of the OOP approach is encapsulation of objects. Object properties are defined in special class files. Process of creating an object from a class called instantiating. Encapsulation means restricted access to objects internal data. Ideally, communication between objects and all modifications should be done via special get and set functions. However, when there are numerous instances (objects) of one class communication between objects becomes complicated and requires manual specification of communication procedures. In our case, I could not get communication working between demes. Demes were instances of one class. In addition, ABM principles, regardless of programming language used, state that agents should be self-sufficient and make their own decisions based on the environment conditions. In other words, the more independence an agent has, the better. Thus, I decided to change the design again, instead of investing time in a workaround development.

Next, I decided to merge the two approaches described above. I specified two identical maps. A map is a field with a coordinate system, which holds the agents. The first map contains individuals, while the second map contains demes. The location of individuals on map one is determined by deme coordinates. For each deme, there is only one coordinate point where all individuals belonging to the deme are stacked.

Each step in the model corresponds to one year in real life. Individuals "decide" when to migrate and die (Fig. 3.2.1). At each step, each individual has 0.05 probability of migrating. If the migration condition becomes true, the individual selects another deme in the map to migrate to. The migration probability of 0.05 is our assumption and could be fitted to real data at some later time. I assume social selective advantage of individuals of Asian ancestry. Each individual carries a boolean flag, reflecting if he had at least one Asian ancestor. Theoretically, an individual can carry no Asian SNPs, but the flag can still be true. However, the probability of getting such an

individual is unlikely. While Melanesian descendants chose only the closest deme to migrate to, Asian descendants gather coordinates of the four closest demes and randomly select one to migrate to. Each individual has an age counter. When the counter value exceeds twenty five, the individual dies. Human generation time estimates vary between 15 to 30 [105, 106] years. The number 25 was chosen for testing purposes, and could be changed to a mortality curve later. Each individual carries one thousand independent SNPs.

One function of demes is the mating of individuals (Fig. 3.2.1). At each step, each deme gathers copies of individuals from the same place on map one and randomly mates them. I fixed the number of mating pairs to eight per year as this allows the population of each deme to fluctuate around five hundred. Five hundred is our assumption of village size based on the Lansing 2008 publication [104]. The mating function randomly picks two individuals, walks along their independent SNPs and at each SNP position randomly selects SNP from individual one or individual two. Offspring is then immediately added to the map according to the deme's coordinates. This model assumption is subject to discussion and potentially can be changed in the future. Another function of the deme is the calculation of admixture proportions. I represent SNPs of Asian origin as ones and SNPs of Melanesian origin as twos. On each step, each deme calculates the proportion of SNPs of Asian origin according to equation 3.2.1:

$$Asian\ ancestry\ proportion = \frac{\sum Asian\ SNPs\ in\ deme's\ popuation}{Population\ size * Total\ number\ of\ SNPs} \quad (3.2.1)$$

For visual representation of the model, the proportion is converted to colour according to the RGB scheme (Fig. 3.2.2). Demes are represented as circles with varying radius. The radius of the circle reflects population size. According to the RGB colour scheme, every colour can be described as a mixture of Red, Green and Blue. Each colour proportion varies from 0 to 255. In our case, I do not need Green colour, so it is always zero. If deme does not contain any Melanesian SNPs, the value would be Red = 255, Green = 0, Blue = 0. This corresponds to a bright red colour. If the deme is admixed in equal proportions, the colour code would be (127.5, 0, 127.5) which corresponds to purple. Finally, if the deme contains only Melanesian SNPs, the colour code is (0, 0, 255), which corresponds to blue. Eventually, the deme colour is set up in the following manner (eq. 3.2.2):

$$(Red, Green, Blue) = (255 * Aap, 0, 255 - 255 * Aap) \quad (3.2.2)$$

Where *Aap* stands for Asian ancestry proportion.

The source code of the model is provided in the supplementary materials, Section S1.

For testing the model, I varied the number of demes. SNP counts per individual and deme population size were always 1,000 and approximately 500 respectively. Table 3.2.1 shows the computational performance of the model. Running the model without a GUI increases the execution speed significantly.

Table 2.3.1: Computational performance of the model

Number of demes	Approximate rate (seconds) with graphical output	Approximate rate (seconds) without graphical output
20	33	33
40	17	18
80	7	8

