Effective Coverage vs. Lives Saved: A Comparison

In this document, we will find the optimal set of nutritional interventions over space and time, first using each micronutrient's effect on effective coverage separately, and then using all interventions together to see if they yield different optimal results, using a measure of the number of lives saved from each intervention.

Summary of Results

```
import import_ipynb
import os
import geopandas as gpd
os.chdir('/home/lordflaron/Documents/minimod')
import optimization_work.demewoz_lives_saved.lives_saved_analysis as lives_saved
import optimization_work.demewoz_lives_saved.folate_effective_coverage_analysis as folate
import optimization_work.demewoz_lives_saved.zinc_effective_coverage_analysis as zinc
import optimization_work.demewoz_lives_saved.vas_effective_coverage_analysis as vas
import pandas as pd

# First only get the optimal values
lives_saved_opt = lives_saved.models['lives_saved'][None].opt_df.loc[lambda df: df['opt_vals']>0].group
lives_saved_high_opt = lives_saved.models['lives_saved_high'][None].opt_df.loc[lambda df: df['opt_vals'
zinc_opt = zinc.models[None].opt_df.loc[lambda df: df['opt_vals']>0].groupby(['region','time']).sum()[[
vas_opt = vas.models[None].opt_df.loc[lambda df: df['opt_vals']>0].groupby(['region','time']).sum()[['c
folate_opt = folate.models[None].opt_df.loc[lambda df: df['opt_vals']>0].groupby(['region','time']).sum

# Get names of optimal interventions
lives_saved_name = lives_saved.models['lives_saved'][None].optimal_interventions
lives_saved_high_name = lives_saved.models['lives_saved_high'][None].optimal_interventions
zinc_name = zinc.models[None].optimal_interventions
vas_name = vas.models[None].optimal_interventions
folate_name = folate.models[None].optimal_interventions

# Get names of bau
lives_saved_bau = lives_saved.models['lives_saved'][None].bau_df.index.get_level_values(level='intervent
lives_saved_high_bau = lives_saved.models['lives_saved_high'][None].bau_df.index.get_level_values(level=
zinc_bau = zinc.models[None].bau_df.index.get_level_values(level='intervention').unique().tolist()
vas_bau = vas.models[None].bau_df.index.get_level_values(level='intervention').unique().tolist()
folate_bau = folate.models[None].bau_df.index.get_level_values(level='intervention').unique().tolist()

opt_dict = {'Zinc\n(Children Eff. Cov.)' : zinc_opt,
'VA\n(Children Eff. Cov.)' : vas_opt,
'Folic Acid\n(WRA Eff. Cov.)' : folate_opt,
'All (Lives Saved)' : lives_saved_opt,
'All (Lives Saved, Alt.)' : lives_saved_high_opt}

df_all = pd.concat(opt_dict.values(), axis=1)
```

```
df_all.columns = pd.MultiIndex.from_product([opt_dict.keys(), ['CB', 'CC', 'BAU* Cost per Benefit']])
```

```
importing Jupyter notebook from /home/lordflaron/Documents/minimod/optimization_work/demewoz_lives_save
.pipe(observation_adjustment,
int1 = "cube",
int2 = 0,
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubezcube",
int2 = 0,
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcube",
int2 = "maxoil",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcube",
int2 = "oil",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubevas",
int2 = "oilvas",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubevas",
int2 = "maxoilvas",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubevas",
int2 = "vas",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubeclinic",
int2 = "clinic",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubeclinic",
int2 = "maxoilclinic",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubeclinic",
int2 = "oilclinic",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubezflour",
int2 = "zflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubezflour",
int2 = "maxoilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubezflour",
```

```
int2 = "oilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubevaszflour",
int2 = "oilvaszflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubevaszflour",
int2 = "vaszflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubevaszflour",
int2 = "maxoilvaszflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubecliniczflour",
int2 = "oilcliniczflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubecliniczflour",
int2 = "maxoilcliniczflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "cubezflourzcube",
int2 = "zflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubezflourzcube",
int2 = "maxoilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubezflourzcube",
int2 = "oilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubevaszflourzcube",
int2 = "oilvaszflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilcubecliniczflourzcube",
int2 = "oilcliniczflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubecliniczflourzcube",
int2 = "maxoilcliniczflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "zcube",
int2 = 0,
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "zflourzcube",
int2 = "zflour",
time_to_replace = [1,2,3])
```

```
.pipe(observation_adjustment,
int1 = "oilvaszflourzcube",
int2 = "oilvaszflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoilcubezflourzcube",
int2 = "maxoilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilzflourzcube",
int2 = "oilzflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "oilzflourfcubefflour",
int2 = "oilzflourfflour",
time_to_replace = [1,2,3])
.pipe(observation_adjustment,
int1 = "maxoil",
int2 = "oil",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcube",
int2 = "oilcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilvas",
int2 = "oilvas",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubevas",
int2 = "oilcubevas",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilclinic",
int2 = "oilclinic",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubeclinic",
int2 = "oilcubeclinic",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilzflour",
int2 = "oilzflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubezflour",
int2 = "oilcubezflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilvaszflour",
int2 = "oilvaszflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcliniczflour",
```

```
int2 = "oilcliniczflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubevaszflour",
int2 = "oilcubevaszflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubecliniczflour",
int2 = "oilcubecliniczflour",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilzflourzcube",
int2 = "oilzflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubezflourzcube",
int2 = "oilcubezflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilvaszflourzcube",
int2 = "oilvaszflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcliniczflourzcube",
int2 = "oilcliniczflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubevaszflourzcube",
int2 = "oilcubevaszflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "maxoilcubecliniczflourzcube",
int2 = "oilcubecliniczflourzcube",
time_to_replace = [1,2])
.pipe(observation_adjustment,
int1 = "zflourfflour33",
int2 = 0,
time_to_replace = slice(None))
.pipe(observation_adjustment,
int1 = "fflour33",
int2 = 0,
time_to_replace = slice(None))
Changed cube to 0
Changed cubezcube to 0
Changed maxoilcube to maxoil
Changed oilcube to oil
Changed oilcubevas to oilvas
Changed maxoilcubevas to maxoilvas
Changed cubevas to vas
Changed cubeclinic to clinic
Changed maxoilcubeclinic to maxoilclinic
Changed oilcubeclinic to oilclinic
Changed cubezflour to zflour
Changed maxoilcubezflour to maxoilzflour
```

```
Changed oilcubezflour to oilzflour
Changed oilcubevaszflour to oilvaszflour
Changed cubevaszflour to vaszflour
Changed maxoilcubevaszflour to maxoilvaszflour
Changed oilcubecliniczflour to oilcliniczflour
Changed maxoilcubecliniczflour to maxoilcliniczflour
Changed cubezflourzcube to zflour
Changed maxoilcubezflourzcube to maxoilzflour
Changed oilcubezflourzcube to oilzflour
Changed oilcubevaszflourzcube to oilvaszflour
Changed oilcubecliniczflourzcube to oilcliniczflour
Changed maxoilcubecliniczflourzcube to maxoilcliniczflour
Changed zcube to 0
Changed zflourzcube to zflour
Changed oilvaszflourzcube to oilvaszflour
Changed maxoilcubezflourzcube to maxoilzflour
Changed oilzflourzcube to oilzflour
Changed oilzflourfcubefflour to oilzflourfflour
Changed maxoil to oil
Changed maxoilcube to oilcube
Changed maxoilvas to oilvas
Changed maxoilcubevas to oilcubevas
Changed maxoilclinic to oilclinic
Changed maxoilcubeclinic to oilcubeclinic
Changed maxoilzflour to oilzflour
Changed maxoilcubezflour to oilcubezflour
Changed maxoilvaszflour to oilvaszflour
Changed maxoilcliniczflour to oilcliniczflour
Changed maxoilcubevaszflour to oilcubevaszflour
Changed maxoilcubecliniczflour to oilcubecliniczflour
Changed maxoilzflourzcube to oilzflourzcube
Changed maxoilcubezflourzcube to oilcubezflourzcube
Changed maxoilvaszflourzcube to oilvaszflourzcube
Changed maxoilcliniczflourzcube to oilcliniczflourzcube
Changed maxoilcubevaszflourzcube to oilcubevaszflourzcube
Changed maxoilcubecliniczflourzcube to oilcubecliniczflourzcube
Changed zflourfflour33 to 0
Changed fflour33 to 0
Changed cube to 0
Changed cubezcube to 0
Changed maxoilcube to maxoil
Changed oilcube to oil
Changed oilcubevas to oilvas
Changed maxoilcubevas to maxoilvas
Changed cubevas to vas
Changed cubeclinic to clinic
Changed maxoilcubeclinic to maxoilclinic
Changed oilcubeclinic to oilclinic
Changed cubezflour to zflour
Changed maxoilcubezflour to maxoilzflour
Changed oilcubezflour to oilzflour
Changed oilcubevaszflour to oilvaszflour
Changed cubevaszflour to vaszflour
Changed maxoilcubevaszflour to maxoilvaszflour
```

```
Changed oilcubecliniczflour to oilcliniczflour
Changed maxoilcubecliniczflour to maxoilcliniczflour
Changed cubezflourzcube to zflour
Changed maxoilcubezflourzcube to maxoilzflour
Changed oilcubezflourzcube to oilzflour
Changed oilcubevaszflourzcube to oilvaszflour
Changed oilcubecliniczflourzcube to oilcliniczflour
