4 Demographic Change

The workers have taken it into their heads that they, with their busy hands, are the necessary, and the rich capitalists, who do nothing, the surplus population.
– Friedrich Engels, *The Condition of the Working Class in England*

4.1 Background

In the previous chapter, we considered a population that maintains constant population size. However, we often need to simulate a population whose size is, or was, in flux. Brief consideration brings to mind many such cases. The population size of our own species has expanded dramatically, particularly over the last century; total census population of the human species grew from ~2 billion in 1927 to ~7 billion in 2011. Invasive species are primarily defined as a category of alien species by their rapid population growth following introduction to a new locality. Declines in population size are also common in the natural world. Nascent island populations—the result of immigration from mainland populations—are necessarily much smaller than the parent population. Anthropogenic changes to the environment have caused countless species throughout the world to decline in size precipitously.

Demographic change has a *genome-wide* effect on genetic variation because changes in population size change the number of individuals and therefore the number of copies of entire genomes present in a population. In other words, the number of copies of a specific locus or chromosome is not altered *to the exclusion* of most other loci/chromosomes. This contrasts sharply with natural selection, which generally only affects quantity and pattern of genetic variation in the immediate vicinity of the locus targeted by selection.

Note that my use of the term "demographic change" is limited to changes in population size. We will not explicitly model changes in birth/death rates

R. J. Haasl, *Nature in Silico*, https://doi.org/10.1007/978-3-030-97381-0_4

or the age structure of populations. In this chapter, N_t represents population size at time t, while N_{100} represents population size at generation 100. A symbological distinction is not made between census and effective population sizes, but we assume *effective* population size throughout the chapter.

4.1.1 Models of Demographic Change

Consider a population whose census size increases monotonically from 1000 individuals to 10,000 individuals over the course of 100 generations. We can imagine a number of trajectories the population might follow to bring about this increase (Fig. 4.1). The simplest are *instantaneous* population expansion in which the increase occurs in one generation and a *linear* expansion in which $(10{,}000-1000)/100 = 90$ individuals are added each generation. Somewhat more complicated are the widely used *exponential* and *logistic* models of population growth, which we now consider in turn.

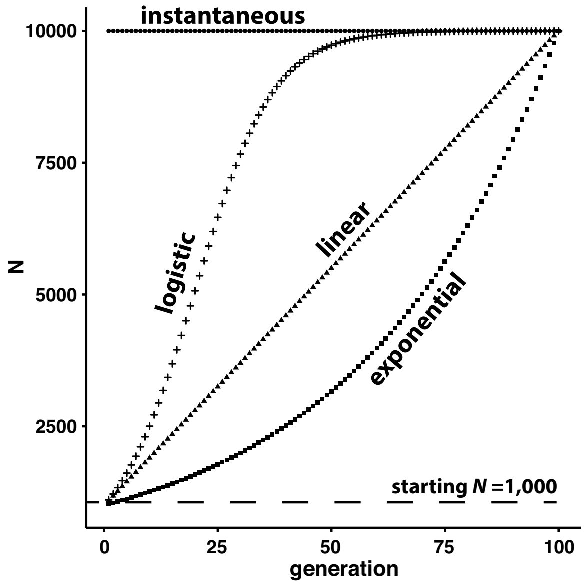

Fig. 4.1 Four models of population expansion from $N = 1000$ to $N = 10{,}000$ over the course of 100 generations. For the logistic model, the low density growth rate $r \approx 0.114$. For the exponential model, the intrinsic rate of increase $r \approx 0.023$

4.1.1.1 Exponential Population Growth or Decline

Each generation, exponential population growth increases population size by a constant fraction of the current population size. Because the base population size increases each generation, the number of individuals added each generation grows rapidly, ultimately leading to runaway population growth. Following t generations of exponential increase, beginning with a population size of N_0, population size N_t is

$$N_t = N_0 e^{rt}. \tag{4.1}$$

Given t, N_0, and N_t (i.e., parameters we are likely to specify for our simulation), we can calculate the corresponding value of r, the *intrinsic rate of increase*, using the following rearrangement of Eq. 4.1:

$$r = \ln\left(\frac{N_t}{N_0}\right)t^{-1} \tag{4.2}$$

Thus, for the example described at the beginning of this section and shown in Fig. 4.1, $r = \ln(10) \times 0.01 \approx 0.023$. Exponential population *decline* follows the same equations, but r is negative in sign. For example, the inverse scenario in which population size declines from $N_0 = 10{,}000$ to $N_{100} = 1000$ requires $r \approx -0.023$.

4.1.1.2 Logistic Population Growth

Runaway population growth is ultimately constrained by limited resources—an idea famously promulgated by Thomas Malthus. A small population may grow exponentially, but as population size approaches its *carrying capacity*, K, growth slows down. Logistic growth is therefore characterized by a sigmoidal growth curve, where growth from a small population size begins at near-exponential rates but then declines precipitously as population size approaches K. The modifying influence of K on growth rate is easily seen in the discrete logistic equation:

$$N_{t+1} = N_t + rN_t\left(1 - \frac{N_t}{K}\right). \tag{4.3}$$

When $N_t << K$, nearly rN_t individuals are added to current population size by the next generation. Under the logistic model, r is therefore sometimes named the *low density growth rate*. However, at carrying capacity ($N_t = K$), rN_t is multiplied by zero, resulting in constant population size of K. Given K and N_0, the logistic model for population size after t generations is

