

Figure 1: overview of the PfLOME layout and relative containment of the environments

**CLASSES**

Notation: camelcase with capitalization on the first letter denotes the class, and camelcase with a lower

case first letter denotes an instantiation of the class (i.e. ImmuneState is the class, immuneState is the

particular immune state within a human object)

**PfPedigree**

The PfPedigree contains a list of attributes of every clonal lineage of Pf parasite, specifically the

pfid, genotype, phenotype, a list of Human ids (ixh) of every Human infected with this specific clonal

variant, the times of the beginning and end of each Human infection, as well as a list of Mosquito ids

(ixm) and times of infection (they are assumed to be infected for the duration of their lives)

The pedigree is referenced at the creation of a new Pf object – the ids of the micro- and

macrogametocytes are used to look up the parent genotypes, which are then recombined following our

genetic algorithm to create the genotype of the new clonal variant.

Fields:

gtype – list of real-valued vectors within the unit n-cube ordered by pfid; vector is of

length nAntigenLoci

ptype – list of integer-valued vectors with component values for component i contained

in (1,nptype[i])

ixh – list of human ids infected by a particular parasite clonal lineage with given pfid

ixm – list of mosquito ids infected by a particular parasite clonal lineage with given pfid

mac – list of ids of macrogametocyte lineage for a given pfid

mic – list of ids of microgametocyte lineage for a given pfid

nAntigenLoci – integer, number of antigens we’re considering in the genotype vector

nptypes – integer-valued vector, number of distinct phenotypes for each of the antigens

of interest

PedLength – integer, current number of distinct clonal lineages on the part of the

landscape the pfPedigree is instantiated in

sib – sibling type (may not be used)

th – list of times a particular human was infected by a clonal lineage with given pfid

thEnd – list of times a particular human cleared a clonal lineage with given pfid

tm – list of times a particular mosquito was infected by a clonal lineage with given pfid

tmEnd – list of times a particular mosquito died/cleared a clonal lineage with a given

pfid

Methods:

initialize( ) – called automatically at instantiation of object, sets default values for

nAntigenLoci and nptypes

get\_\*field\*( ) - accessors for each of the above fields, no input necessary. Output is the

value of the field listed in the method

**Human**

The Human class acts as an environmental container for the immuneState, healthState, and

Pathogen objects. To instantiate a human object, one needs to input the unique human id (ixH), the

human’s age, the sex (M for male, F for female), and the location id. Creating a human object also

queues the creation of a set of internal fields that are instantiations of ImmuneState, HealthState, and

Pathogen (see diagram). Any updates can be queued in the Human, and the cross-talk between the

different internal objects are done through function calls in the Human with inputs using values

accessed within the Human.

Fields:

ixh – unique human id

age – age of human

sex – sex of human (‘M’ for male, ‘F’ for female)

locH – current location id of human

immuneState – immuneState object, see ImmuneState class description

healthState - healthState object, see HealthState class description

pathogen - pathogen object, see Pathogen class description

history – list of past values for various fields

Methods:

initialize( ixh, age, sex, locH ) – initialization function, sets id, age, sex, and location to

given values at declaration

get\_\*field\*( ) – accessor method that returns value of a given field

set\_\*field\*( newFieldValue ) – setter method that replaces field value with a new given

value

infectHuman( t, pfid ) – infect human with a given pf object listed in the pfPedigree

clearPathogen( t, pfid ) – remove the pf object when densities go to zero

infectMosquito( t, pfid, ixm ) – infect mosquito with a given pf object listed in pfPedigree

moveHuman( newlocH ) – move human to new location

updateHuman( t ) – updates human given current time t – updates all internal fields,

including the health/immune states and pathogens

**ImmuneState**

An immuneState object contains lists of immune counters for general immunity, as well as an

array of times when a particular phenotype was last seen within the human.

