
emodpy-malaria

Institute for Disease Modeling

Apr 29, 2021

CONTENTS

- 1 Installation prerequisites 3**
 - 1.1 emodpy-malaria installation 3
- 2 emodpy_malaria 5**
 - 2.1 emodpy_malaria package 5
 - 2.1.1 Subpackages 5
 - 2.1.2 Submodules 33
- Python Module Index 35**
- Index 37**

emodpy-malaria is a collection of Python scripts and utilities created to streamline user interactions with EMOD and idmtools for modeling malaria. Much of the functionality is inherited from the [emod_api](#) and [emodpy](#) packages.

Additional information about how to use idmtools can be found at in [Welcome to idmtools](#). Additional information about EMOD malaria parameters can be found in [EMOD parameter reference](#).

INSTALLATION PREREQUISITES

First, ensure the following prerequisites are met before following the install instructions in *emodpy-malaria installation*.

- Windows 10 Pro or Enterprise, or Linux
- Python 3.6 or 3.7 64-bit (<https://www.python.org/downloads/release>)
- Git client, such as Git Bash or the Git GUI
- pip.ini (Windows) or pip.conf (Linux), containing the following:

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

1.1 emodpy-malaria installation

Follow the steps below to install emodpy-malaria.

Note: Currently, VPN connection is required to run the example.

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

```
python -m venv v-emodpy-malaria
```

2. Activate the virtual environment:

- Windows
- Linux

Enter the following:

```
v-emodpy-malaria\Scripts\activate
```

Enter the following:

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

3. Install emodpy-malaria packages:

```
pip install emodpy_malaria
```

If you are on Python 3.6, also run:

```
pip install dataclasses
```

If you are on Linux, also run:

```
pip install keyrings.alt
```

4. Open a command prompt and clone the emodpy-malaria GitHub repository to a local directory using the following command:

```
git clone https://github.com/InstituteForDiseaseModeling/emodpy-malaria.git
```

5. Verify installation by running the included Python example, `example.py`, located in `/examples/start_here`:

```
python example.py
```

Upon completion you can view the results in COMPS.

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

```
deactivate
```

EMODPY_MALARIA

2.1 emodpy_malaria package

2.1.1 Subpackages

`emodpy_malaria.demographics` package

Submodules

`emodpy_malaria.demographics.MalariaDemographics` module

This module contains the classes and functions for creating demographics files for malaria simulations. For more information on EMOD demographics files, see [Demographics file](#).

```
class emodpy_malaria.demographics.MalariaDemographics(nodes,  
                                                    idref='Gridded  
                                                    world  
                                                    grump2.5arcmin',  
                                                    base_file=None,  
                                                    init_prev=0.2)
```

Bases: `emod_api.demographics.Demographics.Demographics`

This class is derived from `emod_api.demographics.Demographics.Demographics` and sets certain defaults for malaria 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 malaria prevalence of the population.

Returns None

```
emodpy_malaria.demographics.MalariaDemographics.from_template_node(lat=0,  
                                                                    lon=0,  
                                                                    pop=1000000.0,  
                                                                    name=1,  
                                                                    forced_id=1,  
                                                                    init_prev=0.2)
```

Create a single-node *MalariaDemographics* 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.
- **init_prev** – The initial malaria prevalence of the node.

Returns A *MalariaDemographics* instance.

```
emodpy_malaria.demographics.MalariaDemographics.from_pop_csv(pop_filename_in,  
                                                                pop_filename_out='spatial_gridded_pop_di  
                                                                site='No_Site')
```

Create a multi-node *MalariaDemographics* 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 *MalariaDemographics* instance.

```
emodpy_malaria.demographics.MalariaDemographics.from_params(tot_pop=1000000.0,  
                                                                num_nodes=100,  
                                                                frac_rural=0.3,  
                                                                id_ref='from_params')
```

Create a multi-node *MalariaDemographics* 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 *MalariaDemographics* instance.

emodpy_malaria.interventions package

Submodules

emodpy_malaria.interventions.bednet module

This module contains functionality for bednet distribution.

```
emodpy_malaria.interventions.bednet.Bednet(camp, start_day, coverage=1.0, blocking_eff=1, killing_eff=1, repelling_eff=1, usage_eff=1, blocking_decay_rate=0, blocking_predecay_duration=365, killing_decay_rate=0, killing_predecay_duration=365, repelling_decay_rate=0, repelling_predecay_duration=365, usage_decay_rate=0, usage_predecay_duration=365, node_ids=None, insecticide=None)
```

Add a simple insecticide-treated bednet intervention with configurable efficacy and usage that can decay over time to your campaign. This is equivalent to [UsageDependentBednet](#) with exponential waning.

Parameters

- **camp** – The `emod_api.campaign` object to which the intervention will be added.
- **start_day** – The day of the simulation on which the bednets are distributed. We recommend aligning this with the start of the simulation.
- **coverage** – The proportion of the population that will receive a bednet.
- **blocking_eff** – The efficacy of the bednet at blocking mosquitoes from feeding.
- **killing_eff** – The efficacy of the bednet at killing mosquitoes that land on it.
- **repelling_eff** – The efficacy of the bednet at repelling mosquitoes.
- **usage_eff** – The proportion of time that individuals with a bednet use it. This effectively reduces the other efficacy measures by the amount specified; a value of 0.5 will reduce blocking, killing, and repelling efficacies by half.
- **blocking_decay_rate** – The exponential decay length, in days, where the current effect is equal initial efficacy - dt/decay rate.
- **blocking_predecay_duration** – The time, in days, before bednet efficacy begins to decay.
- **killing_decay_rate** – The exponential decay length, in days, where the current effect is equal initial efficacy - dt/decay rate.
- **killing_predecay_duration** – The time, in days, before bednet efficacy begins to decay.
- **repelling_decay_rate** – The exponential decay length, in days, where the current effect is equal initial efficacy - dt/decay rate.
- **repelling_predecay_duration** – The time, in days, before bednet efficacy begins to decay.
- **usage_decay_rate** – The exponential decay length, in days, where the current usage is equal initial efficacy - dt/decay rate.

- **usage_predecay_duration** – The time, in days, before bednet usage begins to decay.
- **node_ids** – The IDs of the nodes in which to distribute the bednets.
- **insecticide** – The name of the insecticide used in the bednet.

Returns The bednet intervention event.

```
emodpy_malaria.interventions.bednet.BasicBednet(camp, start_day, coverage=1.0,  
                                                  blocking_eff=1, killing_eff=1, re-  
                                                  pelling_eff=1, usage_eff=1, insecti-  
                                                  cide=None)
```

Add a simpler insecticide-treated bednet intervention with constant efficacy and usage to your campaign. This is equivalent to [SimpleBednet](#).

Parameters

- **camp** – The `emod_api.campaign` object to which the intervention will be added.
- **start_day** – The day of the simulation on which the bednets are distributed. We recommend aligning this with the start of the simulation.
- **coverage** – The proportion of the population that will receive a bednet.
- **blocking_eff** – The efficacy of the bednet at blocking mosquitoes from feeding.
- **killing_eff** – The efficacy of the bednet at killing mosquitoes that land on it.
- **repelling_eff** – The efficacy of the bednet at repelling mosquitoes.
- **usage_eff** – The proportion of individuals with a bednet who use it.
- **insecticide** – The insecticide used in the bednet.

Returns The bednet intervention event.

```
emodpy_malaria.interventions.bednet.new_intervention_as_file(camp, start_day,  
                                                             filename=None)
```

Write a campaign file to disk with a single bednet event, using defaults. Useful for testing and learning.

Parameters

- **camp** – The `emod_api.campaign` object to which the intervention will be added.
- **start_day** – The day of the simulation on which the bednets are distributed. We recommend aligning this with the start of the simulation.
- **filename** – The campaign filename; can be omitted and default will be used and returned to user.

Returns The campaign filename written to disk.

emodpy_malaria.interventions.common module

This module contains functionality common to many interventions.

```
emodpy_malaria.interventions.common.MalariaDiagnostic(camp, Measurement_Sensitivity, Detec-  
                                                         tion_Threshold, Diagnos-  
                                                         tic_Type)
```

Add a malaria diagnostic intervention to your campaign. This is equivalent to [MalariaDiagnostic](#).

Parameters

- **camp** – The `emod_api.campaign` object to which the intervention will be added.

