TITANdoc Documentation

Release 0.0.1

Maximilian King

Nov 05, 2018

Contents

This document is the reference documentation for the Treatment of Infection and Transmission in Agent-based Networks (TITAN) simulation package developed by Marshall Labs at Brown University (see TitanModel.org)

The program is developed and runs using Python 2.7 on all OS platforms. The Python bindings are included in the `titanmodel` package, but require, additional modules to be installed, namely `numpy` and `networkX`.

Apart from this document, there exists some additional documentation as well:

- Documentation of the program code itself can be found here: code documentation.
- iPython notebook examples: available on GitHub

The development source code of the binaries can be found on GitHub. The TITAN installers, as well as source code packages for the program, can be found in the 'programs' directory.

In case you're interested in running simulations from R, you'll need to have a working Python version installed as well. For MS-Windows this is typically not installed by default; when installing this it's best to use the default directory (e.g. `C:\Python27` or `C:\Python34`).

CHAPTER 1

Contents

1.1 Overview

1.1.1 What is TITAN?

TITAN (Treatment of Infectious Transmissions through Agent-based Network) is an agent-based simulation model used to explore contact transmission in complex social networks. Starting with the initializing agent population, TITAN iterates over a series of stochastic interactions where agents can engage in risk behavior with one another, transmit infections through various medium, and enter and exit the care continuum. The purpose of TITAN is to evaluate the impact of treatment models on incidence and prevalence rates of the targeted disease(s) through the use of data fitting simulated trajectories and rich statistics of primary/sub-population attributable proportions.

TITAN is a has a sophisticated social network structure which allows for various forms of care cascade, interventions and barriers, and dynamic agent parameters such as:

- Stocahstic enrollment/discontinuation of treatment as prevention strategies such as ART (antiretroviral therapy), PrEP (Pre-exposure prophilaxis), and VCT (voluntary testing)
- High risk group behavior such as sex workers, injection drug use, or post-incarceration behavior
- Variable transmission probability based on mode of transmission (anal, vaginal, needle, etc.) and viral load (acute, AIDs, virally supressed)
- K-nearest neighbor partnering algorithm finds partners based of fitness rather than randomness
- Heterogeneous agent populations including dynamics such as: racial disparity, sexual orientations (MSM, WSW, bisexual), sexual activity and promiscuity (main/casual/one-ofs), and assortive age mixing.
- Disruptive barriers in form of incarceration or migration

Agent populations are defined as graphs (nodes connected by edges). Nodes in the graph are used to represent the attributes (or collection of attributes) of an agent (person), and edges define the type of relationship between agents. In practice, a graph represents a social network of connected people through various relationship types, and provides the medium for which agents can interact.

TITAN is developed using the Python 2.7 (https://www.python.org/download/releases/2.7/) programming language. TITAN also implements standard numberical libraries Numpy (http://www.numpy.org/) and SciPY (https://www.scipy.org/), as well as a set of Python v2 bindings for The Qt Company's Qt application framework PYQT (https://wiki.python.org/moin/PyQt) for graphical user interfacing.

To promote efficiency and minimal computational effort, TITAN uses the python library NetworkX (https://networkx.github.io/) to create network graphs and perform network analysis/vizualization such as connectivity measures or Fruchterman-Reingold force-directed diagrams.

TITAN's Developement website: http://titan-simulation.org/

Version History

v1.0 February 2017: Initial Release

1.1.2 Agent-based Modelling

A sub-class of complex systems methods, agent-based models (ABMs) are individual-based models and are increasingly commonplace in studies investigating the social and structural determinants of population health. In brief, an ABM consists of agents (i.e., ''nodes'') who are connected to each other by links (i.e., ''edges''), through which information, behaviors, or some other social process can be transmitted. Every agent's internal state (e.g., HIV infection status and disease stage) are updated at each discrete time step based on pre-programmed rules and interactions with other agents.8 In this model, links represent sexual and injection-related risk behavior between two agents through which contact infectious transmissions can occur. In this model, links represent sexual and injection-related risk behavior between two agents through which contact transmission can occur.