Fields:

nBSImmunecounters – number of different blood-stage immune counters

BSImm – current value of each blood-stage immune counter

wx – vector of waxing rates for each blood-stage immune counter

wn – vector of waning rates for each blood-stage immune counter

dxp - parameter that weights effect of phenotypic difference on type-specific immunity

dtp – parameter that weights effect of temporal difference on type-specific immunity

GenImm – value of total general immune effect of all the counters

ptypesTime – matrix/array of previous time(s) a phenotype was seen by the human

typeImm – overall effect of type-specific immunity to a given type

nptypes – vector of integers, number of phenotypes – read from pfPedigree

history – list of all previous values of the fields

Methods:

initialize( ) – initializing function, automatically called when object is instantiated

get\_\*field\*( ) - accessor method that returns value of a given field

set\_\*field\*( newFieldValue ) – setter method that replaces field value with a new given

value

update\_immuneState( t, Ptot ) – updates all fields of the immune state given current

time and total parasite density

update\_history( ) – updates the historical record of field values

sigmoidX( X, X50, Xs, atMax ) – sigmoidal function with independent variable X,

midpoint at X50, slope parameter Xs, and initial maximum achieved at value

atMax

dynamicXdt ( X, P, PAR ) – Determines the change in immune counters – tracks a moving

equilibrium value determined by parasite densities

daysSinceUnder( X, P, PAR ) – days since under some immunogenic threshold of asexual

parasite number – used to calculate the new immune counter value

daysSinceOver( X, P, PAR ) – days since over some immunogenic threshold of asxexual

parasite number – used to calculate the new immune counter value

gImPAR( wx, wn, P50, Ps, atMax, b, sigma ) – function that collects the immune

parameters and stores them in a list which is referenced in other immune

counter functions

ptype2Mat( ptype, nptypes) – takes phenotype vector and converts the information into

a stochastic binary-valued matrix – rows correspond to the different antigen

loci, columns correspond to the variants within a locus. A 0 entry means that

phenotype was not presented at that locus, 1 means it was. This matrix, when

multiplied by the time of presentation in the host, represents the most recent

time each of the phenotypes were seen by the host

shift( v, places, dir = ‘right’ ) – function that shifts a vector v a certain number of places

in the ‘dir’ direction – dir takes character values ‘right’ and ‘left’, with default

being ‘right’

update\_typeImmunity( t, ptype ) – updates the type-immunity-related data structures,

specifically the array containing the past times a particular phenotype was seen

by the host

crossImm( ptype, nptypes ) – calculates the effects of cross-immunity between types;

using the phenotype presented and the last time that phenotype and similar

phenotypes were seen, it calculates the total effect of the current type-specific

immune response to a given new type

**HealthState**

Class instantiated within a human object that keeps track of many clinically-important fields

such as fever, biomarkers, anemia, and medications. It acts as the switch that drives of individuals to

seek care and determines whether/how they get treated.

Fields:

Fever – current fever state (either binary or actual temperature)

feverThresh – threshold Ptot above which a fever response will be induced

HRP2 – current log10 HRP2 levels

RBC – current RBC count

pLDH – current log10 pLDH levels

Rx – list of possible drugs a person can take

history – list of all previous values of the fields

Methods:

initialize( ) – initializing function, automatically called when object is instantiated

get\_\*field\*( ) - accessor method that returns value of a given field

set\_\*field\*( newFieldValue ) – setter method that replaces field value with a new given

value

update\_healthState( Ptot, RBCHist ) – calls for an update from the human level within

the updateHuman method; calls within the healthState object an update in

Fever, HRP2, pLDH, and RBC counts given current total asexual parasite

densities and a rolling record of RBC for the DDE Lasota model of RBC

production and hematopoiesis

update\_Fever( Ptot ) – updates fever status; compares Ptot to feverThresh, and initiates

a fever if feverThresh is exceeded or ends a fever if it is below. Can change to be

a queued event

update\_HRP2( Ptot ) – updates HRP2 levels based on current total asexual parasite

numbers

update\_pLDH( Ptot ) – updates pLDH levels based on current total asexual parasite

numbers

update\_RBC( Ptot, RBCHist ) – updates the red blood cell count given total asexual

parasite numbers for RBC bursting/lysis, and a rolling record of past RBC to

determine the current rate of hematopoietic stem cell terminal differentiation

update\_history( ) – updates the history for each of the fields in the healthState object