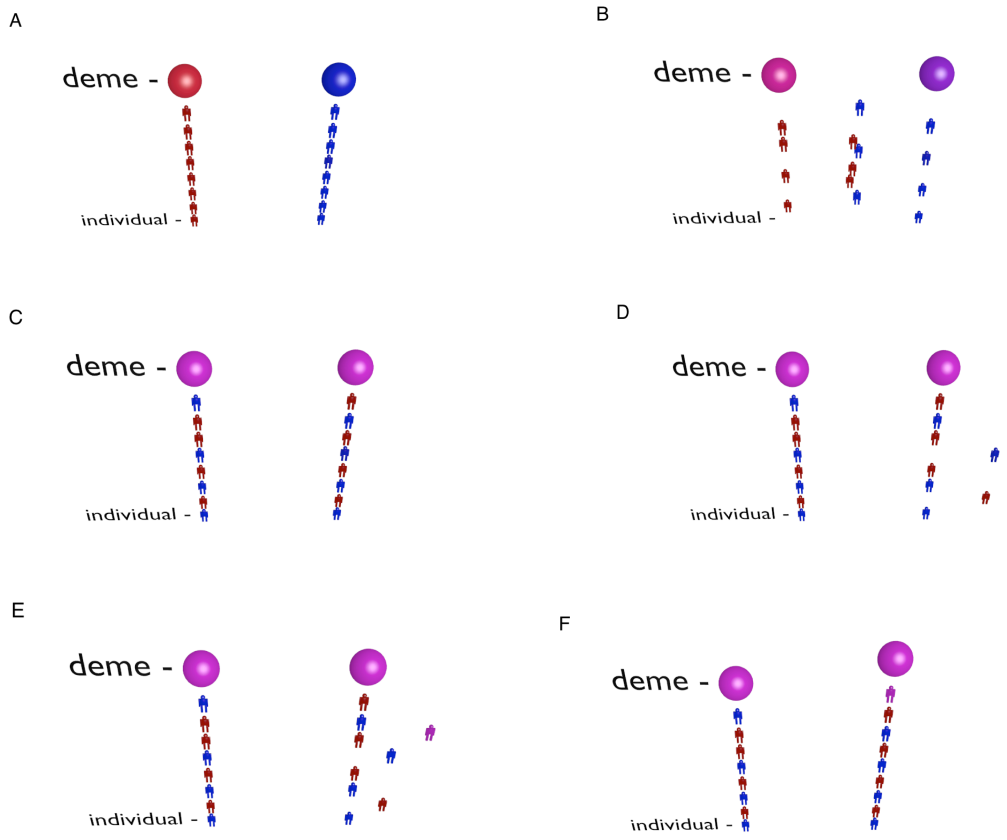

Figure 3.2.3: Agent Based Model scheme. A – beginning of the run. Two neighbouring demes. Asian – red, Melanesian – blue; B – migration process; C – migration process completed. Demes colour reflects equal proportion of Asian and Melanesian SNPs within population. D – two individuals are randomly selected to mate. E – selected individuals produced admixed child. The colour of the child reflects admixed ancestry. In this case, the child has equal proportions of Asian and Melanesian ancestry. F – deme population size increased. From top view, this would be reflected as a change in in circle radius.

Strangely, computational speed is not affected by running the model without the graphical user interface. Usually visualization is the limiting factor. I explain this by suggesting that the limiting factor is operations on a large collection of arrays representing independent SNPs.

Source code of the model is provided in the supplementary materials, Section S1.

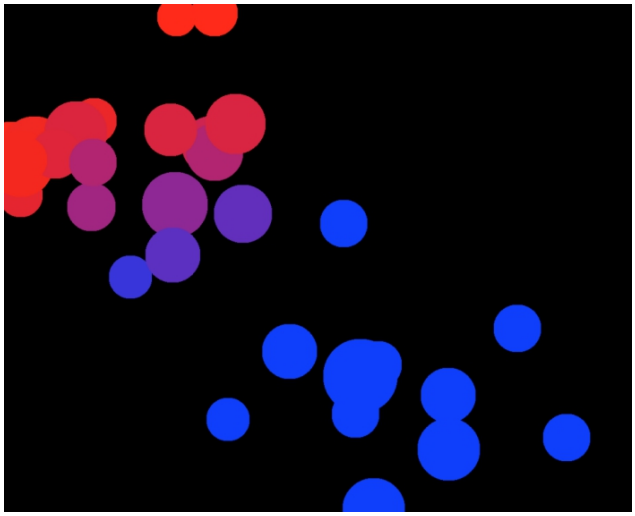

Figure 3.2.4: Visualization of the Agent Based Model. Circles correspond to deme locations. Colour reflects the genetic background. Blue –Melanesian; Red – Asian; Purple – admixed ancestry. Gradations of colour reflect changes in admixture proportions. The circle radius reflects the deme's population size.

3.3 Future Improvements of the Model

The problem of such a visualization approach is that it is sometimes hard to tell the exact admixture proportions by looking at the visual representation of the model. One direction of future work would be to incorporate a bar chart, where each bar would represent one deme, and a fraction of each ancestral population will be shown in its own colour.

At the moment, the map is split into several regions. Asian demes are initially placed to the North-West section, and Melanesian demes are placed along the South-East direction of the map. Asian and Melanesian sections have a small overlap. All demes are placed randomly within their specified sections of the map. Without incorporation of actual geography, model check and selection of the parameters can be difficult. Another possible future direction is incorporation of the actual geography of the region. Demes could be placed at roughly the same distance from each other according to the geography of Indonesia. Incorporation of actual geography of the region is necessary because it will allow different sophisticated selective advantage scenarios to be tried in terms of migration. Thus taking geography into account can dramatically affect the inference of parameters of interest. However, the exact set of parameters is not specified at the moment.

Performance is an object of concern because tens to hundreds of thousands of simulations will need to be run to infer parameters of interest, such as the migration rate, lifespan, reproductive age and various selective advantage scenarios. Different parameter layouts will need to be tried varying some and fixing (if possible) other parameters. An extensive literature review in order to fix as many parameters as possible will be needed, as with an increase in degrees of freedom decreasing the

probability of finding a parameter layout replicating the real world data. The parameter fitting process will need to be automated.

Summarizing possible future modelling tasks:

- Incorporation of actual geography
- Improving visualization
- Implementing several different selective advantage scenarios
- Specifying the exact set of parameters to fit, in order to replicate the real world data
- Inference of parameter values by running tens to hundreds of thousands of simulations, due to a stochastic nature of the ABM approach.

Bibliography

1. Cox, M.P., *The genetic environment of Melanesia: clines, clusters and contact.* Population genetics research progress, 2008: p. 45-83.
2. Cox, M.P., et al., *Autosomal and X-linked single nucleotide polymorphisms reveal a steep Asian-Melanesian ancestry cline in eastern Indonesia and a sex bias in admixture rates.* Proc Biol Sci, 2010. **277**(1687): p. 1589-96.
3. Cox, M.P., *The Genetic Environment of Melanesia: Clines, Clusters and Contact.*, in *Population Genetics Research Progress*. 2008, Nova Science Publishers, Inc: New York. p. 45-83.
4. Stephen Lansing, J., et al., *An ongoing Austronesian expansion in Island Southeast Asia.* Journal of Anthropological Archaeology, 2011. **30**(3): p. 262-272.
5. Bellwood, P.S., *First farmers : the origins of agricultural societies*. 2005, Malden, Mass. ; Oxford: Blackwell.
6. Teeter, K.C., et al., *The variable genomic architecture of isolation between hybridizing species of house mice.* Evolution, 2010. **64**(2): p. 472-85.
7. Guillot, E.G., M.K. Tumonggor, J.S. Lansing, H. Sudoyo and M.P. Cox., *Climate Change Influenced Female Population Sizes through Time across the Indonesian Archipelago.* Human Biology, 2013.
8. Wollstein, A., et al., *Demographic history of Oceania inferred from genome-wide data.* Curr Biol, 2010. **20**(22): p. 1983-92.
9. Alexander, D.H., J. Novembre, and K. Lange, *Fast model-based estimation of ancestry in unrelated individuals.* Genome Res, 2009. **19**(9): p. 1655-64.
10. Tang, H., et al., *Estimation of individual admixture: Analytical and study design considerations.* Genetic Epidemiology, 2005. **28**(4): p. 289-301.
11. Ray, N., et al., *SPLATCHE2: a spatially explicit simulation framework for complex demography, genetic admixture and recombination.* Bioinformatics, 2010. **26**(23): p. 2993-4.
12. Teshima, K.M., G. Coop, and M. Przeworski, *How reliable are empirical genomic scans for selective sweeps?* Genome Res, 2006. **16**(6): p. 702-12.
13. Csillery, K., et al., *Approximate Bayesian Computation (ABC) in practice.* Trends Ecol Evol, 2010. **25**(7): p. 410-8.
14. Saunders, I.W., et al., *A range of simple summary genome-wide statistics for detecting genetic linkage using high density marker data.* Genet Epidemiol, 2007. **31**(6): p. 565-76.