Changed maxoilcubecliniczflourzcube to maxoilcliniczflour
Changed zcube to 0
Changed zflourzcube to zflour
Changed oilvaszflourzcube to oilvaszflour
Changed maxoilcubezflourzcube to maxoilzflour
Changed oilzflourzcube to oilzflour
Changed oilzflourfcubefflour to oilzflourfflour
Changed maxoil to oil
Changed maxoilcube to oilcube
Changed maxoilvas to oilvas
Changed maxoilcubevas to oilcubevas
Changed maxoilclinic to oilclinic
Changed maxoilcubeclinic to oilcubeclinic
Changed maxoilzflour to oilzflour
Changed maxoilcubezflour to oilcubezflour
Changed maxoilvaszflour to oilvaszflour
Changed maxoilcliniczflour to oilcliniczflour
Changed maxoilcubevaszflour to oilcubevaszflour
Changed maxoilcubecliniczflour to oilcubecliniczflour
Changed maxoilzflourzcube to oilzflourzcube
Changed maxoilcubezflourzcube to oilcubezflourzcube
Changed maxoilvaszflourzcube to oilvaszflourzcube
Changed maxoilcliniczflourzcube to oilcliniczflourzcube
Changed maxoilcubevaszflourzcube to oilcubevaszflourzcube
Changed maxoilcubecliniczflourzcube to oilcubecliniczflourzcube
Changed zflourfflour33 to 0
Changed fflour33 to 0
Changed cubezflourzcubefcubefflour to zflourfflour
Changed cubezflourfcubefflour to zflourfflour
Changed oilzflourzcubefcubefflour to oilzflourfflour
Changed fcube to 0
Changed fcube to 0
Changed oilfcube to oil
Changed oilcubeclinicfcube to oilclinic
Changed maxoilzflourfcubefflour to maxoilzflourfflour
Changed maxoilcliniczflourfcubefflour to maxoilcliniczflourfflour
Changed maxoilzflourfcubefflour to oilzflourfcubefflour
Changed maxoilcliniczflourfcubefflour to oilcliniczflourfcubefflour
Running lives_saved with None
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True
```

```
[Note]: Optimizing...
[Note]: Optimal Solution Found
Running lives_saved_high with None
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+-----------------------------+-----------------------------+
| MiniMod Solver Results      |                             |
| Method:                     | MIN                         |
| Solver:                     | CBC                         |
| Optimization Status:        | OptimizationStatus.OPTIMAL  |
| Number of Solutions Found:  | 1                           |
+-----------------------------+-----------------------------+
+-----------------------------+--------+
| No. of Variables:           |   6750 |
| No. of Integer Variables:   |   6750 |
| No. of Constraints          |   1159 |
| No. of Non-zeros in Constr. | 231504 |
+-----------------------------+--------+
Interventions Chosen:
+--------------------+-----------------+
| Minimum Benefit    | 22019.1         |
| Objective Bounds   |     1.15113e+07 |
| Total Cost         |     1.15113e+07 |
| Total Lives Saved  | 23220.5         |
+--------------------+-----------------+
+------------------+--------+
| Cost per Benefit | 495.74 |
+------------------+--------+
+-----------------------------------+--+
| Total Cost and Benefits over Time |  |
+-----------------------------------+--+
|   time |   opt_vals |   opt_benefit |        opt_costs |
|-------:|-----------:|--------------:|-----------------:|
|      1 |          3 |          2290 | 939944           |
|      2 |          3 |          2321 | 796726           |
|      3 |          3 |          2351 |      1.17346e+06 |
|      4 |          3 |          2696 |      1.13025e+06 |
|      5 |          3 |          2732 |      1.23289e+06 |
|      6 |          3 |          2770 |      1.22924e+06 |
|      7 |          3 |          2807 |      1.18669e+06 |
|      8 |          3 |          2844 |      1.20595e+06 |
|      9 |          3 |          2884 |      1.28702e+06 |
```

```
|     10 |          3 |          2919 |      1.32918e+06 |

Optimal Interventions

+-----------------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                                   |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|-----------------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('oilzflourfcubefflour', 'Cities') |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('oilzflourfcubefflour', 'North')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('oilzflourfcubefflour', 'South')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
+-----------------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
importing Jupyter notebook from /home/lordflaron/Documents/minimod/optimization_work/demewoz_lives_save
Changed fcube to 0
Changed fflour to fflour33
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

               MiniMod Nutrition Intervention Tool
               Optimization Method: MIN
               Version: 0.0.6dev
               Solver: CBC,
               Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+----------------------------+----------------------------+
| MiniMod Solver Results     |                            |
| Method:                    | MIN                        |
| Solver:                    | CBC                        |
| Optimization Status:       | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                          |
+----------------------------+----------------------------+
+-----------------------------+------+
| No. of Variables:           |  120 |
| No. of Integer Variables:   |  120 |
| No. of Constraints          |  313 |
| No. of Non-zeros in Constr. | 1623 |
+-----------------------------+------+
Interventions Chosen:
+---------------------------------------+-------------+
| Minimum Benefit                       | 1.45004e+07 |
| Objective Bounds                      | 1.55095e+06 |
| Total Cost                            | 1.55095e+06 |
| Total WRA Effectively Covered (Folate) | 1.45004e+07 |
+---------------------------------------+-------------+
+------------------+----------+
| Cost per Benefit | 0.106959 |
+------------------+----------+
+-----------------------------------+--+
| Total Cost and Benefits over Time |  |
+-----------------------------------+--+
|   time |   opt_vals |   opt_benefit |   opt_costs |
|-------:|-----------:|--------------:|------------:|
```

```
|      1 |          3 |   0           |      286412 |
|      2 |          3 |   0           |      134017 |
|      3 |          3 |   1.92897e+06 |      135594 |
|      4 |          3 |   1.98616e+06 |      137190 |
|      5 |          3 |   2.04362e+06 |      138805 |
|      6 |          3 |   2.10248e+06 |      140439 |
|      7 |          3 |   2.16447e+06 |      142093 |
|      8 |          3 |   2.22715e+06 |      143766 |
|      9 |          3 |   2.29016e+06 |      145460 |
|     10 |          3 |   2.35346e+06 |      147172 |

Optimal Interventions

+-------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                         |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|-------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('fflour33', 'Cities') |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('fflour33', 'North')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('fflour33', 'South')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
+-------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
Running with None
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with time
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with space
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
```

```
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with both
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+-----------------------------+-----------------------------+
| MiniMod Solver Results      |                             |
| Method:                     | MIN                         |
| Solver:                     | CBC                         |
| Optimization Status:        | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                           |
+-----------------------------+-----------------------------+
+------------------------------+------+
| No. of Variables:            |  120 |
| No. of Integer Variables:    |  120 |
| No. of Constraints           |  313 |
| No. of Non-zeros in Constr. | 1623 |
+------------------------------+------+
Interventions Chosen:
+----------------------------------------+-------------+
| Minimum Benefit                        | 1.45004e+07 |
| Objective Bounds                       | 1.55095e+06 |
| Total Cost                             | 1.55095e+06 |
| Total WRA Effectively Covered (Folate) | 1.45004e+07 |
+----------------------------------------+-------------+
+------------------+----------+
| Cost per Benefit | 0.106959 |
+------------------+----------+
+----------------------------------+--+
| Total Cost and Benefits over Time |  |
+----------------------------------+--+
|   time |   opt_vals |   opt_benefit |   opt_costs |
|-------:|-----------:|--------------:|------------:|
|      1 |          3 |   0           |      286412 |
|      2 |          3 |   0           |      134017 |
|      3 |          3 |   1.92897e+06 |      135594 |
|      4 |          3 |   1.98616e+06 |      137190 |
|      5 |          3 |   2.04362e+06 |      138805 |
|      6 |          3 |   2.10248e+06 |      140439 |
|      7 |          3 |   2.16447e+06 |      142093 |
|      8 |          3 |   2.22715e+06 |      143766 |
```

```
|      9 |          3 |   2.29016e+06 |      145460 |
|     10 |          3 |   2.35346e+06 |      147172 |

Optimal Interventions

+------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                        |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('fflour33', 'Cities') |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('fflour33', 'North')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('fflour33', 'South')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
+------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
importing Jupyter notebook from /home/lordflaron/Documents/minimod/optimization_work/demewoz_lives_saved