RDT( ) – runs a rapid diagnostic test using HRP2 levels; gives a binary result from a

Bernoulli random variable with parameter p, based on a probability of testing

positive as a (monotonic increasing) function of current HRP2 levels

HSRDT( ) – highly sensitive rapid diagnostic test; same as RDT, but will have a lower

threshold of detectability (more accurate)

PCR( ) – Polymerase Chain Reaction; used to detect falciparum DNA. Gives a binary

result from a Bernoulli random variable with parameter p, based on a

probability of testing positive as a (monotonic increasing) function of current

parasite counts (10^Ptot+10^Gtot)

LAMP( ) – Loop-Mediated Isothermal Amplification – same as PCR, but will have a higher

threshold of detectability (less accurate)

LightMic( ) – Light Microscopy – gives a binary result from a Bernoulli random variable

with parameter p, based on a probability of testing positive as a (monotonic

increasing) function of current parasite counts (10^Ptot+10^Gtot)

sigmoidX( X, X50, Xs, atMax ) – sigmoidal function with independent variable X,

midpoint at X50, slope parameter Xs, and initial maximum achieved at value

atMax

Treat( Rx ) – Calls for the administration of an particular treatment (given as Rx) to be

added to the human eventQ

**Pathogen**

Class instantiated within the human object that acts as a container for particular pathogens in

the human, as well as keeping summary information about the cumulative burden of all the pathogens

such as total parasite densities and multiplicity of infection. The healthState and immuneState objects

inside the human interact with the pathogen object through these infection summary fields. Each

pathogen will be held in a list with their class name, i.e. pf objects are held in the PfPathogen list.

Fields:

PfPathogen – list containing pf objects (see Pf class description) that classify current pf

infection state within human

Ptot – sum of all individual asexual clonal lineages currently infecting human

Gtot – sum of all individual gametocyte clonal lineages currently infecting human

PfMOI – current multiplicity of infection within human object (i.e. sum of activeP’s)

history – list of previous values of the fields

Methods:

initialize( ) – initializing function, automatically called when object is instantiated

get\_\*field\*( ) – accessor method that returns value of a given field

set\_\*field\*( newFieldVal ) – setter method that replaces field value with a new given

value

add\_Pf( t, pfid) ## currently include more inputs, but can look up from pfped

adds a pf object into the pfpathogen list in the human, and uses the

phenotypic data as well as the tent parameters to determine the natural

course of the infection

remove\_Pf( t, pfid ) – removes the pf object when both activeP and activeG are zero

update\_pathogen( t ) – updates all pf objects inside the pfpathogen list in the pathogen

object of a human

update\_Ptot( t ) – updates the total count of asexual parasites in a human, summing

over all active infections (activeP=1)

update\_Gtot( ) – updates the total count of gametocytes in a human, summing over all

active infections (activeG=1)

update\_history( ) – updates the history list in the human; concatenates current values

with all historical values

log10sum( x ) – internal method that takes the base-10 exponential value of each

component in a vector x, sums them while removing any NaNs, and takes the

log10 of the result; used to compute Ptot and Gtot from individual log10 counts

of the current active infections

log10vals( x ) – internal method that removes the NaNs from a vector x and takes the

log10 value; called within the log10sum method

**Pf**

Plasmodium falciparum object – kept track of in the pedigree and contained in PfPathogen list

when a human is infected. This contains all of the parameters that dictate the natural course of

infection, which will then be modified by the immuneState within an infected human.