15. Wallace, A.R., *The Malay Archipelago, the land of the orang-utan and the bird of paradise : a narrative of travel with studies of man and nature / by Alfred Russel Wallace*. 1890, London: Macmillan.
16. Bais, W.J. and A.W. Verhoef, *On the Biochemical index of Various Races in the East Indian Archipelago.* The Journal of Immunology, 1924. **9**(5): p. 383-386.
17. Howells, W.W., *Physical variation and history in Melanesia and Australia.* American Journal of Physical Anthropology, 1976. **45**(3): p. 641-649.
18. Friedlaender, J.S., *Patterns of Human Variation: The Demography, Genetics, and Phenetics of Bougainville Islanders*. 1975: Harvard University Press.
19. Hanihara, T., *Comparison of craniofacial features of major human groups.* Am J Phys Anthropol, 1996. **99**(3): p. 389-412.
20. Pietrusewsky, M., *Pacific-Asian Relationships: A Physical Anthropological Perspective.* Oceanic Linguistics, 1994. **33**(2): p. 407-429.
21. Harvey, R.G., *Ecological factors in skin color variation among Papua New Guineans.* Am J Phys Anthropol, 1985. **66**(4): p. 407-16.
22. Norton, H.L., et al., *Skin and hair pigmentation variation in Island Melanesia.* Am J Phys Anthropol, 2006. **130**(2): p. 254-68.
23. Bellwood, P.S., *.Prehistory of the Indo-Malaysian Archipelago.* 1997, Honolulu: University of Hawai'i Press.
24. Diamond, J. and P. Bellwood, *Farmers and Their Languages: The First Expansions.* Science, 2003. **300**(5619): p. 597-603.
25. Blust, R., *The Prehistory of the Austronesian-Speaking Peoples: A View from Language.* Journal of World Prehistory, 1995. **9**(4): p. 453-510.
26. Oppenheimer, S., *Eden in the East : the drowned continent of Southeast Asia*. 1999, London: Phoenix.
27. Hertzberg, M., et al., *An Asian-specific 9-bp deletion of mitochondrial DNA is frequently found in Polynesians.* Am J Hum Genet, 1989. **44**(4): p. 504-10.
28. Merriwether, D.A., et al., *Mitochondrial DNA variation is an indicator of austronesian influence in Island Melanesia.* Am J Phys Anthropol, 1999. **110**(3): p. 243-70.
29. Redd, A.J. and M. Stoneking, *Peopling of Sahul: mtDNA variation in aboriginal Australian and Papua New Guinean populations.* Am J Hum Genet, 1999. **65**(3): p. 808-28.
30. Kivisild, T., et al., *The Emerging Limbs and Twigs of the East Asian mtDNA Tree.* Molecular Biology and Evolution, 2002. **19**(10): p. 1737-1751.
31. Karafet, T.M., et al., *New binary polymorphisms reshape and increase resolution of the human Y chromosomal haplogroup tree.* Genome Research, 2008.
32. Kayser, M., et al., *Independent histories of human Y chromosomes from Melanesia and Australia.* Am J Hum Genet, 2001. **68**(1): p. 173-190.
33. Kayser, M., et al., *Melanesian and Asian Origins of Polynesians: mtDNA and Y Chromosome Gradients Across the Pacific.* Molecular Biology and Evolution, 2006. **23**(11): p. 2234-2244.
34. Mona, S., et al., *Patterns of Y-Chromosome Diversity Intersect with the Trans-New Guinea Hypothesis.* Molecular Biology and Evolution, 2007. **24**(11): p. 2546-2555.
35. Karafet, T.M., et al., *Major East–West Division Underlies Y Chromosome Stratification across Indonesia.* Molecular Biology and Evolution, 2010. **27**(8): p. 1833-1844.

36. Kayser, M., *The Human Genetic History of Oceania: Near and Remote Views of Dispersal.* Current Biology, 2010. **20**(4): p. R194-R201.
37. Friedlaender, J.S., et al., *The genetic structure of Pacific Islanders.* Plos Genetics, 2008. **4**(1): p. e19.
38. Pugach, I., et al., *Dating the age of admixture via wavelet transform analysis of genome-wide data.* Genome Biology, 2011. **12**(2): p. R19.
39. Xu, S., et al., *Genetic dating indicates that the Asian-Papuan admixture through Eastern Indonesia corresponds to the Austronesian expansion.* Proc Natl Acad Sci U S A, 2012. **109**(12): p. 4574-9.
40. Gray, R.D. and F.M. Jordan, *Language trees support the express-train sequence of Austronesian expansion.* Nature, 2000. **405**(6790): p. 1052-1055.
41. Greenhill, S.J., R. Blust, and R.D. Gray, *The Austronesian Basic Vocabulary Database: from bioinformatics to lexomics.* Evol Bioinform Online, 2008. **4**: p. 271-83.
42. Gray, R.D., A.J. Drummond, and S.J. Greenhill, *Language Phylogenies Reveal Expansion Pulses and Pauses in Pacific Settlement.* Science, 2009. **323**(5913): p. 479-483.
43. Atkinson, Q.D., *The descent of words.* Proceedings of the National Academy of Sciences, 2013. **110**(11): p. 4159-4160.
44. Pritchard, J.K., M. Stephens, and P. Donnelly, *Inference of population structure using multilocus genotype data.* Genetics, 2000. **155**(2): p. 945-59.
45. Falush, D., M. Stephens, and J.K. Pritchard, *Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies.* Genetics, 2003. **164**(4): p. 1567-87.
46. Falush, D., M. Stephens, and J.K. Pritchard, *Inference of population structure using multilocus genotype data: dominant markers and null alleles.* Molecular Ecology Notes, 2007. **7**(4): p. 574-578.
47. Hubisz, M.J., et al., *Inferring weak population structure with the assistance of sample group information.* Molecular Ecology Resources, 2009. **9**(5): p. 1322-1332.
48. Liu, Y., et al., *Softwares and methods for estimating genetic ancestry in human populations.* Hum Genomics, 2013. **7**: p. 1.
49. Johansen, A.M., *Markov Chain Monte Carlo*, in *International Encyclopedia of Education (Third Edition)*, P. Editors-in-Chief: Penelope, et al., Editors. 2010, Elsevier: Oxford. p. 245-252.
50. Tang, H., et al., *Estimation of individual admixture: analytical and study design considerations.* Genet Epidemiol, 2005. **28**(4): p. 289-301.
51. Dempster, A.P., N.M. Laird, and D.B. Rubin, *Maximum Likelihood from Incomplete Data via the EM Algorithm.* Journal of the Royal Statistical Society. Series B (Methodological), 1977. **39**(1): p. 1-38.
52. Alexander, D. and K. Lange, *Enhancements to the ADMIXTURE algorithm for individual ancestry estimation.* BMC Bioinformatics, 2011. **12**(1): p. 246.
53. Von Neumann, J. and A.W. Burks, *Theory of self-reproducing automata.* 1966.
54. Gardner, M., *Mathematical Games: The fantastic combinations of John Conway's new solitaire game ``life''.* Scientific American, 1970: p. 120-123.
55. Reynolds, C., *Flocks, Herds, and Schools: A Distributed Behavioral Model.* Computer Graphics, 1987. **21**(4): p. 25-34.
56. Bajec, I.L. and F.H. Heppner, *Organized flight in birds.* Animal Behaviour, 2009. **78**(4): p. 777-789.