The driving forces of the model itself is a connected tree of components which have orders of influence on transmission events, as seen in the figure below The risk behav- iors, such as the number of interactions an agent engages in with its partners, are the main source of transmission events of the modeled infection. In this model, agent de- mographics (dfined by the input parameters of the sim- ulation) dominate the risk behavior of a particular agent, and also determines the care continuum of the agent in its environment. The agents' risk to transmit is damped by external preventative measures that are engaged in each component of the model. Testing of the agents can lead to a direct reduction in risk behavior (an agent aware of their infection status may engage in less risk acts or partners), and also cascades downwards into direct pre- ventative measures such as condoms or safe needle ex- changes (for PWID). Additionally, agents may enroll in transmission preventing programs or treatments, classified as the care continuum, which also directly inuences the agents risk behavior and transmissibility. These components work in concert to provide the overall dynamic of the simulation, and we continue to employ complexities that interact with these components in either preventa- tive or disruptive impacts.

//Fig1

1.1.3 Who is TITAN for?

TITAN is primarily intended for academic research of ...

1.2 Components

The population of TITAN is maintained as a graph structure containing nodes (agents) and relationships between agents (edges). In the model, an agent class is defined where any given agent maintains its own information within the data structure. These include variables such as the agents age, demographics, partner tree, infection status, or enrollment in treatment programs. A group of agents is defined as an agent set, where agents within the set can be connected or group based on desirable variables, such as the PWID set which contains all PWID agents. These sets allow the population structure to be well organized while not mutually exclusive so as to ease the searching and sorting of the total populations sub-strata.

The model is initialized generating the total N agents which are classified based on input demographic information in a stochastic process, then placing agents in a network space. As the model proceeds in monthly time steps, links are broken and formed between agents, thus representing a dynamic sexual and injecting network. Agents leave the population when they die from infection related or other natural causes, and are replaced in a stochastic manner (i.e., the new agent does not necessarily carry the same traits as the previous agent, but are replaced according to the distributions specified during model initialization). Thus, a constant population size is maintained over the life of the simulation, where each new iteration, or time step, represents an evolution of the population forwards in time.

1.3 Agent (node)

In TTITAN, agents are individual objects that carry their own set of parameters interally allocated regardless of the process or structure that may govern those parameters. That is to say that each agent maintains their own individual values for each agent parameter even if they were assigned by a global or generalized function. As an example, the probability for testing may be globally set to 20% chance per timestep for every agent in the model, but that value is specifically assigned to each agent upon initialization rather than drawing the value from a global parameter at each testing opportunity.

Therefore, agents carry with them a suite of relationship, environmental, bahvioural, and demographical parameters, some of which are static while others are dynamic and can change.

1.3.1 Parameters

Parameters are initially assigned to an agent upon created based on input distribtutions calculated by empirical data sources.

Static

The following parameters are assigned upon agent creation and are statically defined (ie: cannot change over the span of a single simulation iteration)

```
ID                  - Agents unique identification number
parent              - Parent of agent (for inheritence)
initAge             - Agents starting age
race                - Race identifier for agent
gender              - Gender of the agent
DU                  - Drug using status (NDU, NIDU, IDU)
SO                  - Agents sexual orientation
sexualRole          - Agents sexual role (inserive, receptive, versatile)
```

These parameters serve as agent-defining parameters that are used to influence a host of other dynamic parameters (shown below).