- **Measurement_Sensitivity** – The number of microliters of blood tested to find single parasite/gameotocyte in a traditional smear (corresponds to inverse parasites/microliters sensitivity). This is similar to **Parasite_Smear_Sensitivity** and **Gameotocyte_Smear_Sensitivity** in the config used for reports, but this is for this instance of the diagnostic. In the following equation, if **measurement** is larger than **Detection_Threshold** a positive diagnosis is made:

```
measurement = float(1.0/Measurement_Sensitivity*GetRng()->
                    Poisson(Measurement_Sensitivity*true_parasite_
                    ↪density))
```

Used only when **Diagnostic_Type** is set to BLOOD_SMEAR_PARASITES or BLOOD_SMEAR_GAMETOCYTES.

- **Detection_Threshold** – The diagnostic detection threshold for parasites, in units of microliters of blood.
- **Diagnostic_Type** – The type of malaria diagnostic used. Possible values are:
 - BLOOD_SMEAR_PARASITES
 - BLOOD_SMEAR_GAMETOCYTES
 - PCR_PARASITES
 - PCR_GAMETOCYTES
 - PF_HRP2
 - TRUE_PARASITE_DENSITY
 - FEVER

Returns The diagnostic intervention event.

`emodpy_malaria.interventions.common.AntiMalarialDrug(camp, Drug_Type, ctc=1.0)`

Add an antimalarial drug intervention to your campaign. This is equivalent to [AntimalarialDrug](#).

Parameters

- **camp** – The `emod_api.campaign` object to which the intervention will be added.
- **Drug_Type** – The name of the drug to distribute in a drugs intervention. Possible values are contained in **Malaria_Drug_Params**.
- **ctc** – The cost to consumer.

Returns The antimalarial drug intervention event.

emodpy_malaria.interventions.diag_survey module

This module contains functionality for diagnostic survey interventions.

```
emodpy_malaria.interventions.diag_survey.add_diagnostic_survey(camp, coverage:
                                                                    float = 1, repe-
                                                                    titions: int = 1,
                                                                    tsteps_btwn_repetitions:
                                                                    int = 365, tar-
                                                                    get: object
                                                                    = 'Everyone',
                                                                    start_day: int
                                                                    = 1, diagnos-
                                                                    tic_type: str =
                                                                    'BLOOD_SMEAR_PARASITES',
                                                                    diagnos-
                                                                    tic_threshold:
                                                                    float = 40,
                                                                    measure-
                                                                    ment_sensitivity:
                                                                    float = 0.1,
                                                                    event_name:
                                                                    str = 'Diag-
                                                                    nostic Survey',
                                                                    node_ids: list
                                                                    = None, posi-
                                                                    tive_diagnosis_configs:
                                                                    list =
                                                                    None, nega-
                                                                    tive_diagnosis_configs:
                                                                    list = None, re-
                                                                    ceived_test_event:
                                                                    str = 'Re-
                                                                    ceived_Test',
                                                                    IP_restrictions:
                                                                    list = None,
                                                                    NP_restrictions:
                                                                    list = None,
                                                                    disqualify-
                                                                    ing_properties:
                                                                    list = None, trig-
                                                                    ger_condition_list:
                                                                    list = None,
                                                                    listen-
                                                                    ing_duration:
                                                                    int = - 1, trig-
                                                                    gered_campaign_delay:
                                                                    int = 0,
                                                                    check_eligibility_at_trigger:
                                                                    bool = False, ex-
                                                                    pire_recent_drugs:
                                                                    any = None)
```

Add an intervention to create either a scheduled or a triggered event to the campaign using the *MalariaDiagnostic* class, an individual-level class, to test individuals. Upon positive or negative diagnosis, the list of events to occur (as defined in **positive_diagnosis_configs** or **negative_diagnosis_configs**) is then executed. These events can trigger other listening interventions.

Parameters

- **camp** – The `emod_api.campaign` object for building, modifying, and writing campaign configuration files.
- **coverage** – The probability an individual receives the diagnostic.
- **repetitions** – Number of repetitions of the survey intervention.
- **tsteps_btwn_repetitions** – Timesteps between repetitions.
- **target** – A dictionary targeting an age range and gender of individuals for treatment. In the format {"agemin": x, "agemax": y, "gender": z}. Default is Everyone.
- **start_day** – The day of the simulation on which the intervention is created. If triggered, runs on trigger, not on **start_day**.
- **diagnostic_type** – Type of malaria diagnostic used. Default is **BLOOD_SMEAR_PARASITES**. Available options are:
 - BLOOD_SMEAR_PARASITES
 - BLOOD_SMEAR_GAMETOCYTES
 - PCR_PARASITES
 - PCR_GAMETOCYTES
 - PF_HRP2
 - TRUE_INFECTION_STATUS
 - TRUE_PARASITE_DENSITY
 - FEVER
- **diagnostic_threshold** – The diagnostic detection threshold based on **diagnostic_type**:
 - TRUE_INFECTION_STATUS**
 - BLOOD_SMEAR_PARASITES** In parasites/microliter, use $\text{measurement} = \text{float}(1.0 / \text{measurementSensitivity} * \text{Poisson}(\text{measurementSensitivity} * \text{true_parasite_density}))$.
 - BLOOD_SMEAR_GAMETOCYTES** In gametocytes/microliter, use $\text{measurement} = \text{float}(1.0 / \text{measurementSensitivity} * \text{Poisson}(\text{measurementSensitivity} * \text{true_gametocyte_density}))$.
 - PCR_PARASITES and PCR_GAMETOCYTES** Use the true values and an algorithm based on the paper [Improving statistical inference on pathogen densities estimated by quantitative molecular methods : malaria gametocytaemia as a case study](#).
 - PF_HRP2** Add a new method to get the PfHRP2 value and check against the threshold.
 - TRUE_PARASITE_DENSITY** Check the true parasite density against the threshold.
 - FEVER** Check the person's fever against the threshold.
- **measurement_sensitivity** – Setting for **Measurement_Sensitivity** in *MalariaDiagnostic*.
- **event_name** – Description of the event.
- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of NodeSetAll.
- **positive_diagnosis_configs** – List of events to happen to an individual who receives a positive result from test.

- **negative_diagnosis_configs** – List of events to happen to individual who receives a negative result from test.
- **received_test_event** – String for individuals to broadcast upon receiving diagnostic.
- **IP_restrictions** – List of IndividualProperty key:value pairs that individuals must have to receive the diagnostic intervention. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]. Default is no restrictions.
- **NP_restrictions** – List of NodeProperty key:value pairs that nodes must have to receive the diagnostic intervention. For example, [{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]. Default is no restrictions.
- **disqualifying_properties** – List of IndividualProperty key:value pairs that cause an intervention to be aborted. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].
- **trigger_condition_list** – List of events that will trigger a diagnostic survey.
- **listening_duration** – Duration after start day to stop listening for events, as specified in **trigger_condition_list**. Default is -1, non-stop listening.
- **triggered_campaign_delay** – Delay of running the intervention after receiving a trigger from the **trigger_condition_list**.
- **check_eligibility_at_trigger** – If triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay.
- **expire_recent_drugs** – Adds [{"DrugStatus": None}] to **Property_Restrictions_Within_Node** for positive test config, so only those with that property receive drugs.

Returns None

emodpy_malaria.interventions.drug module

```
emodpy_malaria.interventions.drug.AntiMalarialDrug(camp, start_day=1, coverage=1.0,  
                                                    drug_name='Chloroquine',  
                                                    node_ids=None)
```

AntiMalarialDrug intervention wrapper.

```
emodpy_malaria.interventions.drug.new_intervention_as_file(camp, start_day, file-  
                                                            name=None)
```

emodpy_malaria.interventions.drug_campaign module

```
emodpy_malaria.interventions.drug_campaign.drug_configs_from_code(camp,  
                                                                    drug_code:  
                                                                    str = None)
```

Add a drug config to the simulation configuration based on its code and add the corresponding AntimalarialDrug intervention to the return dictionary. The drug_code needs to be one identified in the drug_cfg dictionary. For example passing the MDA_ALP drug code, will add the drugs config for Artemether, Lumefantrine, Primaquine to the configuration file and will return a dictionary containing a Full Treatment course for those 3 drugs. :param

drug_code: Code of the drug to add :param drug_ineligibility_duration: used as a flag, if anything is present, we add ["DrugStatus:RecentDrug"] :param as Disqualifying_Properties:

Returns A dictionary containing the parameters for an intervention using the given drug

```
emodpy_malaria.interventions.drug_campaign.add_drug_campaign(camp,      cam-
    paign_type:
    str      = 'MDA',
    drug_code: str =
    None, start_days:
    list = None, cov-
    erage: float = 1.0,
    repetitions: int = 1,
    tsteps_btwn_repetitions:
    int = 60, diag-
    nostic_type: str =
    'BLOOD_SMEAR_PARASITES',
    diagnos-
    tic_threshold: float
    = 40, measure-
    ment_sensitivity:
    float = 0.1,
    fmda_radius:
    int = 0,
    node_selection_type:
    str = 'DIS-
    TANCE_ONLY',
    trigger_coverage:
    float = 1.0, snow-
    balls: int = 0,
    treatment_delay:
    int = 0, trig-
    gered_campaign_delay:
    int = 0, node_ids:
    list = None,
    target_group:
    any = 'Everyone',
    drug_ineligibility_duration:
    int = 0,
    node_property_restrictions:
    list = None,
    ind_property_restrictions:
    list = None,
    disqualify-
    ing_properties:
    list = None, trig-
    ger_condition_list:
    list = None, lis-
    tening_duration:
    int = - 1, adher-
    ent_drug_configs:
    list = None, tar-
    get_residents_only:
    int = 1,
    check_eligibility_at_trigger:
    bool =
    False, receiv-
    ing_drugs_event_name='Received_Campaig
```

Add a drug intervention, as specified in **campaign_type**, to the campaign. This intervention uses the **Malaria-**

aDiagnostic class to create either a scheduled or a triggered event to the campaign and the **AntimalarialDrug** class to configure drug parameters. You can also specify a delay period for a triggered event that broadcasts afterwards.

