emodpy-hiv

Institute for Disease Modeling

Feb 25, 2022

CONTENTS

emodpy-hiv is a collection of Python scripts and utilities created to streamline user interactions with EMOD and idmtools for modeling generic diseases. Much of the functionality is inherited from the emod_api package and emodpy package.

Additional information about how to use idmtools can be found at in Welcome to idmtools. Additional information about EMOD HIV parameters can be found in EMOD parameter reference.

If you have questions, see Frequently asked questions for general questions about all emodpy functionality or *Frequently asked questions* for functionality specific to emodpy-hiv.

See Welcome to idmtools for a diagram showing how idmtools and each of the related packages are used in an end-to-end workflow using EMOD as the disease transmission model.

CHAPTER

ONE

EMODPY-HIV INSTALLATION

Follow the steps below to install emodpy-hiv.

1.1 Prerequisites

First, ensure the following prerequisites are met.

- Windows 10 Pro or Enterprise, Linux, or Mac
- Python 3.9 64-bit (https://www.python.org/downloads/release)
- A file that indicates the pip index-url:
  - Windows
  - Linux

  In C:\Users\Username\pip\pip.ini, containing the following:

```
[global]
index-url = https://packages.idmod.org/api/pypi/pypi-production/simple
```

  In $HOME/.config/pip/pip.conf, containing the following:

```
[global]
index-url = https://packages.idmod.org/api/pypi/pypi-production/simple
```

1.2 Installation instructions

1. Open a command prompt and create a virtual environment in any directory you choose. The command below names the environment "v-emodpy-hiv", but you may use any desired name:

```
python -m venv v-emodpy-hiv
```

2. Activate the virtual environment:
   - Windows
   - Linux

   Enter the following:

```
v-emodpy-hiv\Scripts\activate
```

Enter the following:

```
source v-emodpy-hiv/bin/activate
```

3. Install emodpy-hiv packages:

```
pip install emodpy-hiv
```

If you are on Linux, also run:

```
pip install keyrings.alt
```

4. When you are finished, deactivate the virtual environment by entering the following at a command prompt:

```
deactivate
```

CHAPTER

TWO

FREQUENTLY ASKED QUESTIONS

As you get started with emodpy-hiv, you may have questions. The most common questions are answered below. For questions related to functionality in related packages, see Frequently asked questions for idmtools, Frequently asked questions for emod-api, and Frequently asked questions for emodpy.

Contents

2.1 How do I set configuration parameters?

Define your own parameter-setting function such as `set_param_fn` and pass that function to the emodpy task creator as the `param_custom_cb` parameter. In that function, you can set the parameters directly. For example:

```
    config.parameters.Enable_Demographics_Reporting = 0  # just because I don't like␣
↪our default for this

    # config hacks until schema fixes arrive
    config.parameters.pop( "Serialized_Population_Filenames" )
    config.parameters.pop( "Serialization_Time_Steps" )
    config.parameters.Report_HIV_Event_Channels_List = []
```

(continues on next page)

(continued from previous page)

```
    config.parameters.Male_To_Female_Relative_Infectivity_Ages = [] # 15,25,35 ]
    config.parameters.Male_To_Female_Relative_Infectivity_Multipliers = [] # 5, 1, 0.
↪5 ]
    # This one is crazy! :(
    config.parameters.Maternal_Infection_Transmission_Probability = 0
    config.parameters['logLevel_default'] = "WARNING" # 'LogLevel_Default' is not in␣
↪scheme, so need to use the old style dict keys

    return config


def build_camp():
    """
    Build a campaign input file for the DTK using emod_api.
```

See examples/start_here/example.py. for additional information.

If you prefer something more modular, you can call a function in a standalone script/file that sets the configuration parameters.

2.2 Are there parameter defaults?

Great question. If you don't set any configuration parameters, they will have defaults based on the schema. There are not yet configuration parameter defaults specific to HIV.

The HIV team has some demographic parameter defaults set using `emodpy_hiv.demographics.DemographicsTemplates.AddDefaultSociety()`. They can be seen in demographics/DemographicsTemplates.py.