Dynamic

The following are baseline parameters that are assigned upon agent creation and are dynamic throughout the a simulation run (ie: change over the span of a single simulation iteration). They can be broken into four main categories:

Behavioral

Behavioral parameters influence the agents interactions with other agents. These factors include high-risk behavior, which indicate elevated exposure to “risk-acts” where transmission can occur (unprotected sex or needle sharing drug-

use). They also include the agents desirable number of partners for each transmission mode, as well as the number of risk-acts per partner.

```
age                     - Agents instantaneous age
annual_PNtar_SE         - Target number of annual sexual partners
annual_PNtar_DU         - Target number of annual drug-use partners
annual_RAtar_SE         - Target number of annual sexual risk acts (RA) per partner
annual_RAtar_DU         - Target number of annual drug-use risk acts per partner
HR_bool                 - Boolean for if agent is high-risk (HR)
HR_time                 - Time agent has been high-risk
everHR_bool             - Boolean for if agent was ever high-risk during lifetime
```

Infection Status

These parameters include flags and times associated with various stages and types of infection. As the TITAN model is primarily HIV focused, HIV and AIDS related parameters are available for all agents.

```
STI_pool                - List of existing co-infection STIs
HIV_bool                - Boolean for if agent is HIV infected
HIV_time                - Time agent has been HIV infected (effects acute transmission)
AIDS_bool               - Boolean for if agent is HIV+AIDS infected
AIDS_time               - Time agent has been HIV+AIDS infected
PrEPresist_bool         - Boolean for if agent has PrEP resistant HIV infection
```

Treatment Cascade

The treatment parameters related to the agents interactions within the care continuum. HIV infected agents move from either HIV undiagnosed, HIV diagnosed not on treatment, or to HIV diagnosed on treatment. Additionally, HIV uninfected agents are eligible to enroll on PrEP, a preventative treatment to reduce the risk of infection per risk act.

```
tested                  - Boolean for if agent is positively diagnosed wiht HIV
HAART_bool              - Boolean for if agent is enrolled on ART treatment
HAART_time              - Time agent has been enrolled on ART treatment
HAART_adh               - Agents adherence to ART treatment
PrEP_bool               - Boolean for if agent is enrolled on PrEP treatment
PrEP_time               - Time agent has been enrolled on PrEP treatment
PrEP_adh                - Agents adherence to PrEP treatment
PrEP_load               - Agent instantanous PrEP serum level
```

Probabilities

All probabilites are calculated at the monthly timestep as the model runs on intervals of months. These probabilites are typically calculated from empirical data based on the agents demographical/behavioral parameters, but are often influenced by additional parameters such as their infection status or status within the treatment care continuum.

```
P_HIVtest               - Probability of HIV testing
P_progAIDS              - Probability of progressing to AIDS
P_HAARTenrol            - Probability of enrolling on ART treatment
P_HAARTdisc             - Probability of discontinuing HAART treatment
P_PrEPenrol             - Probability of enrolling on PrEP treatment
P_PrEPdisc              - Probability of discontinuing PrEP treatment
P_mortality             - Probability of agent dying
```

Partnering

Partnering parameters are mostly associated with tracking active parters and relationships with those partners, as well as metrics to measure the performance of the individual agent with respect to their desired activity through both sexual and drug-use networks.

```
partners                - List of active agent partners
relationships           - List of active relationships with partners (injecting/sexual)
aN_SE_ptnrs             - Number of unique sexual partners in past year
aN_DU_ptnrs             - Number of unique drug-use partners in past year
```

Statistics

The following are internally tracked agent parameters for statistical analysis reasons. Having agents store these variables independently rather than aggregated or remotely stored allows for a direct history for each agent to be pulled and analyzed.

```
timeAlive               - Time agent has been alive for in the model
tot_SE_acts             - Total number of sexual risk-acts performed
tot_DU_acts             - Total number of drug-use risk acts performed
```

1.3.2 Agent Creation Order of Operations

The agent population is formed upon model initialization until the desired number of total agents are created as defined by model input parameters. For each agent creation, the order of operations for agent parameters is as follows:

1. Agent initialized and assigned unique identifier number (*ID*)
2. Assign race of agent (probabalistic from demographic input params)
3. Assign gender of agent (probabalistic from demographic input params)
4. Assign sexual orientation and sexual role
5. Assign drug-use status
6. Assign initial age of agent
7. Assign infection status (HIV_bool/HIV_time, AIDS_bool/AIDS_time)
8. Assign location in care continuum (tested, HAART treatment status)

As these parameters are all assigned stochastically, there is variance between initial agent populations between individual simulation runs. While this functionality can be disabled and a forced random seed can be implemented to create identical initial populations, the randomness of initial conditions can play a critical role in determining model performance, and thus is left to the discretion of the user.