Note: Dosing/Dosing_Type was removed, please use `adeherent_drug_configs` if you want something besides a full treatment. Also, `drug_ineligibility_duration` does not prevent diagnostics from being distributed, but it does prevent positively-tested patients from receiving the drugs.

Parameters

- **cb** – The `DTKConfigBuilder` object for building, modifying, and writing campaign configuration files.
- **campaign_type** – Type of drug campaign. Default is **MDA**. Available options are: * `MDA` * `MSAT` * `SMC` * `fMDA` * `MTAT` * `rfMSAT` * `rfMDA`
- **drug_code** – The drug code of the drug regimen (AL, DP, etc; allowable types are defined in `malaria_drugs.py`).
- **start_days** – List of start days (integers) when the drug regimen will be distributed. Due to diagnostic/treatment configuration, the earliest start day is 1. When `trigger_condition_list` is used then the first entry of `start_days` is the day to start listening for the trigger(s).
- **coverage** – Demographic coverage of the distribution (fraction of people at home during campaign).
- **repetitions** – Number of repetitions.
- **tsteps_btwn_repetitions** – Timesteps between the repetitions.
- **diagnostic_type** – Diagnostic config for diagnostic-dependent campaigns: * `MSAT` * `fMDA` * `rfMSAT`
- **diagnostic_threshold** – Diagnostic config for diagnostic-dependent campaigns: * `MSAT` * `fMDA` * `rfMSAT`
- **measurement_sensitivity** – setting for **Measurement_Sensitivity** in `MalariaDiagnostic`
- **fmda_radius** – Radius (in km) of focal response upon finding infection. Default is 0. Used with household only.
- **node_selection_type** – Node selection type for broadcasting focal response trigger. Available options are: * `DISTANCE_ONLY`: It will send the event to nodes that are within a given distance. * `MIGRATION_NODES_ONLY`: It will only send the event to nodes that the individual can migrate to. * `DISTANCE_AND_MIGRATION`: It will only send the event to migratable nodes that are within a given distance. Migratable nodes are Local and Regional.
- **trigger_coverage** – Used with RCD (Reactive Case Detection). Fraction of trigger events that will trigger an RCD. Coverage param sets the fraction of individuals reached during RCD response.
- **snowballs** – Number of snowball levels in reactive response.
- **treatment_delay** – For `MSAT` and `fMDA`, the length of time between administering diagnostic and giving drugs; for RCD, the length of time between treating index case and triggering RCD response.

- **triggered_campaign_delay** – When using `trigger_condition_list`, this indicates the delay period between receiving the trigger event and running the triggered campaign intervention.
- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of `NodeSetAll`.
- **target_group** – A dictionary of `{'agemin': x, 'agemax': y}` to target MDA, SMC, MSAT, fMDA to individuals between x and y years of age. Default is Everyone.
- **drug_ineligibility_duration** – When set to `> 0`, use `IndividualProperties` to prevent people from receiving drugs too frequently. Demographics file will need to define the IP `DrugStatus` with possible values `None` and `RecentDrug`. Individuals with status `RecentDrug` will not receive drugs during drug campaigns, though they are still eligible for receiving diagnostics (in MSAT, etc). Individuals who receive drugs during campaigns will have their `DrugStatus` changed to `RecentDrug` for `drug_ineligibility_duration` days.
- **node_property_restrictions** – List of `NodeProperty` key:value pairs that nodes must have to receive the diagnostic intervention. For example, `[{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]`. Default is no restrictions.
- **ind_property_restrictions** – List of `IndividualProperty` key:value pairs that individuals must have to receive the diagnostic intervention. For example, `[{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]`. Default is no restrictions.
- **disqualifying_properties** – List of `IndividualProperty` key:value pairs that cause an intervention to be aborted. For example, `[{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]`.
- **trigger_condition_list** – List of events that will begin a triggerable drug campaign, such as MDA, MSAT, SMC, fMDA, and MTAT.
- **listening_duration** – Duration to listen for the trigger. Default is -1, which listens indefinitely.
- **adherent_drug_configs** – List of adherent drug configurations, which are dictionaries from `configure_adherent_drug`.
- **target_residents_only** – When set to `true (1)`, the intervention is only distributed to individuals for whom the node is their home node. They are not visitors from another node.
- **check_eligibility_at_trigger** – If triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay.
- **receiving_drugs_event_name** – Event to send out when person received drugs. Default: `'Received_Campaign_Drugs'`

Returns Dictionary with drug campaign parameters

```

emodpy_malaria.interventions.drug_campaign.add_MDA(camp, start_days: list =
None, coverage: float =
1.0, drug_configs: list =
None, receiving_drugs_event:
emod_api.interventions.common.BroadcastEvent
= None, repetitions: int =
1, tsteps_btwn_repetitions:
int = 60, node_ids: list =
None, expire_recent_drugs:
emod_api.interventions.common.PropertyValueChanger
=
None, node_property_restrictions: list =
None, ind_property_restrictions:
list = None, disqualify-
ing_properties: list = None,
target_group: any = 'Everyone',
trigger_condition_list: list =
None, listening_duration: int =
- 1, triggered_campaign_delay:
int = 0, target_residents_only: int
= 1, check_eligibility_at_trigger:
bool = False)

```

Add MDA (mass drug administration) drug intervention to campaign. If there are multiple start days in a list and `trigger_condition_list` is empty, then mda's are created to run on the days in the `start_days` list. If the `triggerer_condition_list` is present, then a triggered mda is created and uses the first day of the `start_days`. If there are repetitions or a `triggered_campaign_delay` then separate nodeleveltriggered interventions are created with a delay that sends out an event that triggers the mda. Multiple start days are only valid for non-triggered mdas.

Parameters

- **camp** – The DTKConfigBuilder
- **for building, modifying, and writing campaign configuration files.** (*object*) –
- **start_days** – List of integers.
- **coverage** – Demographic coverage of mda's.
- **drug_configs** – List of dictionaries of drug configurations to be given out, created in `add_drug_campaign`.
- **receiving_drugs_event** – (Optional) Broadcast event container with event to be broadcast when drugs received.
- **repetitions** – Number of repetitions for mda. For triggered mda, this is for a repeated mda after a trigger.
- **tsteps_btwn_repetitions** – Timesteps between repeated scheduled mdas or between once-triggered repeated mdas.
- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of `NodeSetAll`.
- **expire_recent_drugs** – `PropertyValueChanger` intervention that updates `DrugStatus:Recent drug` to individual properties.

- **node_property_restrictions** – List of NodeProperty key:value pairs that nodes must have to receive the diagnostic intervention. For example, [{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]. Default is no restrictions.
- **disqualifying_properties** – List of IndividualProperty key:value pairs that cause an intervention to be aborted. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].
- **target_group** – A dictionary targeting an age range and gender of individuals for treatment. In the format {"agemin": x, "agemax": y, "gender": z}. Default is Everyone.
- **trigger_condition_list** – List of event triggers upon which mda(s) are distributed.
- **listening_duration** – Duration to listen for the trigger. Default is -1, which listens indefinitely.
- **triggered_campaign_delay** – Delay period between the trigger and the mda. Default is 0.
- **target_residents_only** – When set to true (1), the intervention is only distributed to individuals for whom the node is their home node. They are not visitors from another node.
- **check_eligibility_at_trigger** – If triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay.