$$N_t = \frac{KN_0}{N_0 + (K - N_0)e^{-rt}} \tag{4.4}$$

4.1.2 Using Coalescent Simulation to Build Intuition Regarding the Genetic Consequences of Demographic Change

To build basic intuition regarding the effects of population increase *and* decrease on genetic variation, we use coalescent simulation implemented in the R package `coala` (Staab and Metzler 2016, see Chapter 2). To make the effects of population growth and decline as clear as possible, we will compare the genealogies and site frequency spectra of a population at mutation-drift equilibrium ($N_e = 25{,}000$ diploid individuals) to simulations of two extreme scenarios: (1) an instantaneous population expansion from $N_e = 50$ to $N_e = 25{,}000$ and (2) an instantaneous population bottleneck from $N_e = 25{,}000$ to $N_e = 50$. In both cases, the genetic sample is drawn 50 generations following the sudden change in population size. We draw samples of $n = 20$ and $n = 100$ to show the effect of demographic change on the structure of genealogy (left column panels in Fig. 4.2) and the SFS (right column panels of Fig. 4.2), respectively. A bottleneck is a biologically relevant scenario; environmental catastrophe may rapidly decimate a local population, and a small, founder population derived from a large population may in one generation establish itself elsewhere. An instantaneous population expansion of the size simulated here is less relevant from a biological/ecological standpoint. Nevertheless, its extremity is used to showcase the effects of population expansion.

In all scenarios, a point mutation rate of 1×10^{-8} and a 20,000-bp sequence were simulated. We first examine the baseline results from simulating a population at mutation-drift equilibrium. Looking backward in time, the $n = 20$ sequences of a sample *initially* coalesce with rapidity, while coalescent times (the waiting time to the merger of sequence lineages) increase as the number of lineages to coalesce decreases (Fig. 4.2a). This is characteristic of the neutral coalescent process, in which the probability of coalescence between an *unspecified* pair of lineages in the previous generation equals $\frac{n(n-1)}{4N_e}$. As n—the remaining number of genes that have not coalesced—decreases, the probability of coalescence decreases, and coalescence time increases on average.

Under the neutral coalescent, the site frequency spectrum should be distributed geometrically. However, this expectation is only met when spectra are averaged across numerous unlinked loci. Although the SFS of one simulation is not geometrically distributed, it does show hallmarks of what we expect the distribution of genetic variation to look like under neutral, equilibrium conditions (Fig. 4.2b). The most abundant types of polymorphic sites (SNPs) are those in which the frequency of derived alleles is low. In particular, the most abundant type of SNP is a *singleton* in which only one of the 100 sampled sequences shows a derived allele. The SFS for this simulation also shows a declining trend in the number of SNPs with high frequencies of the derived allele. Again, because the results are shown for a single simulation (and, therefore, a single locus), we do see some clear exceptions to this trend. For example, the numbers of SNPs with derived allele counts are remarkably common — 5 and 13, respectively (Fig. 4.2b).

Figure 4.2a and b is derived from the results of one simulation. Devise a hypothesis to explain the bimodal SFS in Fig. 4.2b given the genealogy shown in Fig. 4.2a.

Both the genealogy and SFS under instantaneous population expansion (Fig. 4.2c,d) show a clear departure from that of the equilibrium scenario. The genealogy is comb-like (Fig. 4.2c); all 20 sampled sequences evolve independently of each other until they rapidly coalesce in a compressed period of time in the recent past. The SFS is notable for (1) the small number of polymorphic sites (cf. scale of the y-axis in Fig. 4.2b and d) and (2) derived allele counts exclusively ≤ 3 out of 100 sampled sequences. We expect a drastic increase in population size to alter patterns of genetic variation, but the specific question of interest is why these particular alterations are characteristic of population expansion.

Prior to the expansion, the population of 50 diploid individuals collectively carries just 100 copies of the simulated locus. As detailed in the last chapter, small populations harbor less variation than large populations. It is possible, even likely, that the initial population was devoid of variation at the locus simulated. However, the in-one-generation increase in population size to 25,000 diploid individuals provides a large reservoir of sequences that can incur mutations. In other words, the small population with depauperate genetic variation becomes a large population with depauperate genetic variation. The simulated sequence is a (nearly) blank canvas upon which new variants may be written. However, the long terminal branches of the genealogy mean the vast majority of derived alleles found in a sample are singletons. This process of occasionally generating derived alleles within a genetically depauperate sequence yields the SFS seen in Fig. 4.2d. Polymorphic sites are rare because we have only allowed 50 generations for new mutations to arise on the "blank canvas." Moreover, those polymorphic sites that are identified in the sample show low derived allele counts because they have had little time to spread through the population.

We next consider the genesis of the distinct topology associated with a large and instantaneous population expansion: in our example, looking backward in time from the present, *no* coalescence of the $n = 20$ sequences for an extended period followed by a flurry of coalescent events in the past (Fig. 4.2c). As you might guess, this cluster of coalescent events dates back to the much smaller, pre-expansion population. The long absence of coalescence following the expansion is a direct consequence of the sudden increase in