1.3.3 Set Grouping

Agents are grouped into agent sets based on heirarchal families for easier sorting and searching. Each agent set inherets froms its parent set. This means that there are never agents in subsets that are not also in the subset parent set. The benefit of this inheritence is that specialized subcategories of agents may be quickly created. One example is a sex worker class, where agents of any gender can engage in sex work behavior. While the parameters of the agent may widely remain the same, the agent could have elevated sexual frequency and condom usage. This set inheritence would allow you to include a particular agent into this sex-worker class which would appropriately adjust the relevent agent parameters.

1.3.4 Functions

The following are the core functions associated with an agent class.

```
def get_ID(self):
   #Returns agent ID
   return self._ID

def partner(self, partner):
   #Partner self agent to partner agent
   assert partner in self._partners:
   raise KeyError("Partner %s is already bonded with agent %s"%(partner.get_ID(),␣
↪self._ID))
   self._partners.append(partner)

def unpartner(self, partner):
   #Removes partner from self agent partner list
   try:
       if self != agent:self._partners.remove(agent)
   except KeyError:
       raise KeyError("agent %s is not a member of agent set %s"%(agent.get_ID(),␣
↪self.get_ID()))
```

1.4 Relationship (edge)

Agents are connected to one another through a relationship network which are represented by edges on a graph. They are explicit class objects much like agents, but contain relationship specific parameters and functionality. Unlike agents, relationships are formed between agents with finite durations drawn from distributions guided by input parameters associated with the model agent population and demographics. That is to say, they are not always a fixed number of ongoing relationships at all time points unlike the total agent population size. Instead, the number of active relationships is closely tied with agent behavioral parameters (primarily that of average annual number of desired partners) and transmission network types available (sexual and/or drug-use).

1.4.1 Parameters

Parameters are initially assigned to an relationship upon created based on input distribtutions model parameters provided by the user.

Static

The following parameters are assigned upon relationship creation and are statically defined (ie: cannot change over the span of a single simulation iteration)

```
ID                    - Relationship unique identification number
partner1              - ObjectID for agent one of two
partner2              - ObjectID for agent two of two
rel_type              - Type of relationship classifier (sexual or drug-use)
race                  - Race identifier for agent
annual_RAtar          - Target number of annual risk-acts
```

These parameters serve as relationship-defining parameters that are used to influence a host of other dynamic parameters (shown below).

Dynamic

The following are parameters that are assigned upon relationship creation and are dynamic throughout the a simulation run (ie: change over the span of a single simulation iteration).

```
duration                - Remaining duration of relationship
total_acts              - Total number of acts performed in relationship (risk-act or␣
↪non risk-act)
total_RA                - Total number of risk-acts performed in relationship
HIV_bool                - Boolean for if HIV exists on either partner
sero_bool               - Boolean for if relationship is serodiscordant
```

1.5 Network

Agents are connected to form a sexual and injecting network that evolves (i.e., updates) at each time step. To construct the network, the program assigns a value K to each index agent , where K is defined as the number of partnerships with other agents per time step. The value is determined by a random sampling procedure from negative binomial distribution functions

:: Ki;t NB(p; r) = (ki;t + r 1)!)(r 1)!ki;t! pr(1 p)ki;t ki;t 2 @0

with mean given by:

:: m = pr / 1 p

for all agents per time step. Negative binomial models of partnership formation represent a search process with stopping rules, such that partners are acquired with probability until suitable partners are found. Pre vious studies have demonstrated that negative binomial distributions provide a reasonable approximation to real- world partnership networks, in which the variance of the distribution is greater than would be expected assuming constant-rate function (e.g., Poisson). We de ned negative binomial distribution functions for each class of agent based on previously published estimates of partner- ship degrees and distributions. We use the mean annual number of sexual partners reported by the input parame- ters to determine the partner acquisition and dissolution rates per monthly time step for each agent class.