2.3 How do I create a minimal campaign that just seeds an outbreak?

You can use the following code:

```
def build_campaign():
    import emod_api.campaign as camp
    camp.set_schema( manifest.schema_path ) # don't worry about this for now
    import emodpy_hiv.interventions.outbreak as ob

    event = ob.new_intervention( timestep=365, camp=camp, coverage=0.01 )
    camp.add( event )
    return camp
```

The code above creates a new intervention from the outbreak submodule that causes 1% of the population to get an infection at timestep 365, and then adds that to the campaign. The only remaining thing to do is pass the 'build_campaign' function to the task creator function ('from_default2'). To see the documentation for the hiv outbreak module go here.

2.4 How do I give a therapeutic intervention, like ART, to people?

We're going to divide this into 4 steps:

1. Import the art module for creating the ART intervention.
2. Create the ART intervention the way you want it.
3. import and emod_api.interventions.common module for distribution interventions.
4. Invoke the ScheduledCampaignEvent function.

Let's look at the code that will go into your build_campaign function::

```
def build_campaign():
    import emod_api.campaign as camp
    camp.set_schema( manifest.schema_path ) # don't worry about this for now
    import emodpy_hiv.interventions.art as art
    import emod_api.interventions.common as com

    art_iv = art.new_intervention( camp )
    event1 = com.ScheduledCampaignEvent( camp, Start_Day=123, Intervention_List=[ art_
↪iv ] )
    camp.add( event )

    event2 = com.ScheduledCampaignEvent( camp, Start_Day=366, Node_Ids=[ 1 ], Number_
↪Repetitions = 10 Timesteps_Between_Repetitions = 14, Property_Restrictions =
↪"Risk=High",
                                         Demographic_Coverage = 0.04, Target_Age_
↪Min=20*365, Target_Age_Max=25*365, Target_Gender = "Male", Intervention_List=[ art_
↪iv ]) )
    camp.add( event )
```

The first 4 lines take care of our imports and initializating the campaign module with the schema. The next line creates the simplest possible intervention. Then we create a campaign event that distributes the ART intervention at timestep 123, and we add this to the campaign. Because we leave all the targeting parameters unspecified, the function uses the defaults, which basically means "everybody". In event2, we use all of the targeting and scheduling parameters to distributing ART every 2 weeks, 10 times in a row, starting at t=366, just in node 1, to 4% of the males between the ages of 20 and 25 in the "High Risk" group based on individual properties. Now in practice the repetitions don't make much sense because we're targeting the same people each rep as we got the first time, but it makes the point.

2.5 How do I give out an intervention to people based on a trigger instead of at a particular time?

The key part here is to use the TriggeredCampaignEvent function instead of ScheduledCampaignEvent. Let's look at the code::

```
def build_campaign():
    import emod_api.campaign as camp
    camp.set_schema( manifest.schema_path ) # don't worry about this for now
    import emodpy_hiv.interventions.art as art
    import emod_api.interventions.common as com

    art_iv = art.new_intervention( camp )
```

(continues on next page)

(continued from previous page)

```
    event1 = com.TriggeredCampaignEvent( camp, Start_Day=123, Event_Trigger=
↪"NewInfection", Intervention_List=[ art_iv ] )
    camp.add( event )
```

So we can see that the code is very similar, but we pass a new parameter to this new function, Event_Trigger. This can be any built-in event known to the model – usually related to health events – or an ad-hoc one you publish from another campaign event.

2.6 Are there are any helper functions to make this a little more concise?

Yes. There is a function to do most of the above for you. But you still create and pass the intervention itself.

2.7 What if I want to broadcast an event when I distribute the intervention?

You can use the BroadcastEvent function and use that as the intervention or just one of multiple interventions.

2.8 What if I want to have a delay between the trigger (signal) and when the intervention is actually distributed?

You may want to use this function.

2.9 What if I want to change someone's Individual Property?

That's actually just an intervention, PropertyValueChanger. See an example of this in action.