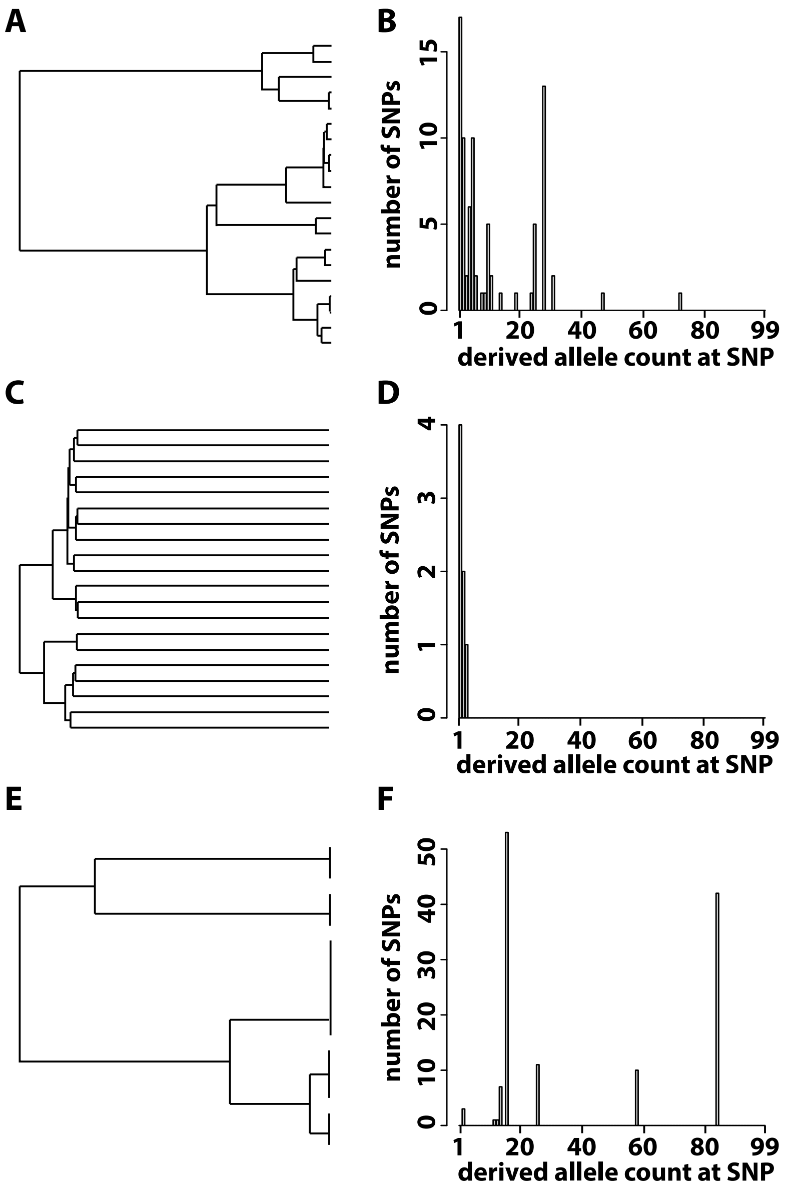

Fig. 4.2 Typical genealogies and site frequency spectra for a neutral, equilibrium model (**a**,**b**), an instantaneous population expansion (**c**,**d**), and an instantaneous population bottleneck (**e**,**f**). Although there are 20 branch tips in (**f**), coalescent events take place so rapidly moving backward in time that it appears the tips are too short to visualize. For clarity, the genealogies are based on a sample of just 20 sequences; the site frequency spectra are calculated from a larger sample of 100 sequences. Samples for the population bottleneck and expansion scenarios were drawn 50 generations following the demographic event. In the genealogies, time flows forward from left to right

the number of sequences present in the population. The probability $P(n)$ that no coalescent event takes place in the previous generation is equal to

$$P(n) = \frac{2N_e - 1}{2N_e} + \frac{2N_e - 2}{2N_e} + \dots + \frac{2N_e - n + 1}{2N_e}, \tag{4.5}$$

where n is the number of sequences in the sample. It can be shown that $P(n) \approx 1 - \binom{n}{2}\frac{1}{2N_e}$ (Hudson 1990). In our current example, then, the probability that *none* of the lineages coalesce for the 50 generations since the expansion—with $n = 20$ and $2N_e = 50{,}000$—equals

$$P(20)^{50} \approx \left[1 - \binom{20}{2}\frac{1}{50{,}000}\right]^{50} \approx 0.827 \tag{4.6}$$

In other words, greater than 80% of the time, *none* of the $\binom{20}{2} = 190$ pairs of sequences will coalesce in the 50 generations following the expansion. Contrast this with the small, pre-expansion population of just 50 individuals ($2N_e = 100$) where the probability that one pair of sequences will coalesce with a common ancestor in the previous generation is a near certainty. Although an instantaneous, 500-fold increase in population size is an extreme example, these ideas show that a comb-like topology (perhaps, more commonly named a star topology) is characteristic of genealogies associated with population expansions.

The average effect of a population bottleneck on a genealogy is opposite to that of a population expansion (Fig. 4.2e); most coalescent events take place post-bottleneck, while the last coalescent events are often much older and traceable to the larger, *pre*-bottleneck population. The resulting SFS (Fig. 4.2f) shows isolated peaks associated with mutations occurring on the long branches that become derived alleles shared by the descendant sequences of these long branches.

Finally, we look at the summary statistics—and estimators of θ—nucleotide diversity π and Watterson's estimator θ_W for the same demographic scenarios just discussed. Our expectations are the following: (1) for the scenario of no demographic change, both estimators should zero-in on the per-locus expected value of $\theta = 20$, and (2) because both summary statistics are directly related to the site frequency spectrum and we have just seen that demographic change has profound impacts on the SFS, both summary statistics should be greatly perturbed by the strong demographic change modeled. Both of these expectations are met by this simple simulation study (Fig. 4.3). In addition, notice that under neutral, equilibrium conditions, θ_W shows superior resolution to π as an estimator of per-locus θ (Fig. 4.3a).