Returns None

```
emodpy_malaria.interventions.drug_campaign.add_MSAT(camp, start_days: list =  
None, coverage: float =  
1.0, drug_configs: list =  
None, receiving_drugs_event:  
emod_api.interventions.common.BroadcastEvent  
= None, repetitions: int = 1,  
tsteps_btwn_repetitions: int  
= 60, treatment_delay: int  
= 0, diagnostic_type: str =  
'BLOOD_SMEAR_PARASITES',  
diagnostic_threshold: float =  
40, measurement_sensitivity:  
float = 0.1, node_ids: list  
= None, expire_recent_drugs:  
emod_api.interventions.common.PropertyValueChanger  
= None,  
node_property_restrictions:  
list = None,  
ind_property_restrictions: list =  
None, disqualifying_properties:  
list = None, target_group:  
any = 'Everyone', trigger  
condition_list: list = None,  
triggered_campaign_delay: int  
= 0, listening_duration: int = -  
1, check_eligibility_at_trigger:  
bool = False)
```

Add a MSAT (mass screening and treatment) drug intervention to campaign. This is either scheduled (on days from `start_days`) or triggered (when `trigger_condition_list` is present).

Parameters

- **camp** – The `DTKConfigBuilder` object for building, modifying, and writing campaign configuration files.
- **start_days** – List of days on which to start the intervention.
- **coverage** – Demographic coverage of the intervention.
- **drug_configs** – List of dictionaries of drug configurations to be distributed, created in `add_drug_campaign`.
- **receiving_drugs_event** – (Optional) Broadcast event container with event to be broadcast when drugs received.
- **repetitions** – How many times the intervention will be repeated.
- **tsteps_btwn_repetitions** – Time steps between repetitions.
- **treatment_delay** – Delay before the triggered drug distribution is done.
- **diagnostic_type** – Diagnostic type. Available options are: `TRUE_INFECTION_STATUS`, `BLOOD_SMEAR`, `PCR`, `PF_HRP2`, `TRUE_PARASITE_DENSITY`, `HAS_FEVER`.
- **diagnostic_threshold** – Diagnostic threshold values, which are based on the selected diagnostic type.
- **measurement_sensitivity** – setting for `Measurement_Sensitivity` in `MalariaDiagnostic`.
- **node_ids** – The list of nodes to apply this intervention to (`Node_List` parameter). If not provided, set value of `NodeSetAll`.
- **expire_recent_drugs** – `PropertyValueChanger` intervention that updates `DrugStatus` to Recent drug in `IndividualProperties`.
- **node_property_restrictions** – List of `NodeProperty` key:value pairs that nodes must have to receive the diagnostic intervention. For example, `[{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]`. Default is no restrictions.
- **ind_property_restrictions** – List of `IndividualProperty` key:value pairs that individuals must have to receive the diagnostic intervention. For example, `[{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]`. Default is no restrictions.
- **disqualifying_properties** – List of `IndividualProperty` key:value pairs that cause an intervention to be aborted. For example, `[{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]`.
- **target_group** – A dictionary of `{'agemin': x, 'agemax': y}` to target MDA, SMC, MSAT, fMDA to individuals between x and y years of age. Default is Everyone.
- **trigger_condition_list** – List of events to trigger MSAT. When `trigger_string` is set, the first entry of `start_days` is the day that is used to start listening for the trigger(s), the campaign happens when the trigger(s) is received.

- **triggered_campaign_delay** – How long to delay the actual intervention (drug giving) for after the trigger is received.
- **listening_duration** – Duration to listen for the trigger. Default is -1, which listens indefinitely.
- **check_eligibility_at_trigger** – If triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay. Default is false, checks at distribution.

Returns None

```
emodpy_malaria.interventions.drug_campaign.add_fmda(camp, start_days: list =  
None, trigger_coverage:  
float = 1, coverage: float  
= 1, drug_configs: list =  
None, receiving_drugs_event:  
emod_api.interventions.common.BroadcastEvent  
= None, repetitions: int = 1,  
tsteps_btwn_repetitions: int  
= 365, treatment_delay: int  
= 0, diagnostic_type: str =  
'BLOOD_SMEAR_PARASITES',  
diagnostic_threshold: float = 40,  
measurement_sensitivity: float  
= 0.1, fmda_radius: int = 0,  
node_selection_type: str = 'DIS-  
TANCE_ONLY', node_ids: list  
= None, expire_recent_drugs:  
emod_api.interventions.common.PropertyValueChanger  
= None,  
node_property_restrictions:  
list = None,  
ind_property_restrictions: list =  
None, disqualifying_properties:  
list = None, target_group:  
any = 'Everyone', trigger  
condition_list: list = None,  
listening_duration: int = - 1,  
triggered_campaign_delay: int  
= 0, check_eligibility_at_trigger:  
bool = False)
```

Add a fMDA (focal mass drug administration) drug intervention to campaign. The fMDA is based on results from diagnostic survey, which is either scheduled (on days from start_days) or triggered (when trigger_condition_list is present).

Parameters

- **camp** – The `DTKConfigBuilder` object for building, modifying, and writing campaign configuration files.
- **start_days** – List of days on which to start the intervention.
- **trigger_coverage** – Demographic coverage of the triggered intervention.
- **coverage** – Demographic coverage of the intervention.
- **drug_configs** – List of dictionaries of drug configurations to be distributed, created in `add_drug_campaign`.

- **receiving_drugs_event** – (Optional) Broadcast event container with event to be broadcast when drugs received.
- **repetitions** – How many times the intervention will be repeated.
- **tsteps_btwn_repetitions** – Time steps between repetitions.
- **treatment_delay** – Delay before the triggered drug distribution is done.
- **diagnostic_type** – Diagnostic type. Available options are: * TRUE_INFECTION_STATUS * BLOOD_SMEAR * PCR * PF_HRP2 * TRUE_PARASITE_DENSITY * HAS_FEVER
- **diagnostic_threshold** – Diagnostic threshold values, which are based on the selected diagnostic type.
- **measurement_sensitivity** – setting for **Measurement_Sensitivity** in MalariaDiagnostic
- **fmda_radius** – Radius of the follow up BroadcastToOtherNodes interventions, uses node_selection_type.
- **node_selection_type** – Node selection type for broadcasting fMDA trigger. Available options are: * DISTANCE_ONLY: Nodes located within the distance specified by fmda_type are selected. * MIGRATION_NODES_ONLY: Nodes that are local or regional are selected. * DISTANCE_AND_MIGRATION: Nodes are selected using DISTANCE_ONLY and MIGRATION_NODES_ONLY criteria.
- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of NodeSetAll.
- **expire_recent_drugs** – PropertyValueChanger intervention that updates DrugStatus to Recent drug in IndividualProperties.
- **node_property_restrictions** – List of NodeProperty key:value pairs that nodes must have to receive the diagnostic intervention. For example, [{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]. Default is no restrictions.
- **ind_property_restrictions** – List of IndividualProperty key:value pairs that individuals must have to receive the diagnostic intervention. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]. Default is no restrictions.
- **disqualifying_properties** – List of IndividualProperty key:value pairs that cause an intervention to be aborted. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].
- **target_group** – A dictionary of {'agemin': x, 'agemax': y} to target MDA, SMC, MSAT, fMDA to individuals between x and y years of age. Default is Everyone.
- **trigger_condition_list** – List of events to trigger fMDA. When trigger_string is set, the first entry of start_days is the day that is used to start listening for the trigger(s), the campaign happens when the trigger(s) is received.
- **listening_duration** – Duration to listen for the trigger. Default is -1, which listens indefinitely.
- **triggered_campaign_delay** – How long to delay the actual intervention (drug giving) after the trigger is received.

- **check_eligibility_at_trigger** – If triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay. Default is false, checks at distribution.

Returns None

```
emodpy_malaria.interventions.drug_campaign.add_rfMSAT(camp, start_day: int =  
0, coverage: float = 1,  
drug_configs: list = None,  
receiving_drugs_event:  
emod_api.interventions.common.BroadcastEvent  
= None, listening_duration:  
int = - 1, treatment_delay: int  
= 0, trigger_coverage: float  
= 1, diagnostic_type: str =  
'BLOOD_SMEAR_PARASITES',  
diagnostic_threshold: float =  
40, measurement_sensitivity:  
float = 0.1, fmda_radius: int  
= 0, node_selection_type: str  
= 'DISTANCE_ONLY', snowballs: int = 0, node_ids: list  
= None, expire_recent_drugs:  
emod_api.interventions.common.PropertyValueChange  
= None,  
node_property_restrictions:  
list = None,  
ind_property_restrictions:  
list = None, disqualifying_properties: list = None)
```

Add a rfMDA (reactive focal mass drug administration) drug intervention to campaign. Detecting malaria triggers diagnostic surveys to run on neighboring nodes and so on, up to the number of triggered interventions defined in **snowballs** parameter.