2.10 Now I want to distribute tests and distribute interventions to only those who test positive?

First, find the test intervention. This code should now seem unsurprising.:

```
def build_campaign():
    import emod_api.campaign as camp
    camp.set_schema( manifest.schema_path )
    import emodpy_hiv.interventions.art as art
    import emodpy_hiv.interventions.rapiddiag as diag
    import emod_api.interventions.common as com

    diagnostic = diag.new_intervention( camp )
    art_iv = art.new_intervention( camp )
    test_event = com.TriggeredCampaignEvent( camp, Start_Day=1, Event_Trigger=
↪"NewInfection", Intervention_List=[ diagnostic ] )
```

(continues on next page)

(continued from previous page)

```
    treat_event = com.TriggeredCampaignEvent( camp, Start_Day=1, Event_Trigger=
↪"TestedPositive", Intervention_List=[ art_iv ] )
    camp.add( test_event )
    camp.add( treat_event )
```

Testing everyone who is infected is obviously a bit naive but it just shows the idea.

2.11 I see lots of HIV_SIM examples. Are there any STI_SIM examples?

Not at this time.

2.12 I pip installed emodpy-hiv, but I want to make changes. How should I do that?

Install at a command prompt using the following:

```
python package_setup.py develop
```

This method is the most popular and proven, though there are some other options. Installing this way means that the emodpy-hiv module in site-packages actually points to the same code as you have checked out in git. For more detail, see this Stack Overflow post.

However, we aim to get the desired changes quickly tested and included in the versioned module we release via pip install.

2.13 What's the command to get all the latest (stable) Python packages?

:: pip install emodpy_hiv –upgrade –upgrade-strategy eager

CHAPTER

THREE

API REFERENCE

3.1 emodpy_hiv package

3.1.1 Subpackages

emodpy_hiv.demographics package

Submodules

emodpy_hiv.demographics.DemographicsTemplates module

emodpy_hiv.demographics.DemographicsTemplates.**add_society_from_template**(*demog*, *key='default'*)

emodpy_hiv.demographics.DemographicsTemplates.**add_default_society**(*demog*)

Adds a PFA config based on IDM defaults.

emodpy_hiv.demographics.HIVDemographics module

This module contains the classes and functions for creating demographics files for HIV simulations. For more information on EMOD demographics files, see Demographics file.

class emodpy_hiv.demographics.HIVDemographics.**HIVDemographics**(*nodes*, *idref='Gridded world grump2.5arcmin'*, *base_file=None*)

Bases: emod_api.demographics.Demographics.Demographics

This class is derived from emod_api.demographics.Demographics.Demographics and sets certain defaults for HIV in construction.

Parameters

- **nodes** – The number of nodes to create.
- **idref** – Method describing how the latitude and longitude values are created for each of the nodes in a simulation. "Gridded world" values use a grid overlaid across the globe at some arcsec resolution. You may also generate the grid using another tool or coordinate system. For more information, see Metadata.

- **base_file** – A basic demographics file used as a starting point for creating more complicated demographics files. For example, using a single node file to create a multi-node file for spatial simulations.
- **init_prev** – The initial HIV prevalence of the population.

Returns None

fertility(*path_to_csv*)
Set fertility based on data. Simulation shall concist of individual pregnancies with rates by woman's age and year-of-simulation using data from provided csv.

mortality(*file_male*, *file_female*, *interval_fit=None*, *which_point='mid'*, *predict_horizon=2050*, *csv_out=False*, *n=0*, *results_scale_factor=0.0027397260273972603*)
For back-compat. This function can go away.

apply_assortivity(*rel_type*, *matrix=None*)
Add an assortivity matrix to Pair-Forming Algo. Right now only applies to RISK IP.

Parameters

- **rel_type** – "COMMERCIAL", "INFORMAL", "MARITAL", or "TRANSITORY"
- **matrix** – 3x3 matrix of assortivity values, row represents male, column represents female.

Returns N/A.

set_concurrency_params_by_type_and_risk(*rel_type*, *ip_value*, *max_simul_rels_male=None*, *max_simul_rels_female=None*, *prob_xtra_rel_male=None*, *prob_xtra_rel_female=None*)
Set concurrent relationship formation params for a given relationship type and risk group.

Parameters

- **rel_type** – Relationship category: "COMMERCIAL", "MARITAL", "INFORMAL" or "TRANS"
- **ip_value** – Usually Risk Group but based on defined IP. "High", "Medium", or "Low"
- **max_simul_rels_male** – Sets Max_Simultaneous_Relationships_Male.
- **max_simul_rels_female** – Sets Max_Simultaneous_Relationships_Female.
- **prob_xtra_rel_male** – Sets Prob_Extra_Relationship_Male.
- **prob_xtra_rel_female** – Sets Prob_Extra_Relationship_Female.