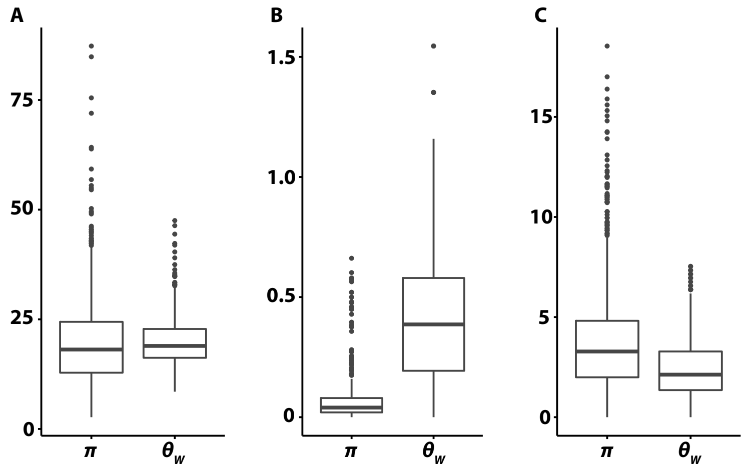

Fig. 4.3 Boxplots showing the diversity in *per-locus* values of nucleotide diversity π and θ_W across 1000 replicate simulations for each of three demographic models. Panels (**a**), (**b**), and (**c**) here correspond to the same demographic models used to produce the results shown in Fig. 4.2 panels (**a-b**), (**c-d**), and (**e-f**), respectively—namely, ((**a**)) No demographic change. ((**b**)) Instantaneous population expansion. ((**c**)) Instantaneous population bottleneck. $N_e = 25{,}000$, per-site $\mu = 1 \times 10^{-8}$, `seqlength` = 20,000 bp, and `sampsize` = 100. **Note the different y-axis scales between panels.** Samples for the population bottleneck and expansion scenarios were drawn 50 generations following the demographic event. On average, π - θ_W is zero for a neutral locus in a population of constant size, a negative value for population expansion, and a positive value for a population decline; the results shown here validate those theoretical expectations. The median estimate of per-locus θ (for *both* estimators) at a neutral locus in a population of constant size (panel A) matches the expectation of $\theta = 4N_e\mu \times seqlen == 4 \times 50{,}000 \times 10^{-8} \times 10{,}000 = 20$. Both summary statistics are therefore good estimators of θ when the atmosphere is ostensibly boring—neutral and static. **In addition, this is another example of validation in which our simulation results match theoretical expectations**

4.2 Forward Simulation of Demographic Change

We now modify the forward simulation program FORTUNA—introduced in Chap. 3—to facilitate simulation of varied demographic scenarios. Modifications also include updates to `summarystatistics.h` that enable calculation of the summary statistic Tajima's D.

4.2.1 Requisite New Parameters

Imagine that the modeled population is subject to a series of changes in population size. The information necessary to model each size change includes the following: (1) type, e.g., instantaneous or logistic; (2) generation numbers for the onset and conclusion of the demographic regime; (3) population size at onset of the demographic regime; and (4) any additional parameter(s) required to specify the demographic model. We first add program parameters to the `parameters` and `params.h` files.

modifications to parameters and params.h files to implement demographic change

```
 1  // additions to parameters
 2  demography 0 1 4
 3  dem_parameter 0 -9900 0.02
 4  dem_start_gen 0 1001 1501
 5  dem_end_gen 1000 1500 20000
 6  carrying_cap 0 0 10000
 7
 8  // additions to params.h
 9  extern vector<int> demography;
10  extern vector<double> dem_parameter;
11  extern vector<int> dem_start_gen;
12  extern vector<int> dem_end_gen;
13  extern vector<int> carrying_cap;
14  extern vector<int> pop_schedule;
```

Pay special attention to the parameter values listed on lines 2–6; each is followed by numbers that specify details of three sequential demographic events in the population. For example, the `demography` parameter specifies the type of demographic event and takes one of five values:

- 0 = constant population size—i.e., no change
- 1 = instantaneous change
- 2 = linear change
- 3 = exponential change
- 4 = logistic change

Thus, the entry `demography 0 1 4` indicates the population first maintains the constant size specified by parameter `popsize` (Chap. 3), experiences an instantaneous change, and then enters a phase of logistic change (growth in this case). The values listed after the other four correspond to these three demographic regimes in the same order. Look again at the parameter values in the previous listing. Take a moment to understand that numbers −9900, 1001, 1500, and 0 provide the needed details to model the second selective regime (an instantaneous change in population size).

The parameters dem_start_gen and dem_end_gen specify the starting and ending generation for a given demographic event, respectively. The value of dem_parameter provides a necessary parameter for a given demographic model:

- Absolute change in population size for instantaneous change
- Per-generation change in population size for linear change
- Rate parameter for both exponential and logistic change

Logistic change requires specification of a *second* model parameter—carrying capacity—provided to the parameter carrying_cap. Each parameter is stored in a vector, and multiple values for a given parameter should be separated by whitespace of any size.

The parameter values in the previous listing will be used in the next section. Collectively, they specify no change in population size for the first 1000 generations, followed by an instantaneous loss of 9900 individuals (out of 10,000, as specified by the parameter popsize) that lasts for 500 generations, and finally logistic growth at a rate of 0.02 with a carrying capacity of 10,000. The logistic demographic regime holds from generation 1501 until the end of the simulation at generation 20,000.

File params.h also declares a vector<int> called pop_schedule, which holds the population size at every generation (generations 0–20,000 in the current case) based on the demographic model parameters just discussed. The values of pop_schedule are calculated in params.cc using the following additions to the file:

additions to **params.cc**

```
...
vector<int> get_multi_int_param(const string &key)
{
  vector<int> vec;
  istringstream iss(parameters[key].c_str());
  string param;
  while(getline(iss, param, ' '))
    vec.push_back(atoi(param.c_str()));
  return vec;
}

vector<double> get_multi_double_param(const string &key)
{
  vector<double> vec;
  istringstream iss(parameters[key].c_str());
  string param;
  while(getline(iss, param, ' '))
    vec.push_back(atof(param.c_str()));
  return vec;
}