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
Changed zcube to 0
Changed zflourzcube to zflour
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+----------------------------+----------------------------+
| MiniMod Solver Results     |                            |
| Method:                    | MIN                        |
| Solver:                    | CBC                        |
| Optimization Status:       | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                          |
+----------------------------+----------------------------+
+-----------------------------+------+
| No. of Variables:           |   90 |
| No. of Integer Variables:   |   90 |
| No. of Constraints          |  313 |
| No. of Non-zeros in Constr. | 1278 |
+-----------------------------+------+
Interventions Chosen:
+------------------------------------------+-------------+
| Minimum Benefit                          | 7.81179e+06 |
| Objective Bounds                         | 6.06955e+06 |
| Total Cost                               | 6.06955e+06 |
| Total Children Effectively Covered (Zinc) | 7.81179e+06 |
+------------------------------------------+-------------+
+------------------+----------+
| Cost per Benefit | 0.776973 |
+------------------+----------+
+-----------------------------------+--+
| Total Cost and Benefits over Time |  |
```

```
+-----------------------------------+--+
|   time |   opt_vals |   opt_benefit |   opt_costs |
|-------:|-----------:|--------------:|------------:|
|      1 |          3 |        833571 |      494593 |
|      2 |          3 |        846033 |      343359 |
|      3 |          3 |        858734 |      641157 |
|      4 |          3 |        872058 |      588019 |
|      5 |          3 |        885833 |      680607 |
|      6 |          3 |        899011 |      666795 |
|      7 |          3 |        912323 |      613962 |
|      8 |          3 |        925599 |      622817 |
|      9 |          3 |        938577 |      693359 |
|     10 |          3 |        951450 |      724881 |

Optimal Interventions

+----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                      |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('zflour', 'Cities') |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('zflour', 'North')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('zflour', 'South')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
+----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
Running with None
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with time
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with space
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
```

```
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with both
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+----------------------------+----------------------------+
| MiniMod Solver Results     |                            |
| Method:                    | MIN                        |
| Solver:                    | CBC                        |
| Optimization Status:       | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                          |
+----------------------------+----------------------------+
+-----------------------------+------+
| No. of Variables:           |   90 |
| No. of Integer Variables:   |   90 |
| No. of Constraints          |  313 |
| No. of Non-zeros in Constr. | 1278 |
+-----------------------------+------+
Interventions Chosen:
+------------------------------------------+-------------+
| Minimum Benefit                          | 7.81179e+06 |
| Objective Bounds                         | 6.06955e+06 |
| Total Cost                               | 6.06955e+06 |
| Total Children Effectively Covered (Zinc) | 7.81179e+06 |
+------------------------------------------+-------------+
+-----------------+----------+
| Cost per Benefit | 0.776973 |
+-----------------+----------+
+----------------------------------+--+
| Total Cost and Benefits over Time |  |
+----------------------------------+--+
|   time |   opt_vals |   opt_benefit |   opt_costs |
|-------:|-----------:|--------------:|------------:|
|      1 |          3 |        833571 |      494593 |
|      2 |          3 |        846033 |      343359 |
|      3 |          3 |        858734 |      641157 |
|      4 |          3 |        872058 |      588019 |
|      5 |          3 |        885833 |      680607 |
```

```
|      6 |          3 |        899011 |      666795 |
|      7 |          3 |        912323 |      613962 |
|      8 |          3 |        925599 |      622817 |
|      9 |          3 |        938577 |      693359 |
|     10 |          3 |        951450 |      724881 |

Optimal Interventions

+----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                      |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('zflour', 'Cities') |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('zflour', 'North')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('zflour', 'South')  |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
+----------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
importing Jupyter notebook from /home/lordflaron/Documents/minimod/optimization_work/demewoz_lives_save
Changed cube to 0
Changed maxoil to oil
Changed maxoilcube to maxoil
Changed oilcube to oil
Changed oilcubevas to oilvas
Changed maxoilcubevas to maxoilvas
Changed cubevas to vas
Changed cubeclinic to clinic
Changed maxoilcubeclinic to maxoilclinic
Changed oilcubeclinic to oilclinic
Changed maxoilcube to oilcube
Changed maxoilvas to oilvas
Changed maxoilcubevas to oilcubevas
Changed maxoilclinic to oilclinic
Changed maxoilcubeclinic to oilcubeclinic
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+----------------------------+----------------------------+
| MiniMod Solver Results     |                            |
| Method:                    | MIN                        |
| Solver:                    | CBC                        |
| Optimization Status:       | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                          |
+----------------------------+----------------------------+
+----------------------------+------+
| No. of Variables:          |  510 |
| No. of Integer Variables:  |  510 |
| No. of Constraints         |  454 |
```

```
| No. of Non-zeros in Constr. | 6933 |
+-----------------------------+------+
Interventions Chosen:
+------------------------------------------+-------------+
| Minimum Benefit                          | 1.24652e+07 |
| Objective Bounds                         | 2.83901e+07 |
| Total Cost                               | 2.83901e+07 |
| Total Children Effectively Covered (VA)  | 1.2496e+07  |
+------------------------------------------+-------------+
+------------------+---------+
| Cost per Benefit | 2.27193 |
+------------------+---------+
+------------------------------------+--+
| Total Cost and Benefits over Time |  |
+------------------------------------+--+
|   time |   opt_vals |   opt_benefit |   opt_costs |
|-------:|-----------:|--------------:|------------:|
|      1 |          3 |   1.345e+06   | 3.56153e+06 |
|      2 |          3 |   1.35927e+06 | 3.52776e+06 |
|      3 |          3 |   1.05806e+06 | 1.72146e+06 |
|      4 |          3 |   1.4436e+06  | 2.66859e+06 |
|      5 |          3 |   1.46313e+06 | 2.70103e+06 |
|      6 |          3 |   1.48317e+06 | 2.79537e+06 |
|      7 |          3 |   1.50492e+06 | 2.76685e+06 |
|      8 |          3 |   1.52759e+06 | 2.88426e+06 |
|      9 |          3 |   1.55068e+06 | 2.89545e+06 |
|     10 |          3 |   1.57489e+06 | 2.8678e+06  |

Optimal Interventions

+---------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
|                           |   1 |   2 |   3 |   4 |   5 |   6 |   7 |   8 |   9 |   10 |
|---------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------|
| ('oilcube', 'Cities')     |   0 |   0 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('oilcube', 'South')      |   0 |   0 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('oilcubevas', 'Cities')  |   1 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |    0 |
| ('oilcubevas', 'North')   |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |   1 |    1 |
| ('oilcubevas', 'South')   |   1 |   1 |   0 |   0 |   0 |   0 |   0 |   0 |   0 |    0 |
+---------------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+------+
Running with None
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with time
[Note]: Processing Data...
```

```
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with space
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
Running with both
[Note]: Processing Data...
[Note]: Creating Base Model with constraints

                MiniMod Nutrition Intervention Tool
                Optimization Method: MIN
                Version: 0.0.6dev
                Solver: CBC,
                Show Output: True


[Note]: Optimizing...
[Note]: Optimal Solution Found
+----------------------------+----------------------------+
| MiniMod Solver Results     |                            |
| Method:                    | MIN                        |
| Solver:                    | CBC                        |
| Optimization Status:       | OptimizationStatus.OPTIMAL |
| Number of Solutions Found: | 1                          |
+----------------------------+----------------------------+
+-----------------------------+------+
| No. of Variables:           |  510 |
| No. of Integer Variables:   |  510 |
| No. of Constraints          |  454 |
| No. of Non-zeros in Constr. | 6933 |
+-----------------------------+------+
Interventions Chosen:
+-----------------------------------------+-------------+
| Minimum Benefit                         | 1.24652e+07 |
| Objective Bounds                        | 2.83901e+07 |
```