57. Macal, C.M. and M.J. North, *Tutorial on agent-based modelling and simulation.* Journal of Simulation, 2010. **4**(3): p. 151-162.
58. Deangelis, D.L., D.K. Cox, and C.C. Coutant, *Cannibalism and size dispersal in young-of-the-year largemouth bass: Experiment and model.* Ecological Modelling, 1980. **8**(0): p. 133-148.
59. Kimmel, M., *Does competition for food imply skewness?* Mathematical Biosciences, 1986. **80**(2): p. 239-264.
60. Łomnicki, A. and J. Ombach, *Resource partitioning within a single species population and population stability: A theoretical model.* Theoretical Population Biology, 1984. **25**(1): p. 21-28.
61. Grimm, V., *Ten years of individual-based modelling in ecology: what have we learned and what could we learn in the future?* Ecological Modelling, 1999. **115**(2): p. 129-148.
62. DeAngelis, D.L., L. Godbout, and B.J. Shuter, *An individual-based approach to predicting density-dependent dynamics in smallmouth bass populations.* Ecological Modelling, 1991. **57**(1–2): p. 91-115.
63. Datta, S.B. and G. Beauchamp, *Effects of Group Demography on Dominance Relationships Among Female Primates. I. Mother-Daughter and Sister-Sister Relations.* The American Naturalist, 1991. **138**(1): p. 201-226.
64. Haefner, J.W. and T.O. Crist, *Spatial Model of Movement and Foraging in Harvester Ants (Pogonomyrmex) (I): The Roles of Memory and Communication.* Journal of Theoretical Biology, 1994. **166**(3): p. 299-313.
65. Carter, J. and J.T. Finn, *MOAB: a spatially explicit, individual-based expert system for creating animal foraging models.* Ecological Modelling, 1999. **119**(1): p. 29-41.
66. Schank, J.C. and J.R. Alberts, *Self-Organized Huddles of Rat Pups Modeled by Simple Rules of Individual Behavior.* Journal of Theoretical Biology, 1997. **189**(1): p. 11-25.
67. Kohler, T., *Agent-based modeling of Anasazi village formation in the northern American Southwest*. 1995.
68. Henein, K., J. Wegner, and G. Merriam, *Population Effects of Landscape Model Manipulation on Two Behaviourally Different Woodland Small Mammals.* Oikos, 1998. **81**(1): p. 168-186.
69. Dieckmann, U. and M. Doebeli, *On the origin of species by sympatric speciation.* Nature, 1999. **400**(6742): p. 354-357.
70. Pertoldi, C. and C. Topping, *The use of agent-based modelling of genetics in conservation genetics studies.* Journal for Nature Conservation, 2004. **12**(2): p. 111-120.
71. Topping, C.J., et al., *ALMaSS, an agent-based model for animals in temperate European landscapes.* Ecological Modelling, 2003. **167**(1): p. 65-82.
72. Yamaguchi, W., M. Kondoh, and M. Kawata, *Effects of evolutionary changes in prey use on the relationship between food web complexity and stability.* Population ecology, 2011. **53**(1): p. 59-72.
73. Robbins, M.M. and A.M. Robbins, *Simulation of the population dynamics and social structure of the Virunga mountain gorillas.* American Journal of Primatology, 2004. **63**(4): p. 201-223.
74. Stroud, P., et al., *Spatial dynamics of pandemic influenza in a massive artificial society.* Journal of Artificial Societies and Social Simulation, 2007. **10**(4): p. 9.