```

```
vector<int> create_pop_schedule()
{
  vector<int> ps;
  int i=0;
  int cursize = popsize;
  for (int step = 0; step < demography.size(); ++step) {
    for (; i<dem_start_gen[step]; ++i)
      ps.push_back(cursize);
    for (; i <= dem_end_gen[step]; ++i) {
      switch(demography[step]) {
        case 0: ps.push_back(cursize); // no size change
              break;
        case 1: if (i == dem_start_gen[step])
                  cursize += dem_parameter[step];
                ps.push_back(cursize); // instantaneous
                break;
        case 2: cursize += dem_parameter[step];
                ps.push_back(cursize); // linear
                break;
        case 3: cursize *= exp(dem_parameter[step]); // exponential
                ps.push_back(cursize);
                break;
        case 4: cursize = (carrying_cap[step] * cursize) /
                          (cursize + (carrying_cap[step] -
                             ↪ cursize)*exp(-1*dem_parameter[step])); //
                             ↪ logistic
                ps.push_back(cursize);
      }
    }
  }
  return ps;
}

// additional variable declarations
vector<int> demography;
vector<double> dem_parameter;
vector<int> dem_start_gen;
vector<int> dem_end_gen;
vector<int> carrying_cap;

 int process_parameters() {
 ...
  demography = get_multi_int_param("demography");
  dem_parameter = get_multi_double_param("dem_parameter");
  dem_start_gen = get_multi_int_param("dem_start_gen");
  dem_end_gen = get_multi_int_param("dem_end_gen");
  carrying_cap = get_multi_int_param("carrying_cap");
...
}
...
vector<int> pop_schedule = create_pop_schedule();
```

The functions `get_multi_int_param( )` and `get_multi_double_param( )` (lines 2–20) allow read-in of multi-valued parameters and are used in the calculation of `pop_schedule` carried out by function `create_pop_schedule( )` (lines 22–51). Population size begins at the value specified by the parameter `popsize` and is stored as current population size in the variable `cursize` (line 26). The `for` loop beginning at line 27 will step through each demographic event and, for each generation of the individual demographic regime, calculate population size. Note that once the starting generation of a demographic event is reached, as determined by the control statement at line 28, the program enters the `for` loop from line 30 through line 48 until the last generation of the demographic event is reached. Each iteration of this `for` loop employs a switch statement (lines 31–47), which queries the type of demographic change (line 31), updates population size (`cursize`) accordingly, and pushes `cursize` to the population schedule (`pop_schedule`). Finally, the calculated population schedule is returned (line 50).

Small changes to `population.h` are also required. However, we will cover these changes in Sect. 4.2.3 following a brief discussion of how to calculate the summary statistic Tajima's *D*.

4.2.2 *Calculating Tajima's D*

As discussed in Chap. 3, Tajima's *D* quantifies the difference between two estimators of θ: nucleotide diversity (π) and Watterson's estimator (θ_W) (Tajima 1989). At equilibrium, we expect these estimators to provide roughly equal estimates, yielding a value of $D = 0$. Positive or negative deviations from zero are characteristic of various evolutionary events, including demographic change. Specifically, Tajima's *D* is defined as

$$D = \frac{\pi - \theta_W}{\sqrt{\mathtt{Var}(\pi - \theta_W)}} \tag{4.7}$$

Now let $a_1 = \sum_{i=1}^{n-1} \frac{1}{i}$, $a_2 = \sum_{i=1}^{n-1} \frac{1}{i^2}$, n be sample size (the number of sequences), and S be the number of segregating sites in the sample. Then, Tajima (1989) defines $\mathrm{Var}(\pi - \theta_W)$ as

$$\mathtt{Var}(\pi - \theta_W) = S\left(\frac{1}{a_1}\right)\left(\frac{n+1}{3(n-1)} - \frac{1}{a_1}\right) + S(S-1)\left(\frac{1}{a_1^2 + a_2}\right)\left(\frac{2(n^2+n+3)}{9n(n-1)} - \frac{n+2}{a_1 n} + \frac{a_2}{a_1^2}\right) \tag{4.8}$$

Keep in mind that $\theta_W = \frac{S}{a_1}$, where the denominator controls for sample size, which is necessary because large samples uncover a greater number of segregating sites than small samples. Similarly, the expected range of Tajima's *D* is expected to be greater for smaller sample sizes; the variance of

D in the denominator controls for this fact, as it is a function of n, a_1, and a_2, the latter two themselves being functions of n. The following function for calculating and returning Tajima's D is added to `summarystats.h`:

summarystats.h: function get_tajimas_d( )

```
 1  double get_tajimas_d (double pi, double watterson, int S) {
 2     double d = pi - watterson; // numerator
 3     double a1 = 0.;
 4     double a2 = 0.;
 5     for (double i=1.; i < sampsize; ++i) {
 6        a1 += 1./i;
 7        a2 += 1./(i*i);
 8     }
 9     double n = sampsize; // for easier expression
10     double var = watterson * ( (n+1) / (3*(n-1)) - 1/a1 ) +
11         ( S * (S-1) * ( 1 / (a1*a1+a2)) *
12        ( (2*(n*n + n+3))/(9*n*(n-1)) - (n+2)/(a1*n) + a2/(a1*a1) ) );
13     return(d / sqrt(var));
14  }
```

As reflected by the arguments to this function (line 1), it can only be called after π and θ_W have been calculated. Furthermore, calculation of Tajima's D requires us to use the per-locus rather than per-site estimates of θ. Therefore, the return values of the functions `get_pi( )` and `get_watterson( )` were also modified to reflect this necessary change. Specifically, `return (sumdiffs / numcomp)` rather than `return (sumdiffs / numcomp / seqlength)` returns the per-locus value of π. Similarly, the change to `return (S / denominator)` returns the *per-locus* number of segregating sites.