Parameters

- **camp** – The DTKConfigBuilder object for building, modifying, and writing campaign configuration files.
- **start_day** – List of days on which to start the intervention.
- **coverage** – Demographic coverage of the intervention.
- **drug_configs** – List of dictionaries of drug configurations to be distributed, created in add_drug_campaign.
- **receiving_drugs_event** – (Optional) Broadcast event container with event to be broadcast when drugs received.
- **listening_duration** – Duration of the existence of the intervention.
- **treatment_delay** – delay before the triggered drug distribution is done
- **trigger_coverage** – Demographic coverage for the triggered intervention
- **diagnostic_type** – Diagnostic type. Available options are: *
TRUE_INFECTION_STATUS * BLOOD_SMEAR * PCR * PF_HRP2 *
TRUE_PARASITE_DENSITY * HAS_FEVER
- **diagnostic_threshold** – Diagnostic threshold values, which are based on the selected diagnostic type.

- **measurement_sensitivity** – setting for **Measurement_Sensitivity** in MalariaDiagnostic
- **fmda_radius** – Radius of the follow up BroadcastToOtherNodes interventions, uses node_selection_type.
- **node_selection_type** – Node selection type for broadcasting fMDA trigger. Available options are: * **DISTANCE_ONLY**: Nodes located within the distance specified by fmda_type are selected. * **MIGRATION_NODES_ONLY**: Nodes that are local or regional are selected. * **DISTANCE_AND_MIGRATION**: Nodes are selected using **DISTANCE_ONLY** and **MIGRATION_NODES_ONLY** criteria.
- **snowballs** – Number of triggered interventions after the first.
- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of NodeSetAll.
- **expire_recent_drugs** – PropertyValueChanger intervention that updates DrugStatus to Recent drug in IndividualProperties.
- **node_property_restrictions** – List of NodeProperty key:value pairs that nodes must have to receive the diagnostic intervention. For example, [{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]. Default is no restrictions.
- **ind_property_restrictions** – List of IndividualProperty key:value pairs that individuals must have to receive the diagnostic intervention. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]. Default is no restrictions.
- **disqualifying_properties** – List of IndividualProperty key:value pairs that cause an intervention to be aborted. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].

Returns None

```
emodpy_malaria.interventions.drug_campaign.add_rfMDA(camp, start_day: int = 0, coverage: float = 1, drug_configs: list = None, receiving_drugs_event: emod_api.interventions.common.BroadcastEvent = None, listening_duration: int = -1, treatment_delay: int = 0, trigger_coverage: float = 1, fmda_radius: int = 0, node_selection_type: str = 'DISTANCE_ONLY', node_ids: list = None, expire_recent_drugs: emod_api.interventions.common.PropertyValueChanger = None, node_property_restrictions: list = None, ind_property_restrictions: list = None, disqualifying_properties: list = None)
```

This function adds two interventions to your campaign file: 1) “Received_Treatment”- triggered BroadcastEventToOtherNodes of “Give_Drugs_rfMDA” event to fmda_radius, with a “treatment_delay” option,

and “coverage” coverage. 2) “Give_Drugs_rfMDA” event-triggered MultiIntervention event distributing “drug_configs” drugs, with “trigger_coverage” demographic coverage, with an option to restrict to {“DrugStatus”: “None”} IndividualProperty and optional receiving_drugs_event.

Upon “Received_Treatment” event a delayed “Give_Drugs_rfMDA” is sent out to (optionally) neighboring nodes, which triggers giving of drugs, and (optionally) an individualproperty change and another event being sent.

Parameters

- **camp** – The DTKConfigBuilder object for building, modifying, and writing campaign configuration files.
- **start_day** – The day the intervention is distributed. Default is 0.
- **coverage** – Demographic coverage of the intervention.
- **drug_configs** – List of dictionaries of drug configurations to be distributed, created in add_drug_campaign.
- **receiving_drugs_event** – Event to be sent out upon receiving drugs. Default is 1, everyone.
- **listening_duration** – Duration of the existence of the intervention. Default is ongoing.
- **treatment_delay** – Delay of treatment (in days) after intervention is triggered.
- **trigger_coverage** – Demographic coverage for intervention triggered by successful treatment.
- **fmda_radius** – Radius (km) of sending event to other nodes.
- **node_selection_type** – Node selection type. Available options are: * DISTANCE_ONLY: Nodes located within the distance specified by fmda_type are selected. * MIGRATION_NODES_ONLY: Nodes that are local or regional are selected. * DISTANCE_AND_MIGRATION: Nodes are selected using DISTANCE_ONLY and MIGRATION_NODES_ONLY criteria.
- **node_ids** – The list of nodes to apply this intervention to (Node_List parameter). If not provided, set value of NodeSetAll.
- **expire_recent_drugs** – PropertyValueChanger intervention that updates DrugStatus to Recent drug in IndividualProperties.
- **node_property_restrictions** – List of NodeProperty key:value pairs that nodes must have to receive the diagnostic intervention. For example, [{"NodeProperty1": "PropertyValue1"}, {"NodeProperty2": "PropertyValue2"}]. Default is no restrictions.
- **ind_property_restrictions** – List of IndividualProperty key:value pairs that individuals must have to receive the diagnostic intervention. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}]. Default is no restrictions.
- **disqualifying_properties** – List of IndividualProperty key:value pairs that cause an intervention to be aborted. For example, [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].

Returns None

```
emodpy_malaria.interventions.drug_campaign.fmda_cfg(camp, fmda_type: any = 0,
                                                    node_selection_type: str = 'DISTANCE_ONLY', event_trigger:
                                                    str = 'Give_Drugs')
```

By Default, this is within the node-only (Distance_Only with distance=0).

Parameters

- **fmda_type** – Radius in km of the follow up BroadcastToOtherNodes interventions, uses node_selection_type.
- **node_selection_type** – Node selection type for broadcasting to other nodes. Available options are: * DISTANCE_ONLY: Nodes located within the distance specified by fmda_type are selected. * MIGRATION_NODES_ONLY: Nodes that are local or regional are selected. * DISTANCE_AND_MIGRATION: Nodes are selected using DISTANCE_ONLY and MIGRATION_NODES_ONLY criteria.
- **event_trigger** – String that triggers the broadcast.

Returns Configured BroadcastEventToOtherNodes

emodpy_malaria.interventions.inputeir module

```
emodpy_malaria.interventions.inputeir.new_intervention(camp, monthly_eir,
                                                         age_dependence)
```

```
emodpy_malaria.interventions.inputeir.InputEIR(camp, eir, start_day=1, node_ids=None,
                                                  age_dep='OFF')
```

InpputEIR intervention wrapper.

```
emodpy_malaria.interventions.inputeir.new_intervention_as_file(camp, start_day,
                                                                eir, file_name=None)
```

emodpy_malaria.interventions.irs module

```
emodpy_malaria.interventions.irs.IRSHousingModification(camp, start_day, coverage=1.0, blocking=1.0,
                                                         blocking_eff=1, killing_eff=1,
                                                         insecticide=None,
                                                         node_ids=None)
```

MCV1 Campaign :param coverage: Demographic Coverage :param blocking: :param killing: Note Start_Day is initialized as 1, recommend that this be aligned with the start of the simulation

```
emodpy_malaria.interventions.irs.new_intervention_as_file(camp, start_day, file_name=None)
```

emodpy_malaria.interventions.ivermectin module

```
emodpy_malaria.interventions.ivermectin.Ivermectin(schema_path_container,  
                                                    killing_effect,      start_day=1,  
                                                    target_coverage=1.0,      tar-  
                                                    get_num_individuals=None,  
                                                    killing_duration_box=0,  
                                                    killing_exponential_decay_rate=0)
```

Parameters

- **start_day** – day to give out this ivermectin
- **target_coverage** – probability of choosing an individual
- **target_num_individuals** – number of individuals to choose
- **killing_effect** – initial parasite killing effect
- **killing_duration_box** – box duration for killing effect
- **killing_exponential_decay_rate** – rate at which killing effect decays per day.
Use 0 for box duration only.