Fields:

pfid – unique identifier of the parasite clonal lineage

PAR – set of parameters, right now corresponding to tent function parameters (time

initiated, initial merozoite numbers released from liver, time and height of peak

asexual parasite numbers, and time of parasite clearance

gdk – death rate of gametocytes (based on “half-life”/clearance rate)

activeP – binary value – 1 if asexual parasite numbers of a particular clonal strain within

the human host are nonzero, 0 if they’ve been cleared

active – binary value – 1 if gametocyte numbers of a particular clonal strain within the

human host are nonzero, 0 if they’ve been cleared

Pt – asexual parasite numbers

Ptt – by default a 10-day rolling record of previous Pt values, used to simulate the delay

to correspond with the Gt maturation period

Gt – gametocyte parasite numbers

mic – microgametocyte id, used to look up the microgametocyte genotype in the

pfPedigree when calculating the new genotype for a pf

mac – macrogametocyte id, used to look up the macrogametocyte genotype in the

pfPedigree when calculating the new genotype for a pf

gtype – vector of length n=nAntigenLoci within the unit n-cube, genotype of the clonal

lineage

ptype – integer-valued vector of length n=nAntigenLoci, computed using the gtype

mu – parameter dictating the per-locus mutation rate; can be a single constant or a

vector showing the mutation rate at a given locus

Methods:

initialize( mic, mac, pfid, seed=F ) - initializing function, automatically called when object

is instantiated – sets mic, mac, pfid. If seed=F, will use mic/mac to create new

gtype; otherwise it will create one at random to help ‘seed’ the pf population

get\_\*field\*( ) - accessor method that returns value of a given field

set\_\*field\*( newFieldVal ) – setter method that replaces field value with a new given

value

getGtype( mic, mac, mu, seed=F ) – calculates a new gtype for a parasite given mic and

mac pfid, and a per-antigen-locus mutation rate. If seed=F, it uses the mic and

mic genetic data; otherwise it chooses a genotype at random

getPtype( gtype, nptypes ) – calculates a phenotype given the genotype and a vector of

the number of phenotypes associated with each locus

mutate( gtype, mu ) – determines the mutational effects during a meitotic event given a

genotype and a per-antigen-locus mutation rate

update\_Pf( t ) – calls the update methods for Pt, Ptt, Gt – usually called from the

pathogen object, which is part of an update\_pathogen method from the human

level

update\_Pt( t ) – updates the asexual parasite population – default uses dPdt\_tent, a

dynamically updated tent function

update\_Ptt( t ) – updates a historical list of asexual parasite population – shifts the list

back by a day and adds the new value. By default holds 10 days of history. This is

used to update Gametocyte populations

update\_Gt( t ) – updates the gametocyte population using the rolling record of asexual

parasite densities Ptt

GamCyGen( t, P, PAR ) – gametocytogenesis, called by the update\_Gt method to

determine the relationship between the rolling record of asexual parasite

densities and gametocyte creation

Pf.MaxPD( N, mn, vr ) – draws value of the maximum asexual parasite number from a

set distribution (normal distribution by default)

Pf.PeakD( min ) – draws the value of the time of the maximum asexual parasite number

from a set distribution (modified log-normal distribution by default)

Pf.MZ0( ) – draws the value of the initial number of asexual parasites emerging from the

liver from a set distribution (normal distribution by default)

Pf.Duration( peakD, N, mn ) – draws the value of the duration of the infection (in days)

from a set distribution (geometric distribution by default)

tentPAR( t, pfid ) – method that calls the above four methods to determine the tent

parameters for the parasite, calculates the derived parameters (growth and

decay rates of the tent), and stores them in a list

gr\_tent( t, PAR ) – determines if, at a given time with given tent parameters, the tent

should be increasing or decreasing

dPdt\_tent( t, P, PAR, PD=0, IM = 0 ) – Using the tent parameters and the gr\_tent

method, determines the next value of parasite densities (as also modified by

pharmacodynamic effects of drugs and the current immuneState)

shift( v, places, dir=”right”) – as in the ImmuneState class, function that shifts a vector v