4.2.3 Final Changes to Program Files

Minor modifications to `population.h` are required to complete the implementation of additional functionality—namely, the ability to simulate demographic change and calculate Tajima's D. Because we are now calculating three summary statistics, we also make modifications to `population.h` that cause all summary statistic output to print to one file.

population.h: Chap. 4 modifications

```
// changes to private variables
ofstream sumstat_file; // all sumstats printed here

// change to function update_alleles( )
   if (current_count == pop_schedule[gen]*2) { // replaces popsize*2 in
        ↪ ch3 listing

```

```
 7 // changes to function reproduce ( )
 8   randomind.param(uniform_int_distribution<int>::param_type(0,pop_schedule
         ↪ [gen]-1));
 9    ...
10   for (int i=0; i< pop_schedule[gen==0 ? gen : gen-1]; ++i) { // replaces
         ↪ popsize in ch3 listing
11    ...
12    for (auto iter = individuals.begin(); iter != individuals.end() -
          ↪ pop_schedule[gen==0 ? gen : gen-1]; ++iter) // replaces
          ↪ popsize in ch3 listing
13    ...
14    individuals.erase(individuals.begin(), individuals.end()-
          ↪ pop_schedule[gen==0 ? gen : gen-1]); // replaces popsize in
          ↪ ch3 listing
15
16 // additions to function get_sample( )
17   double pi = get_pi(sample);
18   double watterson = get_watterson(sample, S);
19   double tajimasd = get_tajimas_d(pi, watterson, S);
20   sumstat_file << gen << " " << pi << " " << watterson << " " << tajimasd
         ↪ << endl;
21
22 // addition to function close_output_files( )
23   sumstat_file.close();
24
25 // changes to Population constructor
26 randomind.param(uniform_int_distribution<int>::param_type(0,pop_schedule[0]
       ↪ - 1)); // replaces popsize in ch3 listing
27    ...
28    for (int i=0; i<pop_schedule[0]; ++i) { // replaces popsize in ch3
          ↪ listing
29   ...
30   fname = "sumstats";
31   sumstat_file.open(fname.c_str());
32   sumstat_file << "gen pi watterson tajimasd" << endl;
```

Changes to the `population.h` file shown in lines 2, 17–20, 23, and 30–32 drive calculation of Tajima's *D* and output of all summary statistics to a single file named `sumstats`.

The remaining changes specified account for the potentially variable value of population size. In Chap. 3, where population size remained constant, we could simply use the value `popsize` every generation. Now, however, the population size at any given generation `gen` is stored in `pop_schedule[gen]`. Because population size may change each generation when modeling demographic change, the random variable `randomind` must be updated each generation so that potential indices of individuals chosen as parents fall between 0 and current population size less one. Initialization of the `randomind` is performed in the constructor (line 26) and updated each generation at the beginning of function `reproduce( )` (line 8). Variable population size also necessitates changes to the test expressions of several `for` loops (lines 10, 12, and 28) as well as an `erase( )` function (line 14). In line 10, the test expression is written as `i < pop_schedule[gen==0 ? gen : gen-1]`. We use the conditional operator for the index because `pop_schedule[-1]` is undefined; thus, for generation 0, we need to use the initial population size `pop_schedule[0]` as the test condition.

4.3 Simulating a Bottleneck Followed by Logistic Growth

Having modified the forward simulation program to allow demographic change and calculation of Tajima's D, we can now use the program to model a broadly realistic case of demographic change. Imagine a population of $N_e = 100{,}000$ diploid individuals at mutation-drift equilibrium. Catastrophic environmental change causes an instantaneous population bottleneck that reduces population size to $N_e = 100$, which lasts for 500 generations (Fig. 4.4a). After this, the environment recovers, and logistic growth ensues at a rapid growth rate of $r = 0.02$ and a carrying capacity of $N_e = 10{,}000$ (Fig. 4.4a).

The demographic parameter values required to specify the demographic model shown in Fig. 4.4a are those focused on at the beginning of Sect. 4.2.1. We sample 100 non-recombining sequences with a per-site mutation rate of $\mu = 1\times10^{-8}$ and a length of 250,000 bp from the simulated population every ten generations. Each simulation uses an initial coalescent simulation to generate sequences for generation 0 in the forward simulation. `mscommand` is set to `./ms 20000 1 -t 100 >ms_output` in `parameters`. Because `popsize` remains 10,000 and we are simulating a diploid locus, MS must generate 20,000 sequences. Furthermore, the value of $\theta = 100$ as the *per-locus* value of θ was obtained as follows: $\theta = 4N_e\mu\times2.5\times10^5 = 4\times10^4\times2.5\times10^{-8}\times2.5\times10^5 = 100$.

Figure 4.4b,c shows per-site estimates of θ for two independent replicates of the simulation setup detailed in the previous paragraph. Although the starting quantity of genetic variation generated by coalescent simulation differs in each case, qualitatively similar behavior is seen in both replicates. The instantaneous bottleneck immediately and drastically reduces genetic variation. Over the course of the next 18,500 generations, as population size increases rapidly to carrying capacity and holds there, genetic variation is slowly restored to expected equilibrium levels. Importantly, π recovers more rapidly than θ_W. Given that the numerator of Tajima's D is $\pi - \theta_W$, we expect this behavior to yield consistently negative values of D. This expectation is observed; roughly between generations 1500 and 5000, Tajima's D is < -2 in both simulations (Fig. 4.4d). In general, Tajima's D greater than 2 or less than -2 is considered a significant indicator of evolutionary change. Although a variety of evolutionary factors may be responsible for this deviation (see next section), we know the cause in our simulated population: rapid population