Returns: campaign event

emodpy_malaria.interventions.mosquitorelease module

```
emodpy_malaria.interventions.mosquitorelease.MosquitoRelease(camp, start_day,  
                                                                by_number=True,  
                                                                number=10000,  
                                                                percentage=0.1,  
                                                                infectious=0.0,  
                                                                species='arabiensis',  
                                                                genome=[['X',  
                                                                'X']],  
                                                                node_ids=None)
```

Release mosquitoes of a given species and genome into each node

Parameters

- **start_day** – The day to release the vectors.
- **by_number** – True if releasing a fixed number of vectors else a percentage of the current population of the gender specified in the genome.
- **number** – The number of vectors to release.
- **percentage** – The percentage of the current poulation of mosquitoes to release. The 'population' will depend on the gender of the mosquitos being released and it will be the population from the previous time step.
- **infectious** – The fraction of vectors released that are to be infectious. One can only use this feature when 'Malaria_Model'!='MALARIA_MECHANISTIC_MODEL_WITH_PARASITE_GENETICS' and there are no 'Genome_Markers'.
- **species** – The case sensitive name of the species of vectors to be released.
- **genome** – This defines the alleles of the genome of the vectors to be released. It must define all of the alleles including the gender 'gene'. '*' is not allowed.

- **node_ids** – The list of node IDs to receive a release of vectors. The same number of vectors will be released to each node.

Returns new event

```
emodpy_malaria.interventions.mosquitorelease.new_intervention_as_file(camp,
                                                                    start_day,
                                                                    file-
                                                                    name=None)
```

emodpy_malaria.interventions.outdoorrestkill module

```
emodpy_malaria.interventions.outdoorrestkill.OutdoorRestKill(schema_path_container,
                                                                killing_effect,
                                                                insecti-
                                                                cide_name=None,
                                                                start_day=1, target_
                                                                get_coverage=1.0,
                                                                killing_predecay_duration=0,
                                                                killing_decay_rate=0,
                                                                node_ids=None)
```

Parameters

- **schema_path_container** –
- **killing_effect** –
- **insecticide_name** –
- **start_day** –
- **target_coverage** –
- **killing_predcay_duration** –
- **killing_decay_rate** –

Returns:

emodpy_malaria.interventions.spacespraying module

```
emodpy_malaria.interventions.spacespraying.SpaceSpraying(camp, start_day, cover-
                                                            age=1.0, killing_eff=1,
                                                            insecticide=None, con-
                                                            stant_duration=100,
                                                            node_ids=None)
```

MCV1 Campaign :param coverage: Demographic Coverage :param blocking: :param killing: Note Start_Day is initialized as 1, recommend that this be aligned with the start of the simulation

```
emodpy_malaria.interventions.spacespraying.new_intervention_as_file(camp,
                                                                    start_day,
                                                                    file-
                                                                    name=None)
```

emodpy_malaria.interventions.sugartrap module

```
emodpy_malaria.interventions.sugartrap.SugarTrap(camp, start_day, coverage=1.0, killing_eff=1, insecticide=None, constant_duration=100, node_ids=None)
```

SugarTrap intervention wrapper.

```
emodpy_malaria.interventions.sugartrap.new_intervention_as_file(camp, start_day, filename=None)
```

emodpy_malaria.interventions.treatment_seeking module

```
emodpy_malaria.interventions.treatment_seeking.add(camp, start_day: int = 1, targets: list = None, drug: list = None, node_ids: list = None, ind_property_restrictions: list = None, drug_ineligibility_duration: int = 0, duration: int = - 1, broadcast_event_name: str = 'ReceivedTreatment')
```

Add an event-triggered drug-seeking behavior intervention to the campaign using the **NodeLevelHealthTriggeredIV**. The intervention will distribute drugs to targeted individuals within the node.

Parameters

- **camp** – object for building, modifying, and writing campaign configuration files.
- **start_day** – Start day of intervention.
- **targets** – List of dictionaries defining the trigger event and coverage for and
- **of individuals to target with the intervention. Default is** (*properties*) – [{ "trigger": "NewClinicalCase", "coverage": 0.8, "agemin": 15, "agemax": 70, "seek": 0.4, "rate": 0.3}, {"trigger": "NewSevereCase", "coverage": 0.8, "seek": 0.6, "rate": 0.5}].
- **drug** – List of drug(s) to administer. Default is ["Artemether", "Lumefantrine"].
- **node_ids** – The list of nodes to apply this intervention to (**Node_List**
- **If not provided, set value of NodeSetAll.** (*parameter*)) –
- **ind_property_restrictions** – List of IndividualProperty key:value pairs that
- **must have to receive the intervention. For example,** (*individuals*) – [{"IndividualProperty1": "PropertyValue1"}, {"IndividualProperty2": "PropertyValue2"}].
- **duration** – How long the intervention is active. Default is -1, where intervention
- **expires.** (*never*) –
- **broadcast_event_name** – Event to broadcast when successful health seeking behavior.
- **is ReceivedTreatment.** (*Default*) –

Returns None

emodpy_malaria.interventions.udbednet module

```
emodpy_malaria.interventions.udbednet.UDBednet (camp, start_day: int = 1, discard_config: dict = None, coverage: float = 1, ind_property_restrictions: list = None, blocking_eff: float = 0.9, blocking_constant_duration: int = 0, blocking_decay_rate: float = 0.0013698630136986301, killing_eff: float = 0.6, killing_constant_duration: int = 0, killing_decay_rate: float = 0.0006849315068493151, repelling_eff: float = 0, repelling_constant_duration: int = 0, repelling_decay_rate: float = 0.0006849315068493151, iv_name: str = 'UsageDependentBednet', age_dependence: dict = None, seasonal_dependence: dict = None, insecticide: str = None, node_ids: list = None, triggered_campaign_delay: dict = None, triggers: list = None, duration: int = -1, check_eligibility_at_trigger: bool = False)
```

Add an insecticide-treated net (ITN) intervention with a seasonal usage pattern to the campaign using the **UsageDependentBednet** class. The arguments **birth_triggered** and **triggered_condition_list** are mutually exclusive. If both are provided, **triggered_condition_list** is ignored. You must add the following custom events to your config.json:

- Bednet_Discarded
- Bednet_Got_New_One
- Bednet_Using

Parameters

- **start** – The day on which to start distributing the bednets (**Start_Day** parameter).
- **coverage** – Fraction of the population receiving bed nets in a given distribution event
- **blocking_config** – The value passed gets directly assigned to the **Blocking_Config** parameter. Durations are in days. Default is **blocking_config**= **WaningEffectExponential**(Decay_Time_Constant=730, Initial_Effect=0.9)

This could be dictionary such as:

```
{
  "Box_Duration": 3650,
  "Initial_Effect": 0,
  "class": "WaningEffectBox"
}
```

- **killing_config** – The value passed gets directly assigned to the **Killing_Config** parameter. Durations are in days. Default is **killing_config** = **WaningEffectExponential**(Decay_Time_Constant=1460, Initial_Effect=0.6)

This could be dictionary such as:

```
{
  "Box_Duration": 3650,
  "Initial_Effect": 0,
  "Decay_Time_Constant": 150,
  "class": "WaningEffectBoxExponential"
}
```

- **repelling_config** – The value passed gets directly assigned to the Repelling_Config parameter. Durations are in days. Default is repelling_config = WaningEffectExponential(Decay_Time_Constant=1460, Initial_Effect=0.0)

This could be dictionary such as:

```
{
  "Box_Duration": 3650,
  "Initial_Effect": 0,
  "Decay_Time_Constant": 150,
  "class": "WaningEffectBoxExponential"
}
```

- **discard_config** – A dictionary of parameters needed to define expiration distribution. No need to define the distribution with all its parameters Default is bednet being discarded with EXPONENTIAL_DISTRIBUTION with Expiration_Period_Exponential of 10 years

Examples:

```
for Gaussian: {"Expiration_Period_Distribution": "GAUSSIAN_
↪DISTRIBUTION",
  "Expiration_Period_Gaussian_Mean": 20, "Expiration_Period_
↪Gaussian_Std_Dev":10}
for Exponential {"Expiration_Period_Distribution": "EXPONENTIAL_
↪DISTRIBUTION",
  "Expiration_Period_Exponential":150}
```

- **age_dependence** – A dictionary defining the age dependence of net use. Must contain a list of ages in years and list of usage rate. Default is uniform across all ages. Times are in years of age Examples:

```
{"Times":[], "Values":[]} or {"youth_cov":0.7, "youth_min_age":3,
↪"youth_max_age":13}
```

- **seasonal_dependence** – A dictionary defining the seasonal dependence of net use. Default is constant use during the year. Times are given in days of the year; values greater than 365 are ignored. Dictionaries can be (times, values) for linear spline or (minimum coverage, day of maximum coverage) for sinusoidal dynamics. Times are days of the year Examples:

```
{"Times":[], "Values":[]} or {"min_cov":0.45, "max_day":300}
```