The construction of the network is an iterative process in which the program precedes sequentially through the network set of N agents at each time step, matching each agent in need of partners to other in need of partners. As the network is an contact transmission risk network (not a social network), and in an effort to improve computa- tional efficiency, the model only engages links between pairs of agents who are of transmissible status and can engage in risk behavior. For example, a link between two heterosexual male agents is not permissible, unless both are PWID (i.e., they can inject together), and their partnership is serodiscordant (i.e., HIV transmission is possible). Once a new agent acquires the target infec- tion, they immediately become part of the transmissible pool and begin engaging in risk behavior. In order to ac- count for assortative mixing, the likelihood that any two agents are connected is weighted to favor links between nodes with similar characteristics by default, as described below in the pairing algorithm, but can be adjusted to fit the model needs. Specifically, the probability of contact is weighted to favor the formation of links between nodes from the same agent class. For example, PWID are more likely to connect with other PWID, while men who have sex with men (MSM) are more likely to connect with other MSM.

At each time step, agents are re-assigned partners based on a retention algorithm in order to avoid overesti- mating sexual and injecting partner turnover that would occur if agent partnerships were randomly mixed at each time step. Specifically, partners are added or removed to each index agent (i's) network according to the random value, determined from the NB distributions (described above). If the agent's target number of partners is higher than its current partnerships, the agent randomly drops links from its network. If the agent has too few partners, the agent becomes part of a set of agents which need part- nerships (NP). Once the total population is checked, we iterate over the NP set, pairing agents in partnerships us- ing the pairing algorithm, until no available matches can be made. Typically the

remaining unpaired agents in the NP pool are less than 3% of those seeking partners at any given timestep, but due to the speed and efficiency of the algorithm, this is deemed acceptable.

1.5.1 Partnering

The partner pairing algorithm uses a novel K-Nearest Neighbor searching algorithm where agents are matched with a set of Z agents within a de ned parameter space. In this algorithm, agents are placed into a S-dimensional space, where S is the total number of relevant param- eters for matching partners (age, race, ethnicity, sexual orientation, monogamy, etc). Next, for a given partner- seeking agent, we find the S-dimensional "feature vector" in this S-dimensional space between the agent of interest and agents in its local space. This feature vector rep- resents the natural extension of a Euclidian vector with additional scaling factors to in uence the weight/impact of each metric or vector indices, and is calculated as such:

```
Fij = ri  rj = ΣSn=1 Qn(in  jn)2
```

where Qn is the scaling factor for the nth matching pa- rameter. This tunable scaling factor allows for control- lable weights for each matching parameter. For example, if agents in the model aim to seek partners primarily of the same age, the age scaling factor Qage can be increased to result in an increased feature vector for agents of dis- similar ages. Conversely, if a parameter is no longer of interest for partnering, the scaling factor can be set to 0, effectively removing that component from two agents' feature vector.

This algorithm returns the n closest agents (minimum feature vectors) of the set of Z agents, then randomly chooses on to become their partner. In order to ensure the chosen agent is an appropriate match (i.e. MSM not paired sexually with a HM), the Z set must be chosen appropriately prior to executing the subroutine. More simply, if a MSM agent is seeking a sexual partner, sub- set Z would be de ned as the intersection between the Need Partner (NP) pool and the total MSM agents, thus returning the subset Z of MSM agents also needing part- ners. If this pool returns null, then our agent has no eligible matches, and is removed from the NP pool itself. In order to increase computational efficiency, rather than measuring the feature vector to all eligible partners, the agent rst searches within their local bin chosen at random from the set of S bins. This allows for a higher stochasticity in the partnering algorithm, and prevents agents from partnering only with agents in their local S space.

1.6 Interactions

1.6.1 Agent-Agent

Agents who are linked in the network can engage in sexual and/or injecting behavior. In order to increase the model's efficiency, risk behavior is only simulated be- tween serodiscordant agents at each time step.