Total Cost	2.83901e+07
Total Children Effectively Covered (VA)	1.2496e+07

Cost per Benefit	2.27193

Total Cost and Benefits over Time	

time	opt_vals	opt_benefit	opt_costs
1	3	1.345e+06	3.56153e+06
2	3	1.35927e+06	3.52776e+06
3	3	1.05806e+06	1.72146e+06
4	3	1.4436e+06	2.66859e+06
5	3	1.46313e+06	2.70103e+06
6	3	1.48317e+06	2.79537e+06
7	3	1.50492e+06	2.76685e+06
8	3	1.52759e+06	2.88426e+06
9	3	1.55068e+06	2.89545e+06
10	3	1.57489e+06	2.8678e+06

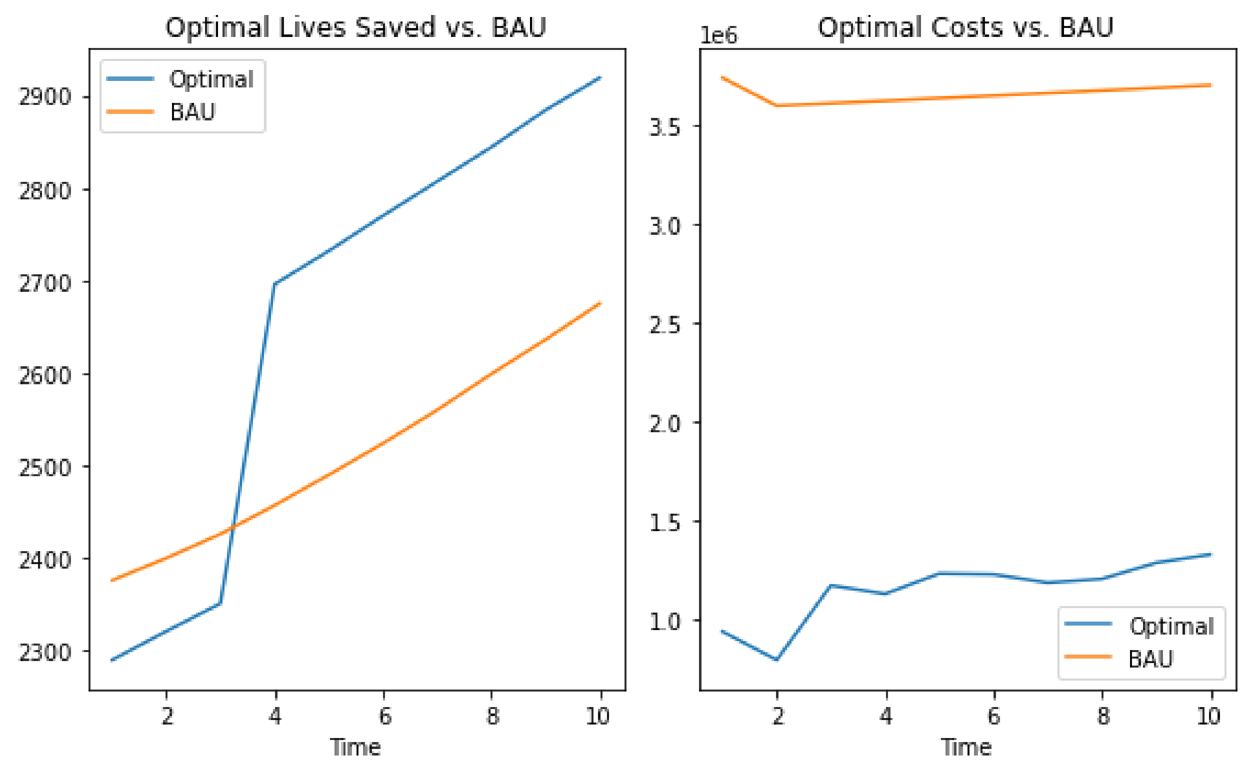
Optimal Interventions

	1	2	3	4	5	6	7	8	9	10
('oilcube', 'Cities')	0	0	1	1	1	1	1	1	1	1
('oilcube', 'South')	0	0	1	1	1	1	1	1	1	1
('oilcubevas', 'Cities')	1	1	0	0	0	0	0	0	0	0
('oilcubevas', 'North')	1	1	1	1	1	1	1	1	1	1
('oilcubevas', 'South')	1	1	0	0	0	0	0	0	0	0