Returns Nothing.

set_pair_form_params(*rel_type*, *new_constant_rate=None*)
Set Formation_Rate_Constant value for the specified relationship type.

Parameters

- **rel_type** – Relationship Type. E.g., "MARITAL"
- **new_constant_rate** – New value for Formation_Rate_Constant for the relationship type.

set_coital_act_rate(*rel_type*, *rate=None*)
Set Coital_Act_Rate value for the specified relationship type.

Parameters

- **rel_type** – Relationship Type. E.g., "MARITAL"
- **rate** – New value for Coital_Act_Rate for the relationship type.

set_condom_usage_probs(*rel_type*, *min=None*, *mid=None*, *max=None*, *rate=None*)
Set Condom_Usage_Probability values for the specified relationship type using 4 values to configure a sigmoid.

Parameters

- **rel_type** – Relationship Type. E.g., "MARITAL"
- **min** – "Min" (a probability)
- **mid** – "Mid" (a year)
- **max** – "Max" (a probability)
- **rate** – "Rate" (a probability)

set_relationship_duration(*rel_type*, *weibull_scale=None*, *weibull_heterogeneity=None*)
Set the Weibull configuration values for the draw that determines the duration of relationships of the specified type.

Parameters

- **rel_type** – Relationship Type. E.g., "MARITAL"
- **weibull_scale** – value of Duration_Weibull_Scale
- **weibull_heterogeneity** – value of Duration_Weibull_Heterogeneity

add_or_update_initial_risk_distribution(*distribution=None*)

emodpy_hiv.demographics.HIVDemographics.**from_template_node**(*lat=0*, *lon=0*, *pop=1000000.0*, *name=1*, *forced_id=1*)
Create a single-node *HIVDemographics* instance from the parameters you supply.

Parameters

- **lat** – Latitude of the centroid of the node to create.
- **lon** – Longitude of the centroid of the node to create.
- **pop** – Human population of the node.
- **name** – The name of the node. This may be a characteristic of the node, such as "rural" or "urban", or an identifying integer.
- **forced_id** – The node ID for the single node.

Returns A `HIVDemographics` instance.

emodpy_hiv.demographics.HIVDemographics.**from_pop_csv**(*pop_filename_in*, *pop_filename_out='spatial_gridded_pop_dir'*, *site='No_Site'*)
Create a multi-node *HIVDemographics* instance from a CSV file describing a population.

Parameters

- **pop_filename_in** – The path to the demographics file to ingest.
- **pop_filename_out** – The path to the file to output.
- **site** – A string to identify the country, village, or trial site.

Returns A `HIVDemographics` instance.

`emodpy_hiv.demographics.HIVDemographics.`**`from_params`**(*tot_pop=1000000.0*, *num_nodes=100*, *frac_rural=0.3*, *id_ref='from_params'*)

Create a multi-node *`HIVDemographics`* instance as a synthetic population based on a few parameters.

Parameters

- **`tot_pop`** – The total human population in the node.
- **`num_nodes`** – The number of nodes to create.
- **`frac_rural`** – The fraction of the population that is rural.
- **`id_ref`** – Method describing how the latitude and longitude values are created for each of the nodes in a simulation. "Gridded world" values use a grid overlaid across the globe at some arcsec resolution. You may also generate the grid using another tool or coordinate system. For more information, see Metadata.

Returns A *`HIVDemographics`* instance.

emodpy_hiv.interventions package

Submodules

emodpy_hiv.interventions.art module

`emodpy_hiv.interventions.art.`**`new_intervention`**(*camp*)

AntiRetroviralTherapy intervention wrapper. Just the intervention. No configuration yet.

`emodpy_hiv.interventions.art.`**`new_intervention_event`**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

ART intervention as complete (scheduled) event.

`emodpy_hiv.interventions.art.`**`new_intervention_as_file`**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.artdropout module

`emodpy_hiv.interventions.artdropout.`**`new_intervention`**(*camp*)

ARTDropout intervention wrapper. Just the intervention. No configuration yet.