75. Schwarzkopf, L. and R.A. Alford, *Nomadic movement in tropical toads.* Oikos, 2002. **96**(3): p. 492-506.
76. Wood, A.J. and G.J. Ackland, *Evolving the selfish herd: emergence of distinct aggregating strategies in an individual-based model.* Proceedings of the Royal Society B: Biological Sciences, 2007. **274**(1618): p. 1637-1642.
77. Poethke, H.J., B. Pfenning, and T. Hovestadt, *The relative contribution of individual and kin selection to the evolution of density-dependent dispersal rates.* 2010.
78. An, G., et al., *Agent-based models in translational systems biology.* Wiley Interdisciplinary Reviews: Systems Biology and Medicine, 2009. **1**(2): p. 159-171.
79. Folcik, V., G. An, and C. Orosz, *The Basic Immune Simulator: An agent-based model to study the interactions between innate and adaptive immunity.* Theoretical Biology and Medical Modelling, 2007. **4**(1): p. 39.
80. Segovia-Juarez, J.L., S. Ganguli, and D. Kirschner, *Identifying control mechanisms of granuloma formation during M. tuberculosis infection using an agent-based model.* Journal of Theoretical Biology, 2004. **231**(3): p. 357-376.
81. Macklin, P., et al., *Patient-calibrated agent-based modelling of ductal carcinoma in situ (DCIS): From microscopic measurements to macroscopic predictions of clinical progression.* Journal of Theoretical Biology, 2012. **301**(0): p. 122-140.
82. Wilensky, U. *NetLogo, http://ccl.northwestern.edu/netlogo/.* 1999.
83. Railsback, S.F., S.L. Lytinen, and S.K. Jackson, *Agent-based simulation platforms: Review and development recommendations.* SIMULATION, 2006. **82**(9): p. 609-623.
84. North, M., et al., *Complex adaptive systems modeling with Repast Simphony.* Complex Adaptive Systems Modeling, 2013. **1**(1): p. 1-26.
85. Collier, N. and M. North, *Parallel agent-based simulation with Repast for High Performance Computing.* SIMULATION, 2012.
86. Luke, S., et al., *MASON: A Multiagent Simulation Environment.* SIMULATION, 2005. **81**(7): p. 517-527.
87. Klein, J., *breve: a 3D environment for the simulation of decentralized systems and artificial life*, in *Proceedings of the eighth international conference on Artificial life*. 2003, MIT Press. p. 329-334.
88. Holcombe, M., S. Coakley, and R. Smallwood. *A General Framework for agent-based modelling of complex systems*. in *Proceedings of the 2006 European Conference on Complex Systems*. 2006.
89. Solovyev, A., et al., *SPARK: A Framework for Multi-Scale Agent-Based Biomedical Modeling*. 2010, IGI Global. p. 18-30.
90. Kingman, J.F.C., *The coalescent.* Stochastic Processes and their Applications, 1982. **13**(3): p. 235-248.
91. Hudson, R.R., *Gene genealogies and the coalescent process.* Oxford surveys in evolutionary biology, 1990. **7**(1): p. 44.
92. Carvajal-Rodríguez, A., *Simulation of genomes: a review.* Current genomics, 2008. **9**(3): p. 155.
93. Excoffier, L., G. Laval, and S. Schneider, *Arlequin (version 3.0): an integrated software package for population genetics data analysis.* Evolutionary bioinformatics online, 2005. **1**: p. 47.

94. Lischer, H.E.L. and L. Excoffier, *PGDSpider: an automated data conversion tool for connecting population genetics and genomics programs.* Bioinformatics, 2012. **28**(2): p. 298-299.
95. Tavare, S., et al., *Inferring coalescence times from DNA sequence data.* Genetics, 1997. **145**(2): p. 505-518.
96. Beaumont, M.A., W. Zhang, and D.J. Balding, *Approximate Bayesian computation in population genetics.* Genetics, 2002. **162**(4): p. 2025-2035.
97. Sunnåker, M., et al., *Approximate bayesian computation.* PLoS computational biology, 2013. **9**(1): p. e1002803.
98. Blum, M., et al., *A comparative review of dimension reduction methods in approximate Bayesian computation.* Statistical Science, 2013. **28**(2): p. 189-208.
99. Csilléry, K., et al., *Approximate Bayesian Computation (ABC) in practice.* Trends in Ecology & Evolution, 2010. **25**(7): p. 410-418.
100. Csilléry, K., O. François, and M.G.B. Blum, *abc: an R package for approximate Bayesian computation (ABC).* Methods in Ecology and Evolution, 2012. **3**(3): p. 475-479.
101. Liepe, J., et al., *ABC-SysBio—approximate Bayesian computation in Python with GPU support.* Bioinformatics, 2010. **26**(14): p. 1797-1799.
102. De Mita, S. and M. Siol, *EggLib: processing, analysis and simulation tools for population genetics and genomics.* BMC Genetics, 2012. **13**(1): p. 27.
103. Purcell, S., et al., *PLINK: a tool set for whole-genome association and population-based linkage analyses.* The American Journal of Human Genetics, 2007. **81**(3): p. 559-575.
104. Lansing, J.S., et al., *Male dominance rarely skews the frequency distribution of Y chromosome haplotypes in human populations.* Proceedings of the National Academy of Sciences, 2008. **105**(33): p. 11645-11650.
105. Langergraber, K.E., et al., *Generation times in wild chimpanzees and gorillas suggest earlier divergence times in great ape and human evolution.* Proceedings of the National Academy of Sciences, 2012. **109**(39): p. 15716-15721.
106. Martin, A.P. and S.R. Palumbi, *Body size, metabolic rate, generation time, and the molecular clock.* Proceedings of the National Academy of Sciences, 1993. **90**(9): p. 4087-4091.