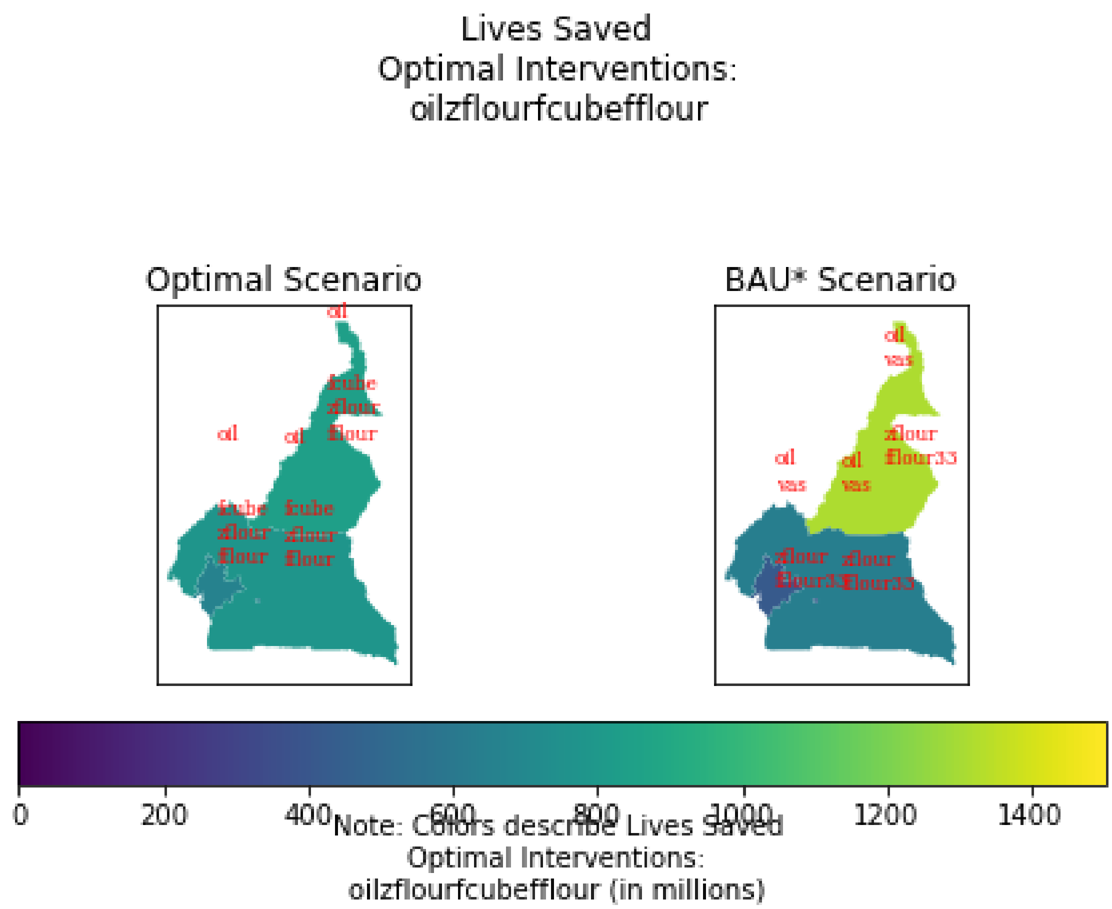
```
/usr/lib/python3.8/site-packages/geopandas/_compat.py:84: UserWarning: The Shapely GEOS version (3.8.0-
  warnings.warn(
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/usr/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3417: PerformanceWarning: indexing pa
  exec(code_obj, self.user_global_ns, self.user_ns)
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/usr/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3417: PerformanceWarning: indexing pa
  exec(code_obj, self.user_global_ns, self.user_ns)
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
```

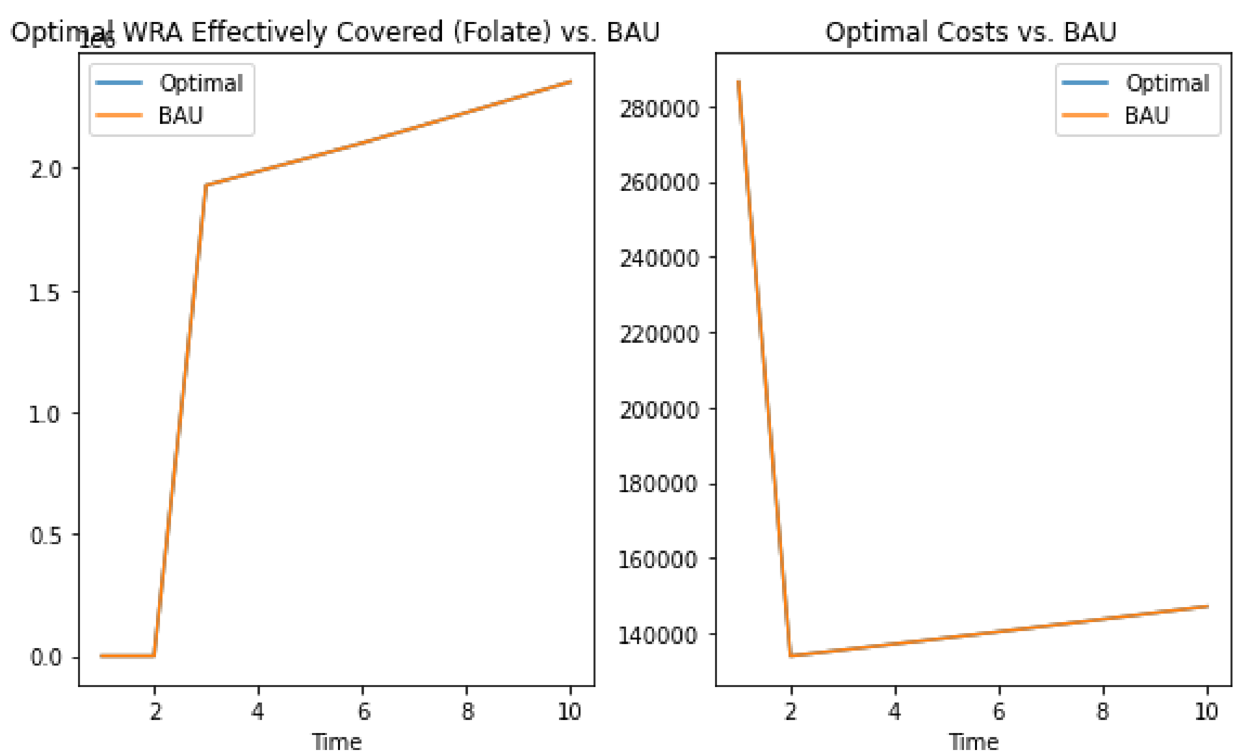
```
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/usr/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3417: PerformanceWarning: indexing pas
  exec(code_obj, self.user_global_ns, self.user_ns)
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/usr/lib/python3.8/site-packages/ipywidgets/widgets/interaction.py in update(self, *args)
    254                     value = widget.get_interact_value()
    255                     self.kwargs[widget._kwarg] = value
--> 256                 self.result = self.f(**self.kwargs)
    257                 show_inline_matplotlib_plots()
    258                 if self.auto_display and self.result is not None:

~/Documents/minimod/optimization_work/demewoz_lives_saved/vas_effective_coverage_analysis.ipynb in plot_

TypeError: plot_map_benchmark() got an unexpected keyword argument 'bau_intervention_bubbles'
```

```
interactive(children=(Dropdown(description='col', options=('lives_saved', 'lives_saved_high'), value='li

interactive(children=(Dropdown(description='col', options=('lives_saved', 'lives_saved_high'), value='li
```