- **node_ids** – The list of nodes to apply this intervention to (**Node_List** parameter). If not provided, set value of NodeSetAll.
- **birth_triggered** – If true, event is specified as a birth-triggered intervention.
- **duration** – If run as a birth-triggered event or a trigger_condition_list, specifies the duration for the distribution to continue. Default is to continue until the end of the simulation.
- **triggered_campaign_delay** – (Optional) After the trigger is received, the number of time steps until the campaign starts. Eligibility of people or nodes for the cam-

paign is evaluated on the start day, not the triggered day. `triggered_campaign_delay` is a dict. Specify the actual delay distribution params, not the distribution type. E.g., { "Delay_Distribution_Constant": 14" } Delay is in days

- **trigger_condition_list** – (Optional) A list of the events that will trigger the ITN intervention. If included, **start** is the day when monitoring for triggers begins. This argument cannot configure birth-triggered ITN (use **birth_triggered** instead).
- **ind_property_restrictions** – The IndividualProperty key:value pairs that individuals must have to receive the intervention (**Property_Restrictions_Within_Node** parameter). In the format [{ "BitingRisk": "High"}, { "IsCool": "Yes"}].
- **node_property_restrictions** – The NodeProperty key:value pairs that nodes must have to receive the intervention (**Node_Property_Restrictions** parameter). In the format [{ "Place": "RURAL"}, { "ByALake": "Yes"}]
- **check_eligibility_at_trigger** – if triggered event is delayed, you have an option to check individual/node's eligibility at the initial trigger or when the event is actually distributed after delay.

Returns None

NOTE: Previous way of setting discard config is no longer available, you can translate it to the current way by: `discard_config` the old way { 'halflife1': 260, 'halflife2': 2106, 'fraction1': float(table_dict['fast_fraction']) } `discard_config` translated = { "Expiration_Period_Distribution": "DUAL_EXPONENTIAL_DISTRIBUTION", "Expiration_Period_Mean_1": discard_halflife, or halflife1 "Expiration_Period_Mean_2": 365 * 40, or halflife2 "Expiration_Period_Proportion_1": 1 or 'fraction1' }

Example:

```
discard_config = {"Expiration_Period_Exponential": 10 * 365}
age_dependence = {"Times": [0, 4, 10, 60],
                  "Values": [1, 0.9, 0.8, 0.5]}
add_ITN_age_season(config_builder, start=1, coverage=1, killing_config=killing_
    ↪config,
                    blocking_config=blocking_config, discard_config = discard_config
                    age_dependence=age_dependence, cost=5, birth_triggered=True,
    ↪duration=-1,
                    node_property_restrictions=[{"Place": "Rural"}]):
```

```
emodpy_malaria.interventions.udbednet.new_intervention_as_file(camp,
                                                                start_day, file-
                                                                name=None)
```

emodpy_malaria.reporters package

Submodules

emodpy_malaria.reporters.builtin module

```
class emodpy_malaria.reporters.builtin.ReportVectorGenetics (class_name: str =
    None, parameters:
    dict = <factory>,
    Enabled: bool =
    True, Pretty_Format:
    bool = True)
```

Bases: `emodpy.reporters.base.BuiltInReporter`

```
    config (config_builder, manifest)
    parameters:    dict
class emodpy_malaria.reporters.builtin.ReportVectorStats (class_name: str = None,
                                                            parameters: dict = <factory>, Enabled: bool
                                                            = True, Pretty_Format:
                                                            bool = True)

    Bases: emodpy.reporters.base.BuiltInReporter
    config (config_builder, manifest)
    parameters:    dict
class emodpy_malaria.reporters.builtin.MalariaSummaryReport (class_name: str =
                                                            None, parameters:
                                                            dict = <factory>,
                                                            Enabled: bool =
                                                            True, Pretty_Format:
                                                            bool = True)

    Bases: emodpy.reporters.base.BuiltInReporter
    config (config_builder, manifest)
    parameters:    dict
class emodpy_malaria.reporters.builtin.MalariaPatientJSONReport (class_name:
                                                            str = None, pa-
                                                            rameters: dict
                                                            = <factory>,
                                                            Enabled:
                                                            bool = True,
                                                            Pretty_Format:
                                                            bool = True)

    Bases: emodpy.reporters.base.BuiltInReporter
    config (config_builder, manifest)
    parameters:    dict
class emodpy_malaria.reporters.builtin.MalariaTransmissionReport (class_name:
                                                            str = None,
                                                            parame-
                                                            ters: dict =
                                                            <factory>,
                                                            Enabled:
                                                            bool = True,
                                                            Pretty_Format:
                                                            bool = True)

    Bases: emodpy.reporters.base.BuiltInReporter
    config (config_builder, manifest)
    parameters:    dict
```

```

class emodpy_malaria.reporters.builtin.FilteredMalariaReport (class_name: str
                                                             = None, param-
                                                             eters: dict =
                                                             <factory>, En-
                                                             abled: bool = True,
                                                             Pretty_Format:
                                                             bool = True)

Bases: emodpy.reporters.base.BuiltInReporter

config (config_builder, manifest)

parameters: dict

class emodpy_malaria.reporters.builtin.ReportEventCounter (class_name: str =
                                                            None, parameters:
                                                            dict = <factory>, En-
                                                            abled: bool = True,
                                                            Pretty_Format: bool =
                                                            True)

Bases: emodpy.reporters.base.BuiltInReporter

config (config_builder, manifest)

parameters: dict

```

2.1.2 Submodules

emodpy_malaria.config module

`emodpy_malaria.config.get_file_from_http(url)`

Get data files from simple http server.

`emodpy_malaria.config.set_team_defaults(config, mani)`

Set configuration defaults using team-wide values, including drugs and vector species.

`emodpy_malaria.config.set_team_vs_params(config, mani)`

`emodpy_malaria.config.get_species_params(cb, species)`

`emodpy_malaria.config.set_team_drug_params(config, mani)`

`emodpy_malaria.config.get_drug_params(cb, drug_name)`

`emodpy_malaria.config.set_species(config, species_to_select)`

Use this function to specify which mosquito species to use in the simulation.

Parameters

- **config** – schema-backed config smart dict
- **species_to_list** – list of 1 or more strings.

Returns

None

`emodpy_malaria.config.set_resistances(config)`

Use this function after you're done calling add_resistance. config is the input and the output

`emodpy_malaria.config.add_alleles(allele_names_in, allele_inits_in)`

This is public API function for user to add alleles. User specifies the list of alleles and corresponding initial distribution.

`emodpy_malaria.config.add_mutation` (*from_allele, to_allele, rate*)

Public API function for user to add mutations as part of vector genetics configuration. A mutation is specified with a source allele, a destination allele, and a rate

`emodpy_malaria.config.add_trait` (*manifest, allele_combo, trait_name, trait_value*)

Use this function to add traits as part of vector genetics configuration. Should produce something like:

```
{
  "Allele_Combinations": [{"X", "X"}, {"a0", "a1"}],
  "Trait_Modifiers": {"INFECTED_BY_HUMAN": 0}
},
```

`emodpy_malaria.config.add_resistance` (*manifest, insecticide_name, species, combo, blocking=1.0, killing=1.0, repelling=1.0, larval_killing=1.0*)

Use this function to add insecticide resistances. An insecticide can have a list of resistances. Add each resistance separately with the same name:

```
Insecticides = [
{
  "Name": "pyrethroid",
  "Resistances": [
    {
      "Allele_Combinations": [
        [
          "a1",
          "a1"
        ]
      ],
      "Blocking_Modifier": 1.0,
      "Killing_Modifier": pyrethroid_killing,
      "Species": "gambiae"
    }
  ]
},
],
```

`emodpy_malaria.config.set_genetics` (*vsp, manifest*)

Don't need to pass these anymore since they are module variables. But actually need to try with more than one set and see where I end up in terms of design.