Appendix

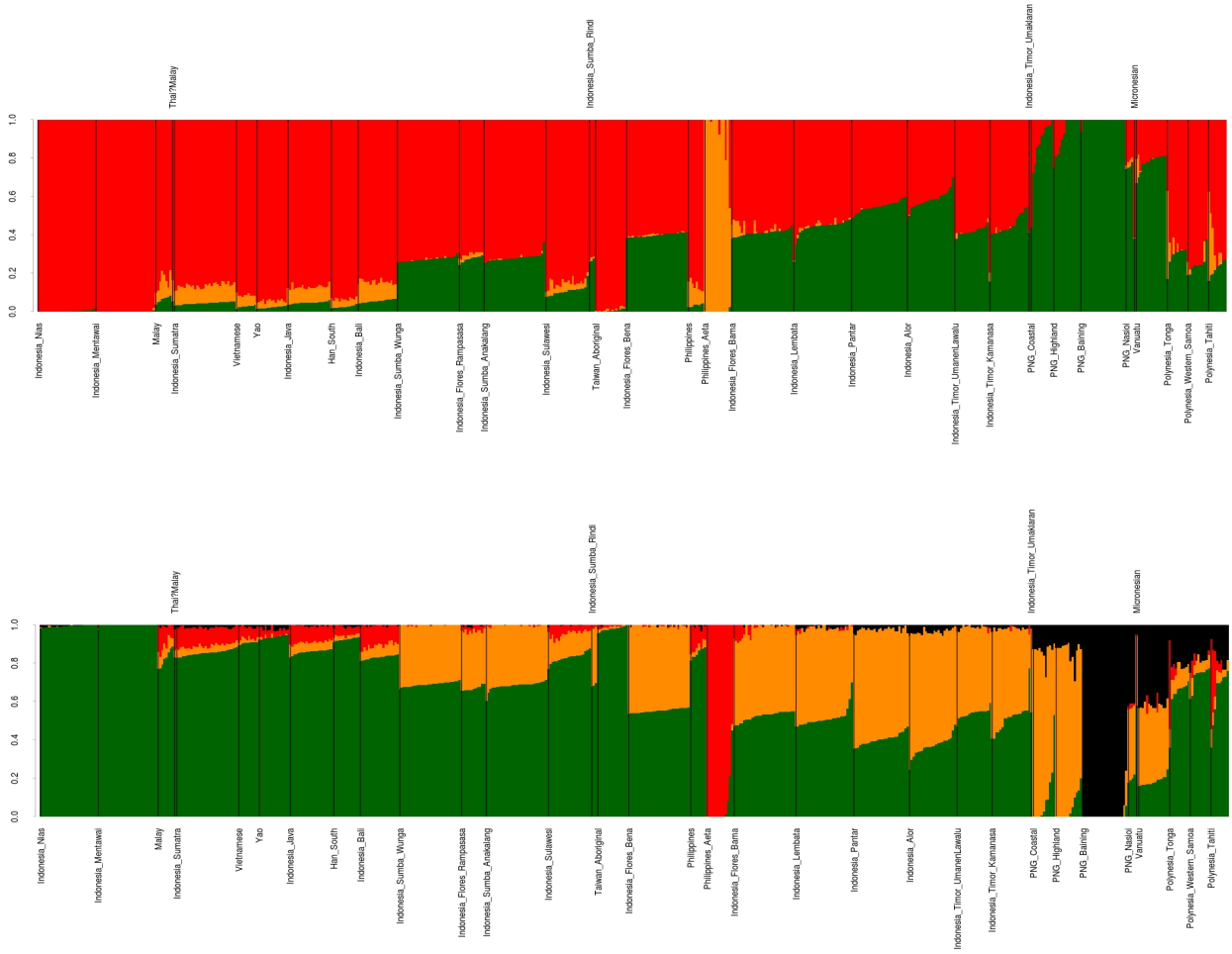

Figure S1: ADMIXTURE results for *K* from 3 to 4 respectively. Color denotes the ancestry component. Interestingly, the Philippines Aeta population appears as a different cluster on all bar plots, except for *K* = 2 (Fig. 3.1.1). One possible explanation for this is that the Philippines Aeta population descends from a Negrito population.

Error rates for *K* from 2-6:

Number of *K*	Error rate
2	0.44693
3	0.44581
4	0.44388
5	0.44250
6	0.44171

Table S1. Samples size by population.

Population	Sample size
Han South	13
Indonesia Flores Bama	30
Indonesia Java	21
Indonesia Nias	28
Indonesia Sumatra	30
Indonesia Sumba Wunga	30
Indonesia Timor Umanen Lawalu	17
Indonesia Alor	23
Indonesia Flores Bena	30
Indonesia Lembata	28
Indonesia Pantar	27
Indonesia Sumba Anakalang	30
Indonesia Timor Kamanasa	19
Indonesia Bali	19
Indonesia Flores Rampasasa	12
Indonesia Mentawai	29
Indonesia Sulawesi	21
Indonesia Sumba Rindi	3
Indonesia Timor Umaklaran	1
Philippines	8
Philippines Aeta	13
Taiwan Aboriginal	15
Malay	8
Vietnamese	10
Yao	15
PNG Coastal	11
PNG Highland	13
PNG Nasioi	4
Polynesia Tahiti	9
Polynesia Tonga	10
Polynesia Western Samoa	10

Section S1. Source code of the latest version of the model. Classes describing Deme, Individual, Individuals (model controller class), and graphical user interface (DemesWithUI.java) are provided.

1. Deme.java

```
package sim.app.pacific_9;

import java.util.HashMap;

import ec.util.MersenneTwisterFast;
import sim.engine.*;
import sim.util.*;

public class Deme implements Steppable {

	public MersenneTwisterFast random = new MersenneTwisterFast();
	int z = 0;
	public double admixture_proportion = 0.0;
	public double asian_markers_counter = 0.0;
	public int population_size = 0;
	public Bag nearest_neighbours = new Bag();
	public DoubleBag distances = new DoubleBag();
	public HashMap distance_coordinates = new HashMap();
	public Bag coordinates = new Bag();
	public Double2D location;
	public Bag neighbours_choice = new Bag();

	public Deme(Double2D loc) {
		location = loc;
	}

	public double get_admixture() {
		return admixture_proportion;
	}

	public int get_population_size() {
		return population_size;
	}

	public Bag get_closest_demes() {
		return neighbours_choice;
	}

	public void step(SimState state) {
		Individuals inds = (Individuals) state;
		z++;
		Double2D me =
inds.map_demes.getObjectLocationAsDouble2D(this);
		if (z == 1) {
			nearest_neighbours =
inds.map_demes.getNeighborsExactlyWithinDistance(me,
200);//getNearestNeighbors(me, 2, false, false, true,
nearest_neighbours)
			for (int neighb = 0; neighb <
nearest_neighbours.size(); neighb++) {
				Deme neighbour = (Deme)
nearest_neighbours.get(neighb);
				Double2D dist = neighbour.location;
				coordinates.add(dist);
```

```
                        double distance = me.distance(dist);
                        distances.add(distance);
                  }
                  for (int conn = 0; conn < distances.size(); conn++)
{
                        double dist = (double) distances.get(conn);
                        Double2D coord = (Double2D)
coordinates.get(conn);
                        distance_coordinates.put(dist, coord);
                  }
                  distances.sort();
                  for (int n = 0; n < distances.size(); n++) {
                        if (distances.get(n)!=0.0)
neighbours_choice.add((Double2D)
distance_coordinates.get(distances.getValue(n)));
                  }
                  nearest_neighbours.clear();
                  nearest_neighbours.shrink(0);
                  coordinates.clear();
                  coordinates.shrink(0);
                  distances.clear();
                  distances.shrink(0);
                  distance_coordinates.clear();

            }
            Bag population =
inds.map_individuals.getObjectsAtLocation(me);//.getNeighborsExactlyW
ithinDistance(me, 2.0);
            Bag gene_pool = new Bag();
            Bag ancestry = new Bag();
            for (int k = 0; k < population.size(); k++) {
                  Individual individual = (Individual)
population.get(k);
                  IntBag ind_genes = individual.getGenes();
                  boolean ind_anc =  individual.getAncestry();
                  gene_pool.add(ind_genes);
                  ancestry.add(ind_anc);
            }
            population.clear();
            population.shrink(0);
            population_size = gene_pool.size();
            int mating_pairs = random.nextInt(7)+7;
            for (int mp = 0; mp < mating_pairs; mp++) {
                  int random_papa = random.nextInt(gene_pool.size());
                  int random_mama = random.nextInt(gene_pool.size());
                  IntBag papa = (IntBag) gene_pool.get(random_papa);
                  IntBag mama = (IntBag) gene_pool.get(random_mama);
                  boolean papa_anc = (Boolean)
ancestry.get(random_papa);
                  boolean mama_anc = (Boolean)
ancestry.get(random_mama);
                  int offspring_num = (random.nextInt(3) + 1);
                  for (int b = 0; b < offspring_num; b++) {
                        IntBag baby = new
IntBag(Individual.genes_number);
                        for (int genes = 0; genes <
Individual.genes_number; genes++) {
                              int snp = random.nextInt(2);
                              if (snp == 0) {
                                    int papa_snp = (int)
papa.get(genes);
                                    baby.add(papa_snp);
                              }
```

```
                              else {
                                    int mama_snp = (int)
mama.get(genes);
                                    baby.add(mama_snp);
                              }
                        }
                        boolean baby_ancestry = papa_anc|mama_anc;
                        //requires validation with reverse
                        //baby.reverse();
                        Individual individual = new Individual(baby,
baby_ancestry);
                        individual.stopper =
inds.schedule.scheduleRepeating(individual);

      inds.map_individuals.setObjectLocation(individual, me);

      //System.out.println(inds.schedule.getSteps());
                  }
            }

            for (int q = 0; q < gene_pool.size(); q++) {
                  IntBag individual = (IntBag) gene_pool.get(q);
                  for (int f = 0; f < individual.size(); f++) {
                        if (individual.get(f) == 1)
asian_markers_counter++;
                  }
            }

            admixture_proportion =
asian_markers_counter/(gene_pool.size()*Individual.genes_number);
            asian_markers_counter = 0.0;

            gene_pool.clear();
            gene_pool.shrink(0);

            /*int people = 0;
            while (people < population.size()) {
                  IntBag individual = (IntBag)
population.get(people);
                  people++;
                  for (int snp_at_pos = 0; snp_at_pos <
individual.size(); snp_at_pos++) {
                        if (individual.get(snp_at_pos) == 1) {
                              pre_adm_prop++;
                        }
                  }
            }
            admixture_proportion =
pre_adm_prop/(population.size()*Individual.genes_number);
            pre_adm_prop = 0.0;
            System.out.println(admixture_proportion);*/
      }
}
```