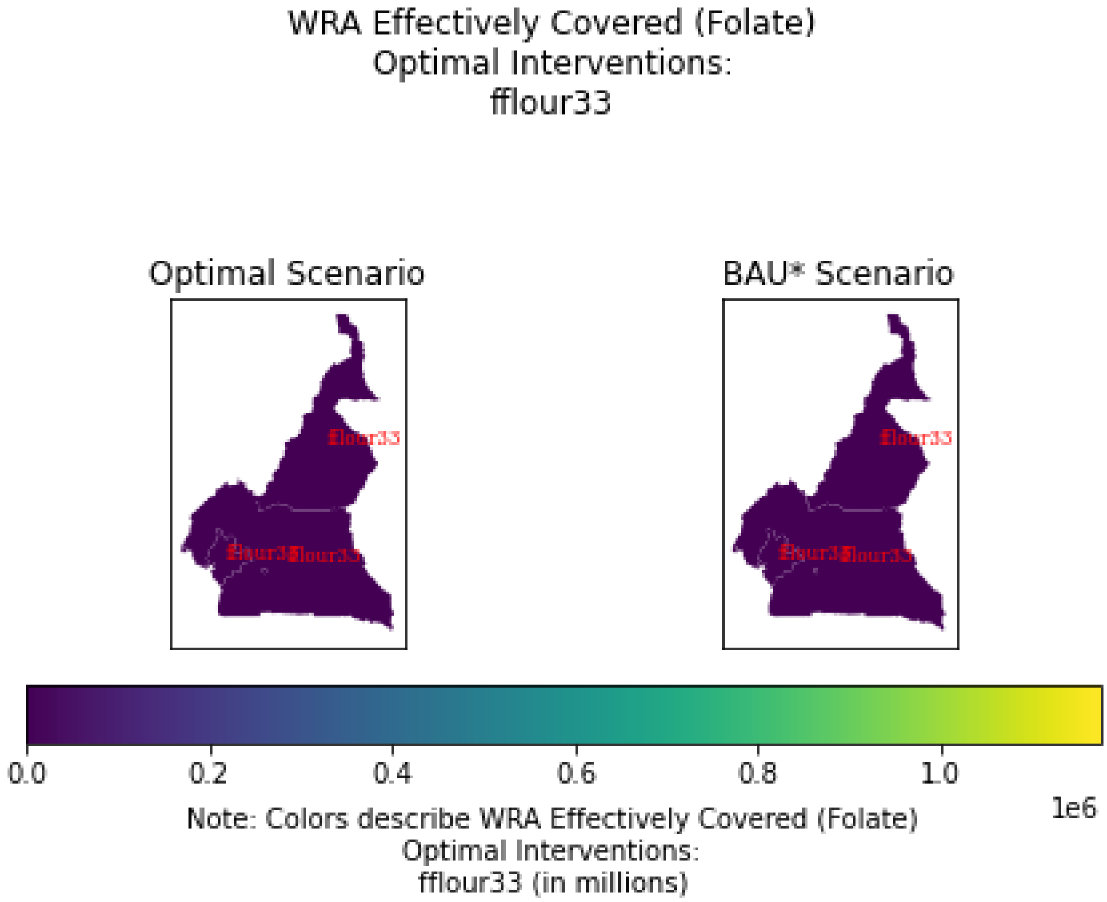



```
interactive(children=(Dropdown(description='col', options=('lives_saved', 'lives_saved_high'), value='li
```

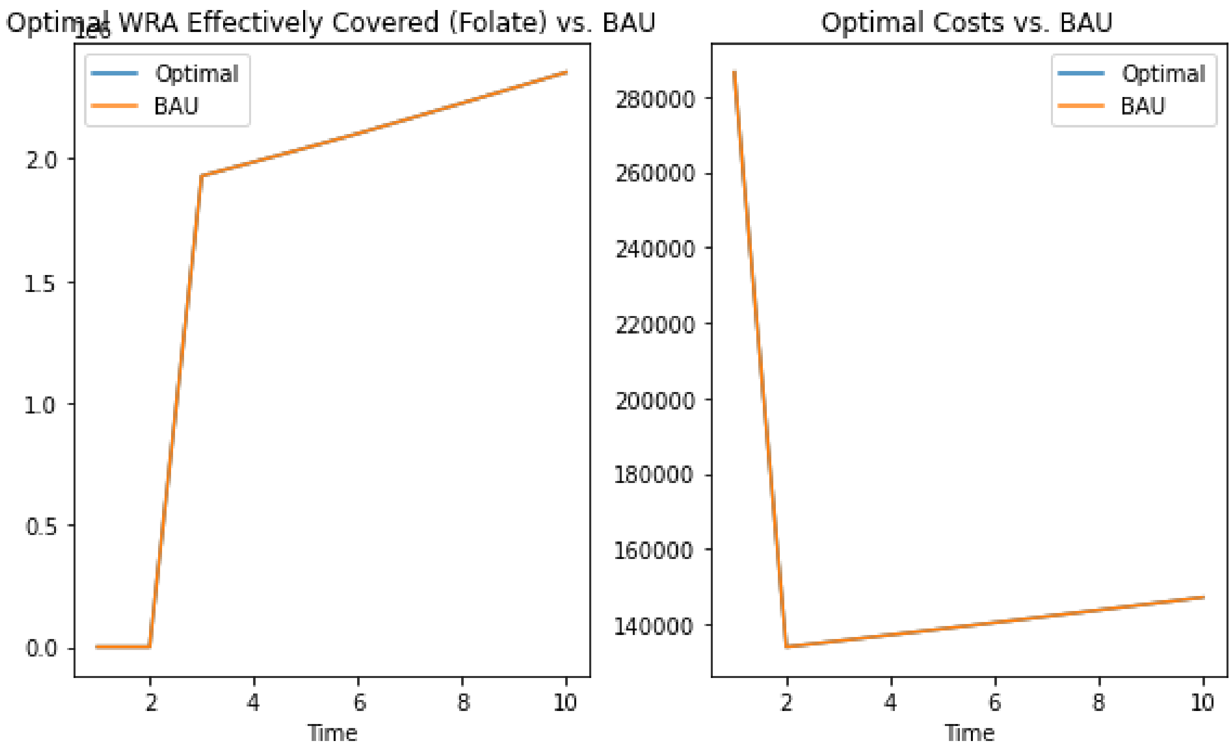

Lives Saved
Optimal Interventions:
oilzflourfcubefflour
Optimal Scenario
BAU* Scenario
oil
fcube
zflour
fflour
vas
fflour33
0
200
400
600
800
1000
1200
1400
Note: Colors describe Lives Saved
Optimal Interventions:
oilzflourfcubefflour (in millions)