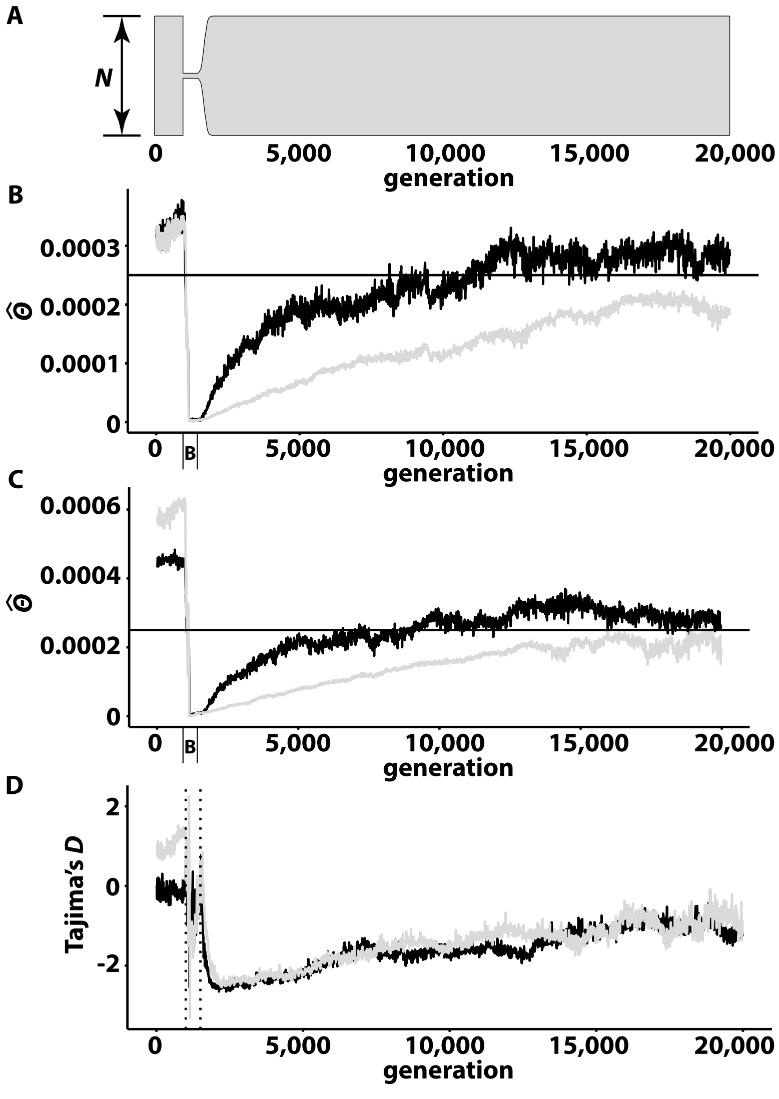

Fig. 4.4 Modeling demographic change. (**a**) The demographic model; a population of $N_e = 10,000$ diploid individuals is simulated using MS; forward simulation consists of 1000 generations at $N_e = 10,000$, followed by an instantaneous bottleneck that reduces the population to $N_e = 100$ diploid individuals for 500 generations, followed by logistic growth at a rate of $r = 0.02$ and a carrying capacity of $K = 10,000$. (**b** and **c**) Results from two independent replicates of the simulation, sampled every ten generations and summarized as two independent estimators of θ, θ_W (black lines) and π (gray lines). (D) The summary statistic Tajima's D for the simulation shown in (**b**; black line) and (**c**; gray line). The period of the 500-generation bottleneck is indicated by **B** or vertical dashed lines

expansion. In other words, Tajima's *D* provides a rather long-lived signal of population expansion.

Interestingly, Tajima's *D* does not provide us with evidence of the drastic population bottleneck that occurs at generation 1000. During the bottleneck, values of *D* become highly variable, but we fail to note a consistent value of $D > 2$, which is indicative of significant population decline (Fig. 4.4d). The loss of nearly all genetic variation with the instantaneous bottleneck explains this. Values of π and θ_W are both nearly zero immediately following the bottleneck, which does not provide a great enough difference between the two estimators to yield a clear signal of population decline. In the next section, we will model a gradual population decline, which does produce a clear increase of *D* to greater than 2. The population expansion provides a much clearer signal because the bottleneck produces a genetically depauperate population that sets the stage for a slow recovery of genetic variation as population size increases. The relative difference in recovery of π and θ_W then yields the significantly negative values of Tajima's *D* shown in Fig. 4.4d.

4.4 The Varying Utility of Summary Statistics for Inference

We now simulate a gradual, linear decline in population size. This situation is of practical relevance, as conservation biologists are often interested in assessing current population dynamics in order to determine if a population merits intervention. The results of this simulation will raise the issue of which summary statistics are best suited for a specific inferential task.

Consider a population of 50,000 diploid individuals at mutation-drift equilibrium. A linear decline of ten individuals per generation begins at generation 1000 until, at generation 5900, the population stabilizes at 1000 diploid individuals. We assume a per-site point mutation rate of $\mu = 1 \times 10^{-8}$ and a 50,000-bp sequence. Once again, we initialize a `Population` object using the results of coalescent simulation in MS. The `parameters` file that describes this population model (Fig. 4.5a) is

`parameters` for the model detailed in Sect. 4.4

```
popsize 50000
mutrate 1e-08
seqlength 5e04
sampsize 100
sampfreq 10
demography 0 2 0
dem_parameter 0 -10 0
dem_start_gen 0 1001 5901
dem_end_gen 1000 5900 8500
carrying_cap 0 0 0
useMS 1
mscommand ./ms 100000 1 -t 100 >ms_output
```

Figure 4.5b shows the per-site values of π (gray) and θ_W (black) sampled every ten generations. During the population decline, a *very* gradual increase in Tajima's D is observed (Fig. 4.5c). Furthermore, while θ_W begins to decline at 3000 generations, likely due to the loss of rare variants, π remains elevated until after the population decline ends at 5900 generations (Fig. 4.5b). Shortly after the decline stops at generation 5900, significant values of D are observed and maintained during the next 2000 generations (Fig. 4.5c).

At roughly generation 7000, there is a sharp drop in both θ_W and π. Generate a hypothesis that explains this seemingly anomalous "blip."