a certain number of places in the ‘dir’ direction – dir takes character values

‘right’ and ‘left’, with default being ‘right’

log10sum( x ) - internal method that takes the base-10 exponential value of each

component in a vector x, sums them while removing any NaNs, and takes the

log10 of the result; used to compute Ptot and Gtot from individual log10 counts

of the current active infections

log10vals( x ) - internal method that removes the NaNs from a vector x and takes the

log10 value; called within the log10sum method

immuneMod\_tent( BSImm ) – calculates the blood stage immune counters’ modulatory

effect on the tent parameters of a particular pf parasite

sigmoid01( x, xh, b, max ) – sigmoidal function with domain in the unit interval [0,1] with

independent variable x, halfway point xh, slope parameter b, and max value of

sigmoid max

**Typical Pf lifecycle in PfLOME:**

1. Mosquito probes host and bites an infected human
2. Gametocytes are sampled from the human, weighted by their relative Gt numbers
3. The mic and mac ids are retrieved, looked up in the pfpedigree, and a new pf object is

created using the mic and mac gtypes

1. \*\* current idea – a similar tent function is queued in the mosquito that is delayed from the

biting event corresponding to sporozoite densities in the mouth of the mosquito – only the

pfids of the new pf objects are stored in the mosquito

1. When the mosquito bites another human, n pfids are sampled with relative weights of their

sporozoite values – the n is randomly drawn with a mean increasing with sporozoite

densities

1. This queues a tent function in the human of asexual Pt, and corresponding dynamics of

gametocytes Gt, biomarkers, and immune counters

1. The tent function parameters are modified at the start of the new infection by the current

levels of immune counters

1. Fever events are currently tied to parasite densities – if Pt exceeds feverThreshold, then this

can queue care-seeking behavior and increases probabilities of diagnostic test taking/Rx

administration to curb the densities

* 1. Rx has a pharmacodynamic (PD) daily log10 killing effect specific to each class of

drug

1. If the human is bitten again while they have a nonzero number of gametocytes, go to step 2

**Example of use (walkthrough of object\_practice.R)**

1. Create a pedigree on the tile:

pfped = PfPedigree$new( )

This instructs the PfPedigree class to create a new object we call pfped – no input is required for

this instantiation

1. Populate your human population:

someGuy = Human$new( ixh = 1, age = 20, sex = ’M’, locH = 7)

This creates an object from the human class with an id of 1, sets their age to be 20 years old,

their sex to be male, and puts them in the location with id 7. This automatically initiates the

instantiation within the newly created human environment objects called pathogen,

immuneState, and healthState from their respective classes

1. Populate your parasite population:

pf = Pf$new( mic = 1, mac = 1, pfid = 1, seed = T)

This creates an object from the parasite class with a pfid of 1; seed = T means we’re seeding our

population, so it creates a genotype randomly without using any prior data from mic and mac

information, but values are still required for now. If seed = F (which is the default) then mic and

mac ids will be used to look up the parents’ genotypes, and the new genotype will be calculated

from those. Tent function parameters are also drawn at instantiation of the object, which will be

modified by the current immune state of whichever human host it eventually infects.

Importantly we must also add the newly created pf object to the pedigree:

pfped$add2Pedigree(pf)

This uses the pedigree’s method and takes the pf object as input to store its information – it can

be referenced as necessary by any other object, and some components (such as when a human

is infected by that object) can be modified.

1. We can infect someGuy from the human population at time t with our pf:

pfid = pf$get\_pfid( )

someGuy$infectHuman( t, pfid )

First we accessed our pfid through the publicly accessible method in the pf object. Then we used

the current time and the pfid of the pathogen to infect the human; both the t and ixH will be

recorded in pfped, and a tent function will be queued in the human corresponding to the

asexual parasite population.

1. We can update the infection status of the human (including Pt, Gt, biomarkers, fever, immune

markers, etc) through the command

someGuy$updateHuman( t )

This uses the current time to update Pt through dPdt, and all of the other components are

updated in turn. The history for each internal object is also updated automatically, so we have a

rolling list of every past value of each field of interest.