Optimal WRA Effectively Covered (Folate) vs. BAU
1e6
Optimal
BAU
2.0
1.5
1.0
0.5
0.0
2
4
6
8
10
Time
Optimal Costs vs. BAU
Optimal
BAU
280000
260000
240000
220000
200000
180000
160000
140000
2
4
6
8
10
Time

```
interactive(children=(IntSlider(value=1, description='time', max=10, min=1), Dropdown(description='opti
```

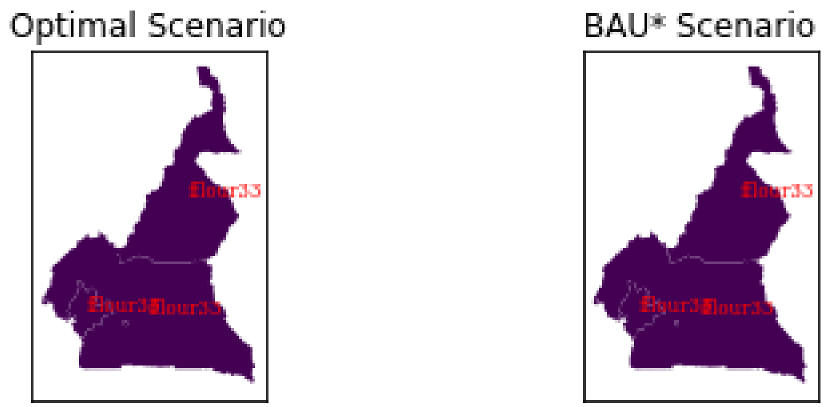


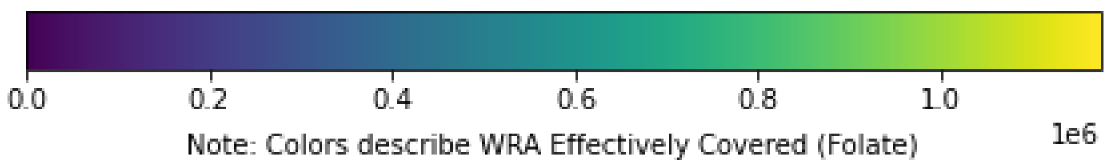

```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol

interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol
```



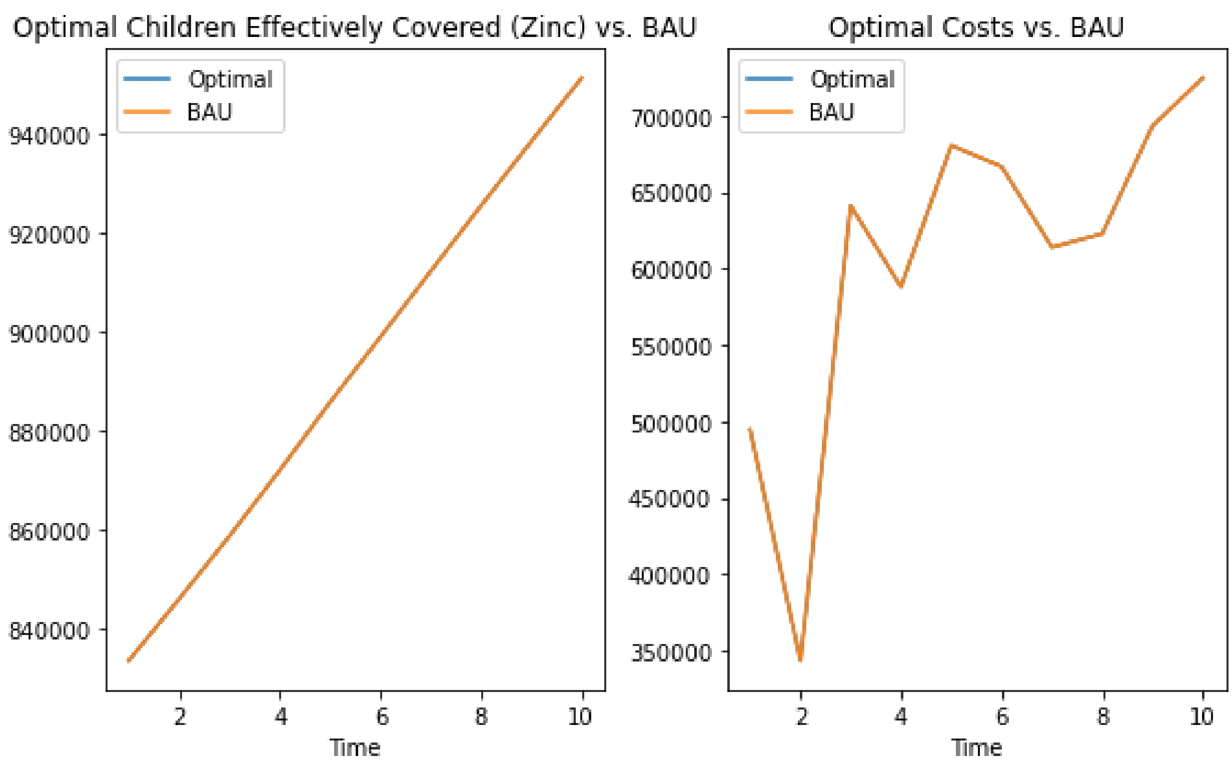

```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver o
```