From the practical standpoint of the conservation biologist hoping to determine if a population is subject to a sustained population decline, these results are worrisome. During the nearly 4000 generations that the population is bleeding individuals, both π and Tajima's D fail to signal such. Moreover, the decline in θ_W is only visible to us because we have access data from the entire history of the population decline. θ_W at any one sample point would not signal a population decline.

On another note of caution, Fig. 4.4 makes clear that Tajima's D drops below the critical value of -2 following a drastic, though transient, population bottleneck. As will be shown in the last chapter of this volume, strong positive selection also drives Tajima's D to less than -2. As is often the case in population genetics, distinct evolutionary scenarios produce the same pattern of genetic variation. On a more hopeful note, we can sometimes find an additional layer to the genetic pattern that allows us to go further with our inference. Returning to our example, both a population bottleneck and selective sweep can produce a significantly negative value of Tajima's D. On average, the former will be observed across the whole genome, while the latter will be limited to a region proximate to the molecular target of natural selection.

Given the nosiness of our summary statistic clues, one inferential tactic is to create as many unique ways of summarizing the genetic data as possible in the hope that a legion of summary statistics will somehow collectively capture the nuance of a sequence alignment. Superficially, the recent use of convolutional neural networks (CNN) on input "images" of sequence alignments seems to obviate the need for summary statistics (Flagel et al. 2018). Of course, the CNN is learning high-dimensional summaries of the data that we would never imagine natively (i.e., as humans). So summary statistics are still in play, but they are of the multidimensional chess variety.

Nevertheless, there are summary statistics that intuitively should capture aspects of the sequence alignment not tracked by π or θ_W. For example, how many unique haplotypes are found in a sample? The answer is the summary statistic K. Each of the K haplotypes has a frequency in the sample.

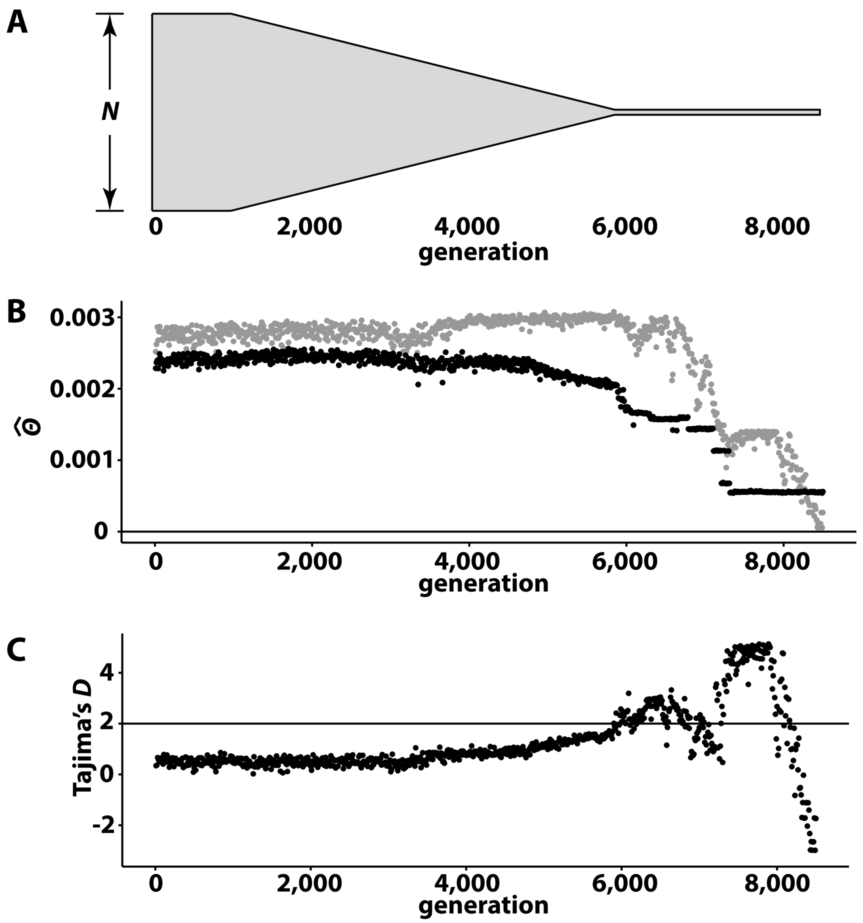

Fig. 4.5 Modeling demographic change: a gradual, linear decline over the course of 4000 generations. (**a**) The demographic model: a population of $N_e = 50,000$ diploid individuals at mutation-drift equilibrium is simulated using MS; forward simulation begins with a further 1000 generations of equilibrium followed by a linear decline lasting 4900 generations in which N_e declines by ten individuals each generation. At this point, $N_e = 1000$ diploid individuals and a further 4600 generations are simulated. Results of simulation of this model include (**b**) the per-site values of θ_W (black dots) and π (gray dots) as well as (**c**) Tajima's D

The summary statistic K is the discrete frequency distribution of each of the K haplotypes. Sometimes, variant-specific summaries such as observed and expected heterozygosity, or their difference, can prove valuable.

Successful inference often requires us to act creatively, conjuring up new summaries of the data as well as creating compound summary statistics (such as Tajima's *D*) that contrast individual summaries of the data. Imagine an evolutionary scenario and a sequence alignment sampled from a population. What other summary statistics can you come up with that might facilitate insight into the population's evolution?

References

Flagel L, Brandvain Y, Shrider D (2018) The unreasonable effectiveness of convolutional neural networks in population genetic inference. Mol Biol Evol 36:220–238

Hudson RR (1990) Gene genealogies and the coalescent process. In: Futuyma D, Antonovics J (eds) Oxford surveys in evolutionary biology, vol 7. Oxford University Press, pp 1–44

Staab PR, Metzler D (2016) Coala: an R framework for coalescent simulation. Bioinformatics

Tajima F (1989) Statistical method for testing the neutral mutation hypothesis by dna polymorphism. Genetics 123:585–595