1. Finally we can plot the results after the human object has been modified by accessing the

history list inside the human. For example, if we want to plot Pt over time, we can plot it

through the following command:

history = someGuy$get\_history( )

Pt = history$Pt

plot( time, Pt )

This extracts the history list from the human, then extracts Pt from history, then plots it; of

course this can also be done on a single line:

plot( time, someGuy$get\_history( )$Pt )

Quantities of interest available to plot:

Ptot

Gtot

PfMOI

Fever

HRP2

pLDH

RBC

BSImm

GenImm

**Descriptions of Modules**

**Tent Module**

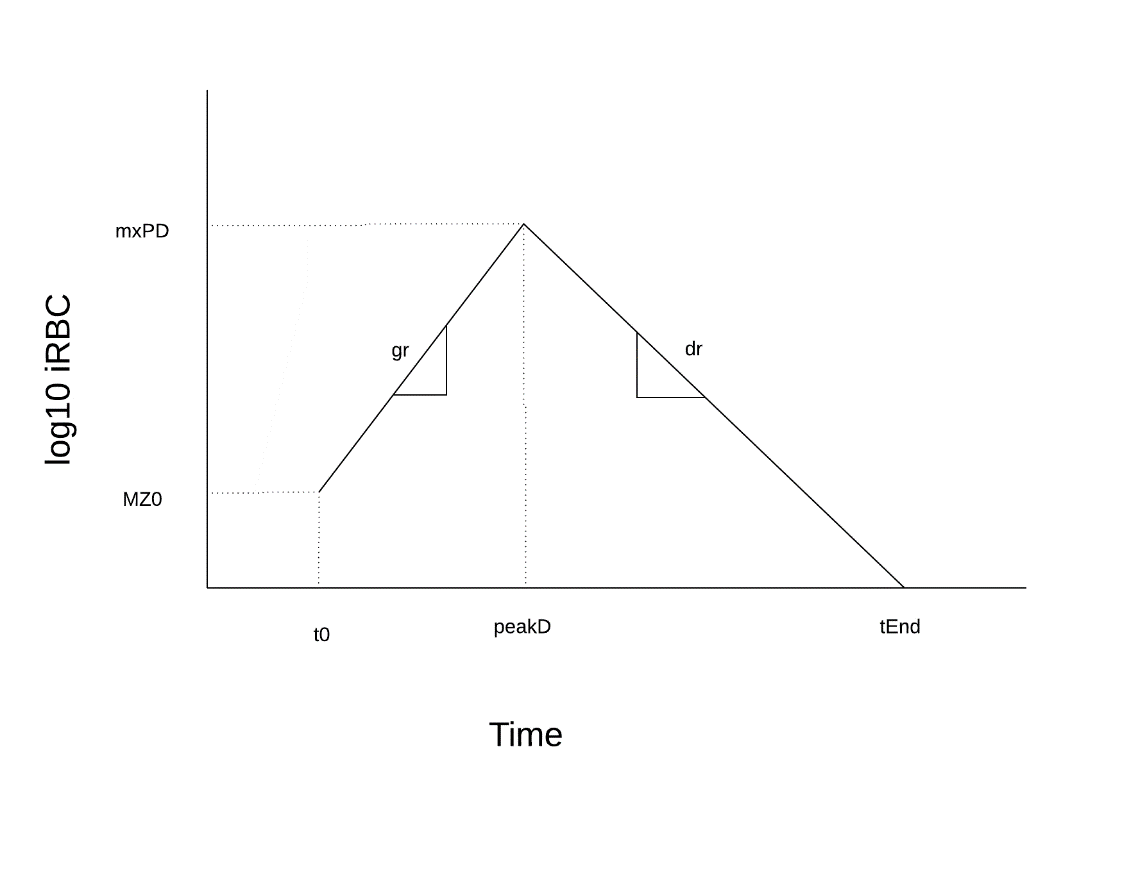


Figure 2: example tent function with parameters labeled

Description

log10 profile of a particular infection. The biggest underlying assumption is that the infection is

uncomplicated and is eventually resolved naturally. All immune effects are expressed through a

modification of the four core parameters (MZ0, peakD, mxPD, tEnd) that act as necessary and sufficient

summaries of a particular tent. Possible variants on this module include a “noisy tent” with

appropriately scaled noise to the tent function, and (in the case of vivax) a tent function with regular

oscillations about the baseline.