`emodpy_hiv.interventions.artdropout.`**`new_intervention_event`**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

ARTDropout intervention as complete (scheduled) event.

`emodpy_hiv.interventions.artdropout.`**`new_intervention_as_file`**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.artstagingbycd4agnosticdiag module

emodpy_hiv.interventions.artstagingbycd4agnosticdiag.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*, *abp_tvmap*, *abt_tvmap*, *abw_tvmap*, *cua_tvmap*, *cbt_tvmap*, *cbw_tvmap*)

Wrapper function to create and return a HIVARTStagingCD4AgnosticDiagnostic intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.artstagingbycd4agnosticdiag.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *abp_tvmap*, *abt_tvmap*, *abw_tvmap*, *cua_tvmap*, *cbt_tvmap*, *cbw_tvmap*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.artstagingbycd4agnosticdiag.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.artstagingbycd4diag module

emodpy_hiv.interventions.artstagingbycd4diag.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*, *Threshold_TVMap*, *IP_TVMap*, *IAT_TVMap*)

Wrapper function to create and return a HIVARTStagingByCD4Diagnostic intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.artstagingbycd4diag.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *thresh_tvmap*, *pregnant_tvmap*, *tb_tvmap*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.artstagingbycd4diag.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.cascade_helpers module

emodpy_hiv.interventions.cascade_helpers.**reset**(*camp*)

Utility function to clear out the campaign object, mostly useful to test. This function is not auto-imported with the module like the rest of the functions in this submodule. Caller has to explicitly import emodpy_hiv.interventions.cascade_helpers. :param camp: emod_api.campaign object.

Returns None.

emodpy_hiv.interventions.cascade_helpers.**seed_infection**(*camp*, *timestep*, *coverage*)

Seed an infection by time and %age of population infected.

emodpy_hiv.interventions.cascade_helpers.**add_triggered_event**(*camp*, *in_trigger*, *out_iv*, *coverage=1.0*, *target_sex='All'*, *target_risk=''*, *event_name=''*)

An alias for triggered_event_common. Naming things well is hard.

emodpy_hiv.interventions.cascade_helpers.**triggered_event_common**(*camp*, *in_trigger*, *out_iv*, *coverage=1.0*, *target_sex='All'*, *target_risk=''*, *event_name=''*)

Parameterized utility function used by rest of functions in this submodule that listen for a trigger and distribute an intervention (or list thereof) as a result.

emodpy_hiv.interventions.cascade_helpers.**add_choice**(*camp*, *sympto_signal='HIVSymptomatic'*, *get_tested_signal='GetTested'*)

Listen for HIVSymptomatic trigger. Then toss a coin (Random Choice), heads get tested, tails maybe it's just a cold.

emodpy_hiv.interventions.cascade_helpers.**add_test**(*camp*, *get_tested_signal='GetTested'*)

Listen for GetTested signal. Then get HIV RapidDiagnostic test after a delay of 30 days.

emodpy_hiv.interventions.cascade_helpers.**trigger_art_from_pos_test**(*camp*, *input_signal='HIVPositiveTest'*, *output_signal='StartTreatment'*, *lag_time=30*)

Listen for HIVPositiveTest trigger. Then Trigger ART. Note that Trigger ART isn't same as starting it.

emodpy_hiv.interventions.cascade_helpers.**add_art_from_trigger**(*camp*, *signal='StartTreatment'*)

Actually distribute ART if a StartTreatment signal/trigger is observed. Broadcast a StartedART signal synchronously.

emodpy_hiv.interventions.cascade_helpers.**trigger_art**(*camp*, *timestep*, *coverage*, *trigger='StartTreatment'*)

Schedule a broadcast of StartTreatment (or equivalent), not based on any observed signals.

emodpy_hiv.interventions.delay module

emodpy_hiv.interventions.delay.**new_delay**(*camp*, *Bcast_Event*, *Expire_Event=''*, *Coverage=1*, *Delay=1*, *Shelf_Life=36500*, *Name=''*)

Wrapper function to create and return a HIVDelayedIntervention intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.delay.**new_intervention_event**(*camp*, *bcast_event*, *expire_event=''*, *coverage=1*, *delay=1*, *shelf_life=0*)

Delay as scheduled event.

