**algorithm** wf\_sims\_with\_mut **is**

**input** *mutation\_rates* A, B

*fitness\_landscape* W

*population\_size* N

*Generations* G

*Replicates* R

(Note that A and B are 3x3 matrices where matrix element aij/bij is the rate of mutation event i -> j on locus a/b)

(Note that W is a 3x3 matrix where matrix element wij is the fitness of genotype ij)

**Output** *frequencies* F

(Note that F is a Rx9 vector where component fij is the frequency of jth genotype at the end of ith replicate; j belongs to ordered array [LL,L0,LH,0L,00,0H,HL,H0,HH])

(Note that genotypes [LL,L0,LH,0L,00,0H,HL,H0,HH] represented by integers [0,1,2,10,11,12,20,21,22])

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**function** count **is**

**input** *population* P

**output** *frequencies* F

**initialize** *frequencies* F

**set** F to 1x9 array of counts of p in P where p belongs to ordered array [0,1,2,10,11,12,20,21,22]

**return** F

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**function** reproduce **is**

**input** *population* P

*fitness\_landscape* W

**output** *population* P

replace each element p of P with w(p) many copies in P

// w(p) is fitness of individual p

**return** P

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**function** mutate **is**

**input**  *population* P

*mutation\_rates* A, B

**output** *population* P

**initialize** *number\_of\_mutations* U, V

**initialize** *number\_of\_mutations* U, V

(Note that E and F are 3x3 matrices were matrix element Uij/Vij is the count of mutation event i -> j on locus A/B)

**set** U, V to 3x3 array of counts of mutation events from Poisson distribution

// The Law of Rare Events allows us to use Poisson distribution

pick Uij/Vij random individual from P and change 1st/2nd digit from i to j

**return** P

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**initialize** *frequencies* F

**set** F to Rx9 array of zeros

**for** r := 1 to r := R step 1 **do**

**initialize** *population* P

**set** P to 1xN array of elevens

// individuals are elements of P

**for** g := 1 to g := G step 1 **do**

P := reproduce(P)

// updates P after reproduction

// size of P changed to sum of fitness of all individuals

P := mutate(P)

// updates P after mutation events

// size of P stays unchanged

P := pick N random elements from P

// updates P after random genetic drift

// size of P changed to N

// end of a generation

**end**

F(r):= count(P)

// returns genotype frequencies in P

// F(r) is rth row of matrix F

// end of a replicate

**end**

**return** F

// end of algorithm

**end**