A PWID-PWID agent dyad can engage in sexual ac- tivity exclusively, injecting activity exclusively, or both. The probability that a pair of PWID agents engage in syringe sharing was based on previously published es- timates. We assumed that undiagnosed, HIV-infected PWID agents are equally likely to share syringes as un- infected PWID agents. Moreover, we assumed that diag- nosed (i.e., HIV-infected and tested positive) PWID were 50 less likely to share syringes than undiagnosed agents, based on previous studies. For each PWID-PWID agent dyad that is selected to share syringes, the specific number of syringe sharing events in a time step was determined from a uniform distribution, with minimum value 1 (i.e., once in a month) and maximum value 30 (i.e., sharing sharing every day).

To model sexual HIV transmission, we simulated the monthly number of sexual acts and the proportion of those acts that are unprotected between each pair of connected, serodiscordant agents. We assumed that the number of sex acts per month between members of the same agent class followed a Poisson distribution. Speci - cally, at every time step, the number of sex acts for each agent was drawn from a Poisson distribution, with means provided by the input parameters. If the values for each agent in a dyad differed, the average of the two values was taken. Next, we estimated the proportion of these sex acts that were unprotected, based on previously pub- lished estimates. We considered "unprotected" to be any sexual activity in which HIV transmission can occur (no condom use, incorrect

condom use, etc.) Similarly to syringe sharing events, we assumed that HIV-diagnosed agents were 50 less likely to engage in unprotected in- tercourse with their partners, based on previously con- ducted studies.[? ?] We assumed that HIV-infected agents of unknown status engaged in sexual risk behav- ior at the same probability as HIV negative individuals.

1.6.2 Transmission

If a pair of serodiscordant agents engage in sexual or injecting risk behavior, ACTsim implements a stochas- tic algorithm to determine whether contact transmission occurs. To calculate the probability of transmission to an unin- fected partner for each type of risk behavior, we use per-act probabilities, based on previously published estimates. As shown, the risk of transmission during unprotected anal intercourse between men is greater than that during un- protected vaginal intercourse; however, we did not model partner roles among MSM (e.g., insertive vs. receptive) explicitly. Furthermore, we did not model unprotected anal intercourse between women and men.

The probabilities listed in Table ?? represent the aver- age risk of transmission during an unprotected coital act or syringe-sharing event during latent stage HIV infec- tion. Viral load is model implicitly, such that these val- ues rep- resent a mean set-point viral load (approximately 4 log10 copies/mL) in the HIV-infected population. To account for higher viral load and an increased risk of transmission during acute phase infection, we multi- plied these probabilities by a factor of 4.3 during the first three time steps following seroconversion, which rep- resents the average increase in transmission risk during acute HIV infection.[? ?] In order to calculate the over- all transmission risk per partnership per time step, we employed a Binomial process model,[?] i.e.: where is the per-act transmission probability, speci c to the type of risk activity engaged in between two serodiscordant agents (i.e., parenteral, anal intercourse, or vaginal inter- course). These probabilities were also dependent on the HIV treatment status and adherence pattern of the HIV- infected partner. The number of "trials" (total number of unprotected sex or syringe sharing events) per part- nership per time step, n, was determined for each dyad as described above.

A detailed description of our acute HIV disease model has been published previously. Following acute infec- tion (represented by three monthly time steps following seroconversion), HIV-infected agents in latent stage in- fection progress to AIDS at a rate dependent on HIV treatment status and adherence. In a manner similar to that of an ABM examining HIV transmission among MSM, we do not model CD4 count explicitly. Instead, we assigned monthly probabilities of receiving an AIDS diagnosis such that the median time to this event ap- proximated that which has been observed empirically.