2. Individual.java

```
package sim.app.pacific_9;

import ec.util.MersenneTwisterFast;
import sim.engine.*;
import sim.util.*;
```

```
import sim.field.continuous.*;

public class Individual implements Steppable {

	Stoppable stopper;
	int z = 0;
	public MersenneTwisterFast random = new MersenneTwisterFast();
	public static int genes_number = 1000;
	public IntBag genes = new IntBag(genes_number);
	boolean ancestry;
	public int max_age = 25;
	public Bag neighbours_choice = new Bag();
	public double a_migration_prob = 0.9;
	public double m_migration_prob = 0.94;

	public Individual(int num, boolean anc) {
		int[] genes_array = new int[genes_number];
		for (int pos = 0; pos < genes_number; pos++) {
			genes_array[pos] = num;
		}
		genes.addAll(genes_array);
		ancestry = anc;
	}

	public Individual(IntBag baby, boolean anc) {
		genes.addAll(baby);
		ancestry = anc;
	}

	public IntBag getGenes() {
		return genes;
	}

	public boolean getAncestry() {
		return ancestry;
	}

	public void step(SimState state) {
		z++;
		if (z > max_age) {
			Individuals inds = (Individuals) state;
			inds.map_individuals.remove(this);
			stopper.stop();
			return;
		}
		else {
			if (z%5 == 0) {
				if (ancestry) {
					if (random.nextDouble() >
a_migration_prob) {
						Individuals inds = (Individuals)
state;
						Double2D me =
inds.map_individuals.getObjectLocation(this);
						Bag demes =
inds.map_demes.getObjectsAtLocation(me);
						Deme deme = (Deme) demes.get(0);
						neighbours_choice.clear();
						neighbours_choice.shrink(0);

	neighbours_choice.addAll(deme.get_closest_demes());
						int deme_choice = 0;
```

```
                                    if (neighbours_choice.size() > 3)
{
                                          deme_choice =
random.nextInt(4);
                                    }
                                    Double2D migration_point =
(Double2D) neighbours_choice.get(deme_choice);

      inds.map_individuals.setObjectLocation(this, migration_point);
                              }
                        }

                        else {
                              if (random.nextDouble() >
m_migration_prob) {
                                    Individuals inds = (Individuals)
state;
                                    Double2D me =
inds.map_individuals.getObjectLocation(this);
                                    Bag demes =
inds.map_demes.getObjectsAtLocation(me);
                                    Deme deme = (Deme) demes.get(0);
                                    //need to check if clear is
necessary
                                    neighbours_choice.clear();
                                    neighbours_choice.shrink(0);

      neighbours_choice.addAll(deme.get_closest_demes());
                                    Double2D migration_point =
(Double2D) neighbours_choice.get(0);

      inds.map_individuals.setObjectLocation(this, migration_point);
                              }
                        }
                  }
                  /*if (z%5 == 0) {
                        Individuals inds = (Individuals) state;
                        if (random.nextDouble() > 0.94) {
                              Double2D migration_point = new
Double2D();
                              DoubleBag distances = new DoubleBag();
                              Double minimal_distance = 0.0;
                              Bag locations = inds.locations;
                              Double2D my_loc =
inds.map_individuals.getObjectLocationAsDouble2D(this);
                              for (int loc = 0; loc <
locations.size(); loc++) {
                                    Double2D said_deme =
(Double2D)locations.get(loc);
                                    Double distance =
my_loc.distance(said_deme);
                                    distances.add(distance);
                              }
                              for (int min_dis = 0; min_dis <
distances.size(); min_dis++) {
                                    if (minimal_distance <
distances.get(min_dis)) {
                                          minimal_distance =
distances.get(min_dis);
                                          migration_point =
(Double2D) locations.get(min_dis);
                                    }
                              }
```

```
      inds.map_individuals.setObjectLocation(this, migration_point);
                        }
                  }*/
            }
      }

}
```