emodpy_hiv.interventions.delay.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.drawblood module

emodpy_hiv.interventions.drawblood.**new_diagnostic**(*camp*, *Positive_Event*, *Base_Sensitivity=1*, *Base_Specificity=1*, *Days_To_Diagnosis=0*, *Treatment_Fraction=1*)

Wrapper function to create and return a HIVDrawBlood intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.drawblood.**new_intervention_event**(*camp*, *pos_event*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.drawblood.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.malecirc module

emodpy_hiv.interventions.malecirc.**new_intervention**(*camp*)
MaleCircumcision intervention wrapper. Just the intervention. No configuration yet.

emodpy_hiv.interventions.malecirc.**new_intervention_event**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
VMMC intervention as complete (scheduled) event.

emodpy_hiv.interventions.malecirc.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.modcoinf module

emodpy_hiv.interventions.modcoinf.**new_intervention**(*camp*)
ModifySTICoInfection intervention wrapper. Just the intervention. No configuration yet.

emodpy_hiv.interventions.modcoinf.**new_intervention_event**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
ModifySTICoInfection intervention as complete (scheduled) event.

emodpy_hiv.interventions.modcoinf.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.outbreak module

emodpy_hiv.interventions.outbreak.**new_intervention**(*timestep*, *camp*, *coverage=0.01*)
Seed HIV infection at a certain timestep, with a certain prevalence.

Parameters

- **timestep** (*float*) – When? Timestep at which outbreak should occur.
- **camp** – emod_api.campaign object that has schema_path.
- **coverage** – How Much? The intended level of initial prevalance.

Returns event as dict

Return type event

emodpy_hiv.interventions.outbreak.**seed_infections**(*camp*, *start_day=365*, *coverage=0.075*, *target_properties=None*, *target_min_age=0*, *target_max_age=125*, *target_gender='All'*)
Create outbreak event with more targeting than 'new_intervention'.

emodpy_hiv.interventions.pmtct module

emodpy_hiv.interventions.pmtct.**new_intervention**(*camp*, *efficacy=1.0*)
PMTCT intervention wrapper. Just the intervention. No configuration yet.

emodpy_hiv.interventions.pmtct.**new_intervention_event**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
PMTCT intervention as complete (scheduled) event.

emodpy_hiv.interventions.pmtct.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.prep module

emodpy_hiv.interventions.prep.**new_intervention**(*camp*)
PrEP intervention wrapper. Just the intervention. No configuration yet.

emodpy_hiv.interventions.prep.**new_intervention_event**(*camp*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
PrEP intervention as complete (scheduled) event.

emodpy_hiv.interventions.prep.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.random module

emodpy_hiv.interventions.random.**new_diagnostic**(*camp*, *choices*)
Wrapper function to create and return a HIVRandomChoice intervention.

Parameters

- **camp** – emod_api.campaign object with schema_path set.
- **choices** – dict of events:probability, with probs summing up to 1.0

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.random.**new_intervention_event**(*camp*, *choices*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
Diagnostic as scheduled event.

emodpy_hiv.interventions.random.**new_intervention_as_file**(*camp*, *start_day*, *choices*, *filename=None*)

emodpy_hiv.interventions.rapiddiag module

emodpy_hiv.interventions.rapiddiag.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*)
Wrapper function to create and return a HIVRapidHIVDiagnostic intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.rapiddiag.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.rapiddiag.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.reftracker module

emodpy_hiv.interventions.reftracker.**DistributeIVByRefTrack**(*camp*, *Start_Day*, *Intervention*, *TVMap*, *node_ids=None*, *Event_Name='Scheduled_Campaign_Event'*, *Property_Restrictions=None*, *Target_Age_Min=0*, *Target_Age_Max=45625*, *Target_Gender='All'*, *Update_Period=None*, *IV_Tracking_Name=None*)

Wrapper function to create and return a ScheduledCampaignEvent intervention. The alternative to a ScheduledCampaignEvent is a TriggeredCampaignEvent.