Fields

MZ0

peakD

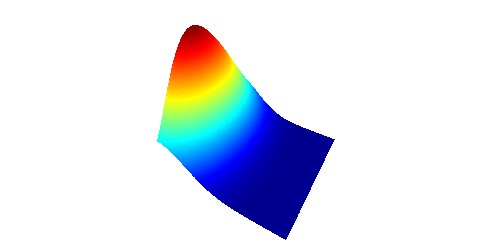
mxPD

tEnd

Methods

dPdt\_tent

**Gogg Type-Specific Immunity Module**



Description

This module is based largely on the immunity model presented in Julia Gogg’s paper,

“Dynamics and Selection of Many-Strain Pathogens”. Immune modulatory effects act (once

normalized through a sigmoidal translation) on the infection profile parameters. As an example

with the tent module, the parameters (MZ0, peakD, mxPD, tEnd) will all be decreased with

increased immunity. Immunity is acquired when humans are encountered with a new infection,

and wanes over time once the infection is cleared. Similar “adjacent” phenotypes in the

phenotypic space interact through cross-immunity weighted by how antigenically distant they

are; cross immunity should be a monotonically decreasing function of time since last presence

of a particular phenotype and antigenic difference. Currently we use a one-dimensional model

of antigenic space, and the effects decrease exponentially in both dimensions.

Fields

[associated fields]

Methods

[associated methods]

**Mendelian Genetics Module**

Description

The Mendelian inheritance module assumes no linkage in any of the genes; The

genotype at each locus is inherited with equal probability from either of the parents,

and each is chosen independently. The loci are represented in the one-dimensional case

as a number between 0 and 1. Each locus also has some set probability of a mutation,

which by default is a uniform movement of the genotype to another part of the unit

interval.

Fields

[associated fields]

Methods

[associated methods]

PDG Version

Due to the complexity of the underlying dynamics of malaria transmission we have allowed for

MASH to be flexible in the available options – ranging from the simplistic PfSI model to the exquisitely

detailed full PfLOME suite of models. However many combinations of components may result in

qualitatively similar or even identical behaviors, and policy questions are generally agnostic to the

underlying algorithm. In addition, often the practical gains from pursuing an optimal model are marginal

(if they are even possible) with the available data. This led us to pursue a middle-ground standard set of

modules that we collectively call the Pretty Damn Good (or PDG) model. It is intended to be the simplest

implementation that is complex enough to include the most prominent features of the system that

aren’t adequately described by the standard Ross Macdonald models. The PDG version of PfLOME

should be calibrated with local data as much as is possible for a particular setting, and will be updated

iteratively as more data is made available.

Current component ideas for PDG:

Pf

* Gtypes/ptypes if known, or as proxies for resistance types if necessary; tent

parameters & distributions

HealthState

* Biomarkers (HRP2), diagnostic test(s) (RDT, microscopy), treatment(s) with at least 2 regimens
  + Adherence rates should most likely either be a measured quantity or measured in a “comparable” area…?

ImmuneState

* # of BSImmuneCounters will be set and calibration will be conditioned on them; the

number will be augmented and calibration will run again. Ideally finding trade

off between best fit and fewest counters

* Type-immunity is some proportion of the overall immune strength, p in [0,1]

(however, type-immunity requires prior knowledge of genetic makeup of

parasite population at the beginning of the simulation) – needs to be

implemented in areas where resistance is a large factor