3. Individuals.java

```
package sim.app.pacific_9;

import ec.util.MersenneTwisterFast;
import sim.engine.*;
import sim.util.*;
import sim.field.continuous.*;

public class Individuals extends SimState {

      public int asian_demes_number = 40;
      public int melanesian_demes_number = 40;
      public int individuals_per_deme = 100;
      public Continuous2D map_individuals = new Continuous2D(10,
600.0, 600.0);
      public Continuous2D map_demes = new Continuous2D(10, 600.0,
600.0);
      //public Bag locations = new
Bag(asian_demes_number+melanesian_demes_number);
      public Individuals(long seed) {
            super(seed);
      }

      public void start() {
            super.start();
            map_demes.clear();
            map_individuals.clear();
            for (int i = 0; i < asian_demes_number; i++) {
                  double latitude = random.nextDouble()*200;
                  double longitude = random.nextDouble()*200;
                  Double2D location = new Double2D(latitude,
longitude);
                  Deme deme = new Deme(location);
                  schedule.scheduleRepeating(1.0, 1, deme);
                  //locations.add(location);
                  map_demes.setObjectLocation(deme, location);
                  for (int z = 0; z < individuals_per_deme; z++) {
                        Individual individual = new Individual(1,
true);
                        individual.stopper =
schedule.scheduleRepeating(individual);
                        map_individuals.setObjectLocation(individual,
location);
                  }
            }

            for (int i = 0; i < melanesian_demes_number/4; i++) {
                  double latitude = random.nextDouble()*200+400;
                  double longitude = random.nextDouble()*200+400;
                  Double2D location = new Double2D(latitude,
longitude);
                  Deme deme = new Deme(location);
```

```
                  schedule.scheduleRepeating(1.0, 1, deme);
                  //locations.add(location);
                  map_demes.setObjectLocation(deme, location);
                  for (int z = 0; z < individuals_per_deme; z++) {
                        Individual individual = new Individual(2,
false);
                        individual.stopper =
schedule.scheduleRepeating(individual);
                        map_individuals.setObjectLocation(individual,
location);
                  }
            }
            for (int i = 0; i < melanesian_demes_number/4; i++) {
                  double latitude = random.nextDouble()*200+300;
                  double longitude = random.nextDouble()*200+300;
                  Double2D location = new Double2D(latitude,
longitude);
                  Deme deme = new Deme(location);
                  schedule.scheduleRepeating(1.0, 1, deme);
                  //locations.add(location);
                  map_demes.setObjectLocation(deme, location);
                  for (int z = 0; z < individuals_per_deme; z++) {
                        Individual individual = new Individual(2,
false);
                        individual.stopper =
schedule.scheduleRepeating(individual);
                        map_individuals.setObjectLocation(individual,
location);
                  }
            }
            for (int i = 0; i < melanesian_demes_number/4; i++) {
                  double latitude = random.nextDouble()*200+200;
                  double longitude = random.nextDouble()*200+200;
                  Double2D location = new Double2D(latitude,
longitude);
                  Deme deme = new Deme(location);
                  schedule.scheduleRepeating(1.0, 1, deme);
                  //locations.add(location);
                  map_demes.setObjectLocation(deme, location);
                  for (int z = 0; z < individuals_per_deme; z++) {
                        Individual individual = new Individual(2,
false);
                        individual.stopper =
schedule.scheduleRepeating(individual);
                        map_individuals.setObjectLocation(individual,
location);
                  }
            }
            for (int i = 0; i < melanesian_demes_number/4; i++) {
                  double latitude = random.nextDouble()*200+100;
                  double longitude = random.nextDouble()*200+100;
                  Double2D location = new Double2D(latitude,
longitude);
                  Deme deme = new Deme(location);
                  schedule.scheduleRepeating(1.0, 1, deme);
                  //locations.add(location);
                  map_demes.setObjectLocation(deme, location);
                  for (int z = 0; z < individuals_per_deme; z++) {
                        Individual individual = new Individual(2,
false);
                        individual.stopper =
schedule.scheduleRepeating(individual);
```

```
                        map_individuals.setObjectLocation(individual,
location);
                  }
            }
      }

      public static void main(String[] args) {
            doLoop(Individuals.class, args);
            System.exit(0);
      }

}
```

4. DemesWithUI.java

```
package sim.app.pacific_9;

import sim.engine.*;
import sim.display.*;
import sim.portrayal.DrawInfo2D;
import sim.portrayal.grid.*;
import sim.portrayal.continuous.*;


import java.awt.*;

import javax.swing.*;

import ec.util.MersenneTwisterFast;

public class DemesWithUI extends GUIState {

      org.jfree.data.xy.XYSeries series;
      sim.util.media.chart.TimeSeriesChartGenerator chart;

      public Display2D display;
      public JFrame displayFrame;
      ContinuousPortrayal2D demesPortrayal = new
ContinuousPortrayal2D();


      public static void main(String[] args) {
            DemesWithUI ex = new DemesWithUI();
            Console c = new Console(ex);
            c.setVisible(true);
      }

      public DemesWithUI() { super(new
Individuals(System.currentTimeMillis())); }

      public DemesWithUI(SimState state) { super(state); }

      public static String getName() { return "Demes Simulation"; }

      public void start() {
            super.start();
            setupPortrayals();
      }

      public void setupPortrayals() {
            demesPortrayal.setField(((Individuals)state).map_demes);
```

```
        demesPortrayal.setPortrayalForAll( new
sim.portrayal.simple.OvalPortrayal2D() {
            public void draw(Object object, Graphics2D graphics,
DrawInfo2D info) {
            Deme deme = (Deme)object;
            int admixtureLevel = (int) (deme.get_admixture() * 255);
            //double demeDiameter = (double) deme.getDiameter();
            int population_size = (int)deme.get_population_size();
            double demeDiameter = (double) (population_size/10);
            if (admixtureLevel > 255) admixtureLevel = 255;
            paint = new Color(admixtureLevel, 0, 255 -
admixtureLevel);
            scale = demeDiameter;
            super.draw(object, graphics, info);
            }
        });
        display.reset();
      }

      public void quit() {
            super.quit();
            if (displayFrame != null) displayFrame.dispose();
            displayFrame = null;
            display = null;
      }

      public void init(Controller c) {
            super.init(c);
            display = new Display2D(600, 600, this);
            display.setClipping(true);
            displayFrame = display.createFrame();
            displayFrame.setTitle("Pacific Island Display");
            c.registerFrame(displayFrame);
            displayFrame.setVisible(true);
            display.setBackdrop(Color.black);
            display.attach(demesPortrayal, "DemesSimulation");
      }

      public Object getSimulationInspectedObject() {
            return state;
      }


}
```