Parameters

- **camp** – emod_api.campaign object with schema_path set.
- **Start_Day** – When to start.
- **Intervention** – Valid intervention to be distributed together as necessary to track coverage targets. Can be single intervention or list (list is useful where you want a co-event). If list, actual intervention should be first.
- **Event_Name** – Name for overall campaign event, of not functional meaning.
- **node_ids** – Nodes to target with this intervention, return from utils.do_nodes().
- **Property_Restrictions** – Individual Properties a person must have to receive the intervention(s).
- **Number_Repetitions** – N/A
- **Timesteps_Between_Repetitions** – N/A
- **Target_Demographic** – Everyone, ExplicitAgeRanges, etc.
- **Target_Age_Min** – Minimum age (in years).
- **Target_Age_Max** – Maximum age (in years).
- **Target_Gender** – All, Male, or Female.
- **Update_Period** – Number representing how frequently the distributions are done.
- **IV_Tracking_Name** – Optional string parameter to distinguish one intervention from another if you're doing multiple campaigns with the same underlying intervention.

Returns Schema-based smart dictionary representing a new ScheduledCampaignEvent intervention ready to be added to a campaign.

Return type ReadOnlyDict

emodpy_hiv.interventions.reftracker.**new_intervention_as_file**(*camp*, *actual_intervention*, *start_day=1*, *filename=None*)

emodpy_hiv.interventions.sigmoiddiag module

emodpy_hiv.interventions.sigmoiddiag.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*, *ramp_min*, *ramp_max*, *ramp_midyear*, *ramp_rate*)

Wrapper function to create and return a HIVSigmoidByYearAndSexDiagnostic intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.sigmoiddiag.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *ramp_min=0*, *ramp_max=1*, *ramp_midyear=2000*, *ramp_rate=1*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.sigmoiddiag.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.stipostdebut module

emodpy_hiv.interventions.stipostdebut.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*, *Base_Sensitivity=1*, *Base_Specificity=1*, *Days_To_Diagnosis=0*, *Treatment_Fraction=1*)

Wrapper function to create and return a STIIsPostDebut intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.stipostdebut.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *start_day=1*, *coverage=1.0*, *node_ids=None*)

Diagnostic as scheduled event.

emodpy_hiv.interventions.stipostdebut.**new_intervention_as_file**(*camp*, *start_day*, *filename=None*)

emodpy_hiv.interventions.utils module

emodpy_hiv.interventions.utils.**set_tvmap_lists_from_map**(*tvmap*, *param*)

emodpy_hiv.interventions.utils.**declutter**(*event*)
These are mostly temporary hacks that clean up the output json; should go away with subsequent cherry-picks of schema enhancements from other branches.

emodpy_hiv.interventions.utils.**broadcast_event_immediate**(*camp*, *event_trigger: str = 'Births'*)

emodpy_hiv.interventions.utils.**broadcast_event_delayed**(*camp*, *event_trigger*, *delay=None*)

emodpy_hiv.interventions.yearandsexdiag module

emodpy_hiv.interventions.yearandsexdiag.**new_diagnostic**(*camp*, *Positive_Event*, *Negative_Event*, *TVMap*)
Wrapper function to create and return a HIVPiecewiseByYearAndSexDiagnostic intervention.

Parameters **camp** – emod_api.campaign object with schema_path set.

Returns Schema-based smart dictionary representing a new

Return type ReadOnlyDict

emodpy_hiv.interventions.yearandsexdiag.**new_intervention_event**(*camp*, *pos_event*, *neg_event*, *tvmap*, *start_day=1*, *coverage=1.0*, *node_ids=None*)
Diagnostic as scheduled event.

emodpy_hiv.interventions.yearandsexdiag.**new_intervention_as_file**(*camp*, *start_day*, *tvmap*, *filename=None*)

3.1.2 Submodules

emodpy_hiv.download module

emodpy_hiv.download.**download**(*experiment_id*, *local_output_path=''*, *files_to_get=None*)
Download HIV output file(s) to local disk. Just gets ReportHIVByAgeAndGender.csv. Intended to be used command line: python -m emopyhiv.download <COMPS Experiment ID> <Optional Output Path> <Files To Get> Note that target files need to include "output/" if that's where they are.

emodpy_hiv.utils module

emodpy_hiv.utils.**non_schema_checks**(*config*)
Do additional voluntary checks for config consistency. There's no real fixed list for what should be here.

PYTHON MODULE INDEX

e

INDEX

F

H

M

N

R

S

T