1.6.3 Functions

The following are the core functions associated with an agent class.

```
def get_ID(self):
   #Returns agent ID
   return self._ID

def partner(self, partner):
   #Partner self agent to partner agent
   assert partner in self._partners:
   raise KeyError("Partner %s is already bonded with agent %s"%(partner.get_ID(),␣
↪self._ID))
   self._partners.append(partner)

def unpartner(self, partner):
   #Removes partner from self agent partner list
   try:
       if self != agent:self._partners.remove(agent)
   except KeyError:
       raise KeyError("agent %s is not a member of agent set %s"%(agent.get_ID(),␣
↪self.get_ID()))
```

(continues on next page)

(continued from previous page)

1.7 Care Continuum

1.7.1 Testing

All HIV-infected agents in a given class have a monthly probability of undergoing HIV testing. We use empiri- cal data on past-year HIV testing rates for each agent class in a setting of interest and scaled these values to obtain monthly testing probabilities. In addition to the assumption that the likelihood of HIV testing was con- stant over time, we also assumed that HIV testing has 100% sensitivity and speci city. Thus, all HIV-infected agents who undergo testing take on the attributes of di- agnosed agents. As described above, diagnosed agents are less likely to engage in HIV risk behavior with other agents.

1.7.2 HAART

Diagnosed agents are also able to initiate highly active antiretroviral therapy (HAART). At model initialization, the proportion of diagnosed agents on HAART was set to approximate HIV care continuum surveillance estimates for each sub-class of agent. We interpolated the monthly probability of HAART initiation among diagnosed agents in each class such that the total proportion on treatment remained stable over the course of the simulation (after accounting for HAART discontinuation, see below).

In order to account for high rates of HIV treatment interruptions and discontinuation observed among some HIV-infected populations, we assumed that some agents on HART may discontinue therapy. The monthly probability of HAART discontinuation for agents in the community was estimated from previously published es- timates. Agents who discontinue therapy at time step j may re-initiate care at any time at the same rate as those who are newly diagnosed.

At model initialization, we set the proportion of HIV- diagnosed agents achieving >90 adherence to match those values reported by HIV care continuum surveil- lance activities. We assumed that agents who are >90 adherent to therapy have an undetectable viral load at a threshold of less than 200 copies/mL, with a small but non-zero probability of transmission. Agents that newly initiate HAART are assigned an adherence value A ranging from A1 - A5. The proba- bility of achieving >90 adherence (A5) for agents newly initiating HAART varies by class, and were estimated from HIV care continuum surveillance activities in the same manner as agents on HAART at model initializa- tion. Agents that do not achieve >90 adherence were as- signed to one of four other adherence quartiles (A1-A4) with equal probability. We assumed that adherence is constant while an agent is on therapy. We also note that, in this model, we do not account for type of HAART reg- imen or the development of virologic resistance; as such, the effect of adherence on virologic suppression and sub- sequent risk of transmission represent mean values ob- served in the treated population.

We model the relationship between HAART adherence and the suppression of viral replication implicitly, such that, for each adherence value (A), we assigned a differ- ent value for per-contact risk of HIV transmission. The higher values of HAART adherence reduce the per-event probability of HIV transmission. These values have been estimated from previously conducted studies investigat- ing the relationship between adherence and viral load,[?] as well as the effect of viral suppression on HIV trans- mission.

1.7.3 PReP

1.7.4 Functions

The following are the core functions associated with an agent class.

```
def get_ID(self):
   #Returns agent ID
   return self._ID

def partner(self, partner):
   #Partner self agent to partner agent
   assert partner in self._partners:
   raise KeyError("Partner %s is already bonded with agent %s"%(partner.get_ID(),␣
↪self._ID))
   self._partners.append(partner)

def unpartner(self, partner):
   #Removes partner from self agent partner list
   try:
       if self != agent:self._partners.remove(agent)
   except KeyError:
       raise KeyError("agent %s is not a member of agent set %s"%(agent.get_ID(),␣
↪self.get_ID()))
```

1.8 License

Author(s): Maximilian King (previous authors: Lars Seemann - lseemann@uh.edu) Email: Maximilian_King@brown.edu Organization: Marshall Lab, Department of Epidemiology - Brown University

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the <organization> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL <COPYRIGHT HOLDER> BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search