PYTHON MODULE INDEX

e

- `emodpy_malaria`, 5
- `emodpy_malaria.config`, 33
- `emodpy_malaria.demographics`, 5
- `emodpy_malaria.demographics.MalariaDemographics`, 5
- `emodpy_malaria.interventions`, 7
- `emodpy_malaria.interventions.bednet`, 7
- `emodpy_malaria.interventions.common`, 8
- `emodpy_malaria.interventions.diag_survey`, 9
- `emodpy_malaria.interventions.drug`, 12
- `emodpy_malaria.interventions.drug_campaign`, 12
- `emodpy_malaria.interventions.inpoteir`, 25
- `emodpy_malaria.interventions.irs`, 25
- `emodpy_malaria.interventions.ivermectin`, 26
- `emodpy_malaria.interventions.mosquitorelease`, 26
- `emodpy_malaria.interventions.outdoorrestkill`, 27
- `emodpy_malaria.interventions.spacespraying`, 27
- `emodpy_malaria.interventions.sugartrap`, 28
- `emodpy_malaria.interventions.treatment_seeking`, 28
- `emodpy_malaria.interventions.udbednet`, 29
- `emodpy_malaria.reporters`, 31
- `emodpy_malaria.reporters.builtin`, 31

A

add() (in module *emodpy_malaria.interventions.treatment-seeking*), 28

add_alleles() (in module *emodpy_malaria.config*), 33

add_diagnostic_survey() (in module *emodpy_malaria.interventions.diag_survey*), 9

add_drug_campaign() (in module *emodpy_malaria.interventions.drug_campaign*), 13

add_fMDA() (in module *emodpy_malaria.interventions.drug_campaign*), 20

add_MDA() (in module *emodpy_malaria.interventions.drug_campaign*), 16

add_MSAT() (in module *emodpy_malaria.interventions.drug_campaign*), 18

add_mutation() (in module *emodpy_malaria.config*), 33

add_resistance() (in module *emodpy_malaria.config*), 34

add_rfMDA() (in module *emodpy_malaria.interventions.drug_campaign*), 23

add_rfMSAT() (in module *emodpy_malaria.interventions.drug_campaign*), 22

add_trait() (in module *emodpy_malaria.config*), 34

AntiMalarialDrug() (in module *emodpy_malaria.interventions.common*), 9

AntiMalarialDrug() (in module *emodpy_malaria.interventions.drug*), 12

B

BasicBednet() (in module *emodpy_malaria.interventions.bednet*), 8

Bednet() (in module *emodpy_malaria.interventions.bednet*), 7

C

clear() (in module *emodpy_malaria.reporters.builtin.FilteredMalariaReport* method), 33

config() (in module *emodpy_malaria.reporters.builtin.MalariaPatientJSONReport* method), 32

config() (in module *emodpy_malaria.reporters.builtin.MalariaSummaryReport* method), 32

config() (in module *emodpy_malaria.reporters.builtin.MalariaTransmissionReport* method), 32

config() (in module *emodpy_malaria.reporters.builtin.ReportEventCounter* method), 33

config() (in module *emodpy_malaria.reporters.builtin.ReportVectorGenetics* method), 31

config() (in module *emodpy_malaria.reporters.builtin.ReportVectorStats* method), 32

D

drug_configs_from_code() (in module *emodpy_malaria.interventions.drug_campaign*), 12

E

emodpy_malaria
module, 5

emodpy_malaria.config
module, 33

emodpy_malaria.demographics
module, 5

emodpy_malaria.demographics.MalariaDemographics
module, 5

emodpy_malaria.interventions
module, 7

emodpy_malaria.interventions.bednet
module, 7

emodpy_malaria.interventions.common
module, 8

emodpy_malaria.interventions.diag_survey
module, 9

emodpy_malaria.interventions.drug
module, 12

emodpy_malaria.interventions.drug_campaign
module, 12

emodpy_malaria.interventions.inpoteir	Ivermectin()	(in module
module, 25	emodpy_malaria.interventions.ivermectin),	
emodpy_malaria.interventions.irs	26	
module, 25		
emodpy_malaria.interventions.ivermectin	M	
module, 26	MalariaDemographics	(class in
emodpy_malaria.interventions.mosquitorelease	emodpy_malaria.demographics.MalariaDemographics),	
module, 26	5	
emodpy_malaria.interventions.outdoorrestkill	MalariaDiagnostic()	(in module
module, 27	emodpy_malaria.interventions.common),	
emodpy_malaria.interventions.spacespraying	8	
module, 27	MalariaPatientJSONReport	(class in
emodpy_malaria.interventions.sugartrap	emodpy_malaria.reporters.builtin), 32	
module, 28	MalariaSummaryReport	(class in
emodpy_malaria.interventions.treatment_seeking	emodpy_malaria.reporters.builtin), 32	
module, 28	MalariaTransmissionReport	(class in
emodpy_malaria.interventions.udbednet	emodpy_malaria.reporters.builtin), 32	
module, 29	module	
emodpy_malaria.reporters	emodpy_malaria, 5	
module, 31	emodpy_malaria.config, 33	
emodpy_malaria.reporters.builtin	emodpy_malaria.demographics, 5	
module, 31	emodpy_malaria.demographics.MalariaDemographics	
	5	
F	emodpy_malaria.interventions, 7	
FilteredMalariaReport	emodpy_malaria.interventions.bednet,	
(class in	7	
emodpy_malaria.reporters.builtin), 32	emodpy_malaria.interventions.common,	
fmda_cfg()	8	
(in module	emodpy_malaria.interventions.diag_survey,	
emodpy_malaria.interventions.drug_campaign),	9	
24	emodpy_malaria.interventions.drug,	
from_params()	12	
(in module	emodpy_malaria.interventions.drug_campaign,	
emodpy_malaria.demographics.MalariaDemographics),	12	
6	emodpy_malaria.interventions.inpoteir,	
from_pop_csv()	25	
(in module	emodpy_malaria.interventions.irs, 25	
emodpy_malaria.demographics.MalariaDemographics),	emodpy_malaria.interventions.ivermectin,	
5	26	
G	emodpy_malaria.interventions.mosquitorelease,	
get_drug_params()	26	
(in module	emodpy_malaria.interventions.outdoorrestkill,	
emodpy_malaria.config), 33	27	
get_file_from_http()	emodpy_malaria.interventions.spacespraying,	
(in module	27	
emodpy_malaria.config), 33	emodpy_malaria.interventions.sugartrap,	
get_species_params()	28	
(in module	emodpy_malaria.interventions.treatment_seeking,	
emodpy_malaria.config), 33	28	
I	emodpy_malaria.interventions.udbednet,	
InputEIR()	29	
(in module	emodpy_malaria.reporters, 31	
emodpy_malaria.interventions.inpoteir),	emodpy_malaria.reporters.builtin, 31	
25	MosquitoRelease()	(in module
IRSHousingModification()		
(in module		
emodpy_malaria.interventions.irs), 25		

- emodpy_malaria.interventions.mosquitorelease*), [S](#)
[26](#)
- N**
- new_intervention()* (in module *emodpy_malaria.interventions.inpoteir*), [25](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.bednet*), [8](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.drug*), [12](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.inpoteir*), [25](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.irs*), [25](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.mosquitorelease*), [27](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.spacespraying*), [27](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.sugartrap*), [28](#)
- new_intervention_as_file()* (in module *emodpy_malaria.interventions.udbednet*), [31](#)
- O**
- OutdoorRestKill()* (in module *emodpy_malaria.interventions.outdoorrestkill*), [27](#)
- P**
- parameters(emodpy_malaria.reporters.builtin.FilteredMalariaReport attribute)*, [33](#)
- parameters(emodpy_malaria.reporters.builtin.MalariaPatientJSONReport attribute)*, [32](#)
- parameters(emodpy_malaria.reporters.builtin.MalariaSummaryReport attribute)*, [32](#)
- parameters(emodpy_malaria.reporters.builtin.MalariaTransmissionReport attribute)*, [32](#)
- parameters(emodpy_malaria.reporters.builtin.ReportEventCounter attribute)*, [33](#)
- parameters(emodpy_malaria.reporters.builtin.ReportVectorGenetics attribute)*, [32](#)
- parameters(emodpy_malaria.reporters.builtin.ReportVectorStats attribute)*, [32](#)
- R**
- ReportEventCounter* (class in *emodpy_malaria.reporters.builtin*), [33](#)
- ReportVectorGenetics* (class in *emodpy_malaria.reporters.builtin*), [31](#)
- ReportVectorStats* (class in *emodpy_malaria.reporters.builtin*), [32](#)
- set_genetics()* (in module *emodpy_malaria.config*), [34](#)
- set_resistances()* (in module *emodpy_malaria.config*), [33](#)
- set_species()* (in module *emodpy_malaria.config*), [33](#)
- set_team_defaults()* (in module *emodpy_malaria.config*), [33](#)
- set_team_drug_params()* (in module *emodpy_malaria.config*), [33](#)
- set_team_vs_params()* (in module *emodpy_malaria.config*), [33](#)
- SpaceSpraying()* (in module *emodpy_malaria.interventions.spacespraying*), [27](#)
- SugarTrap()* (in module *emodpy_malaria.interventions.sugartrap*), [28](#)
- U**
- UDBednet()* (in module *emodpy_malaria.interventions.udbednet*), [29](#)