WRA Effectively Covered (Folate)
Optimal Interventions:
fflour33

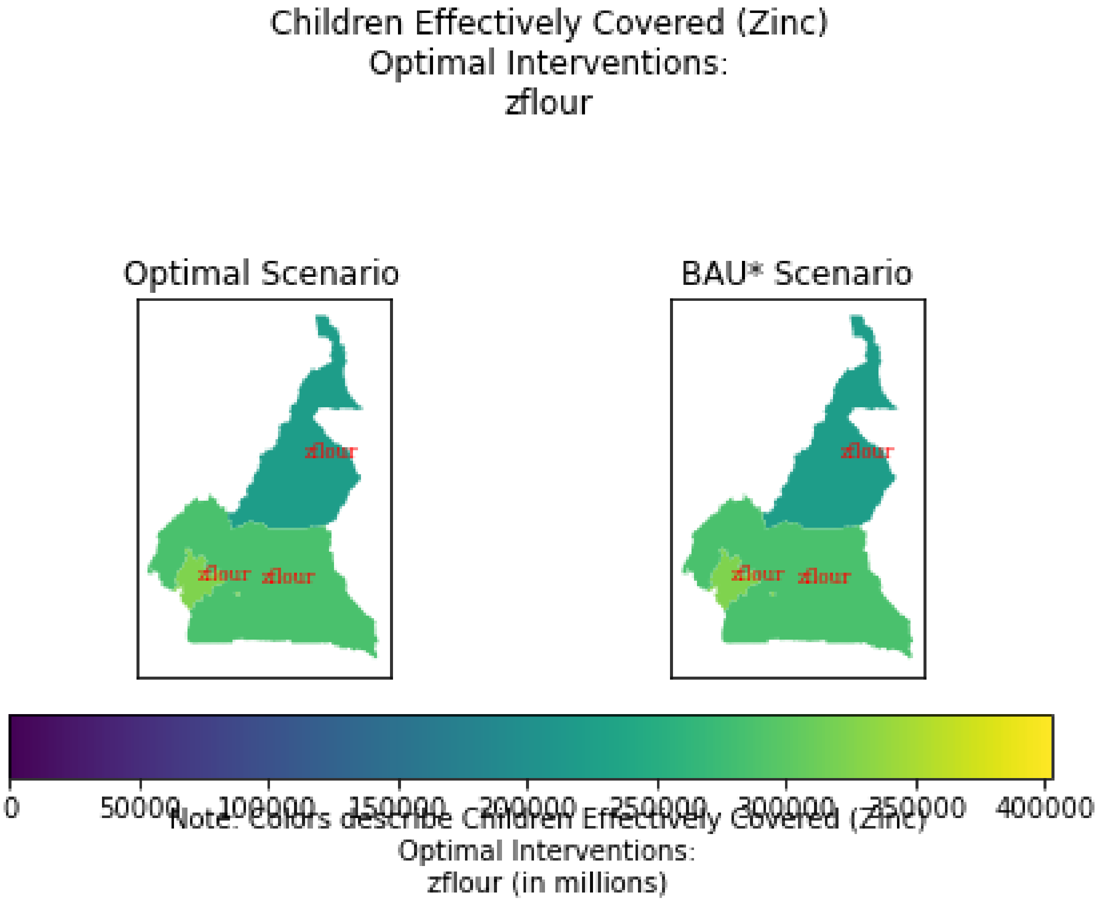




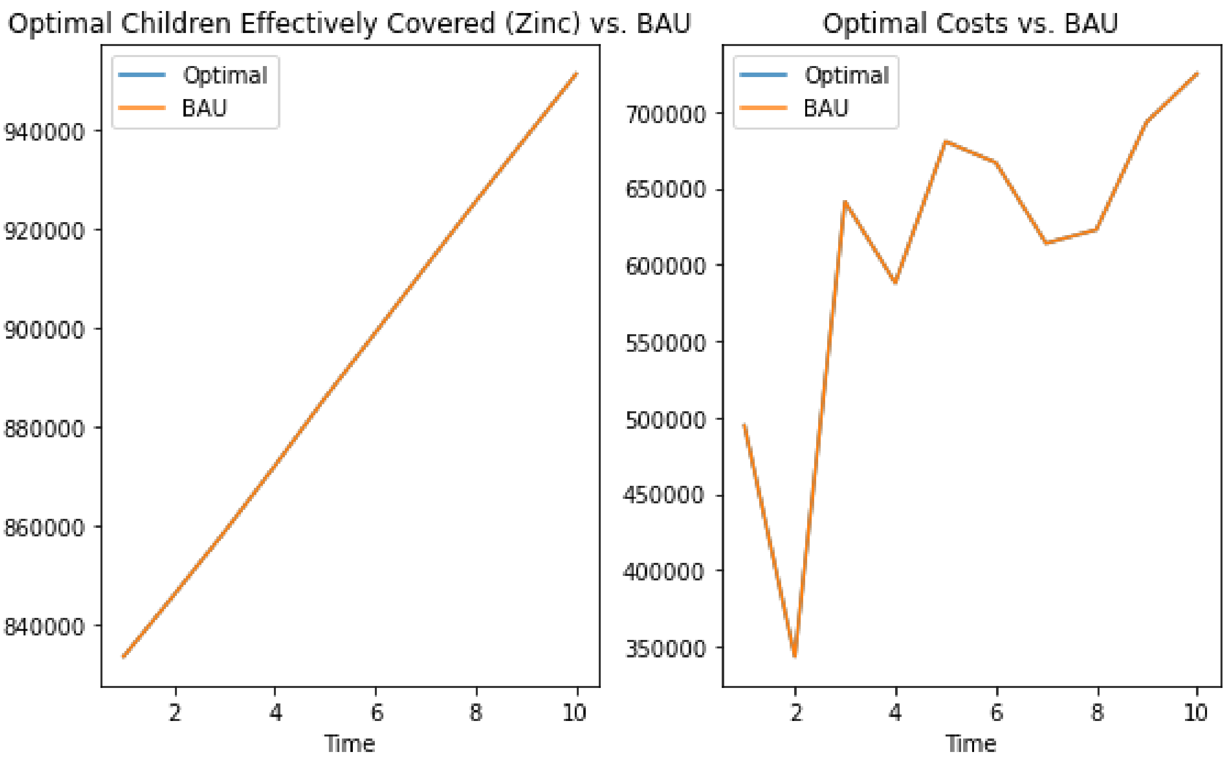
Note: Colors describe WRA Effectively Covered (Folate)
Optimal Interventions:
fflour33 (in millions)



```
interactive(children=(IntSlider(value=1, description='time', max=10, min=1), Dropdown(description='opti
```

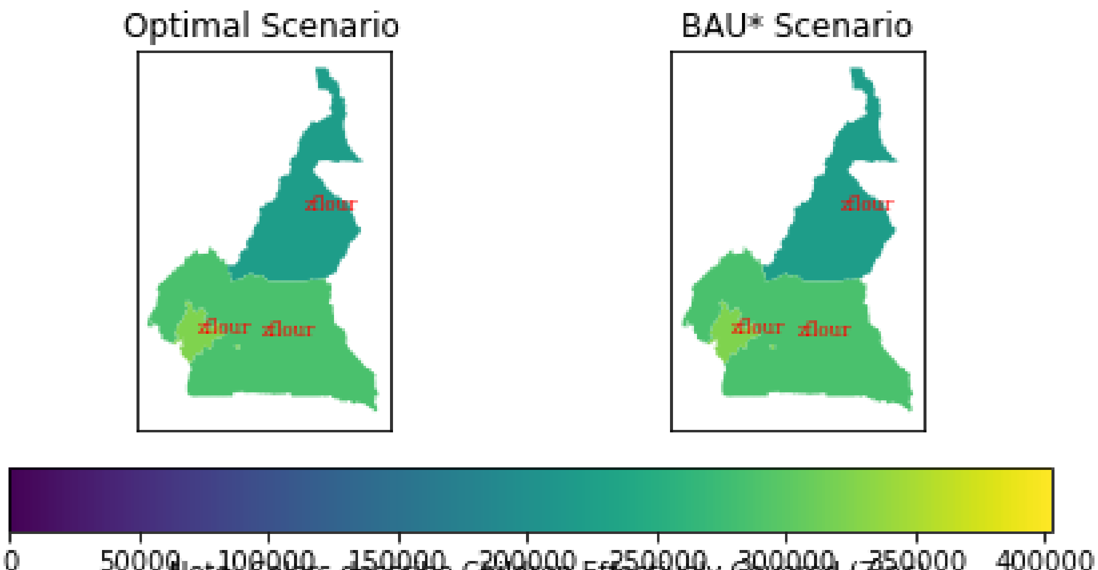



```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol

interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol
```

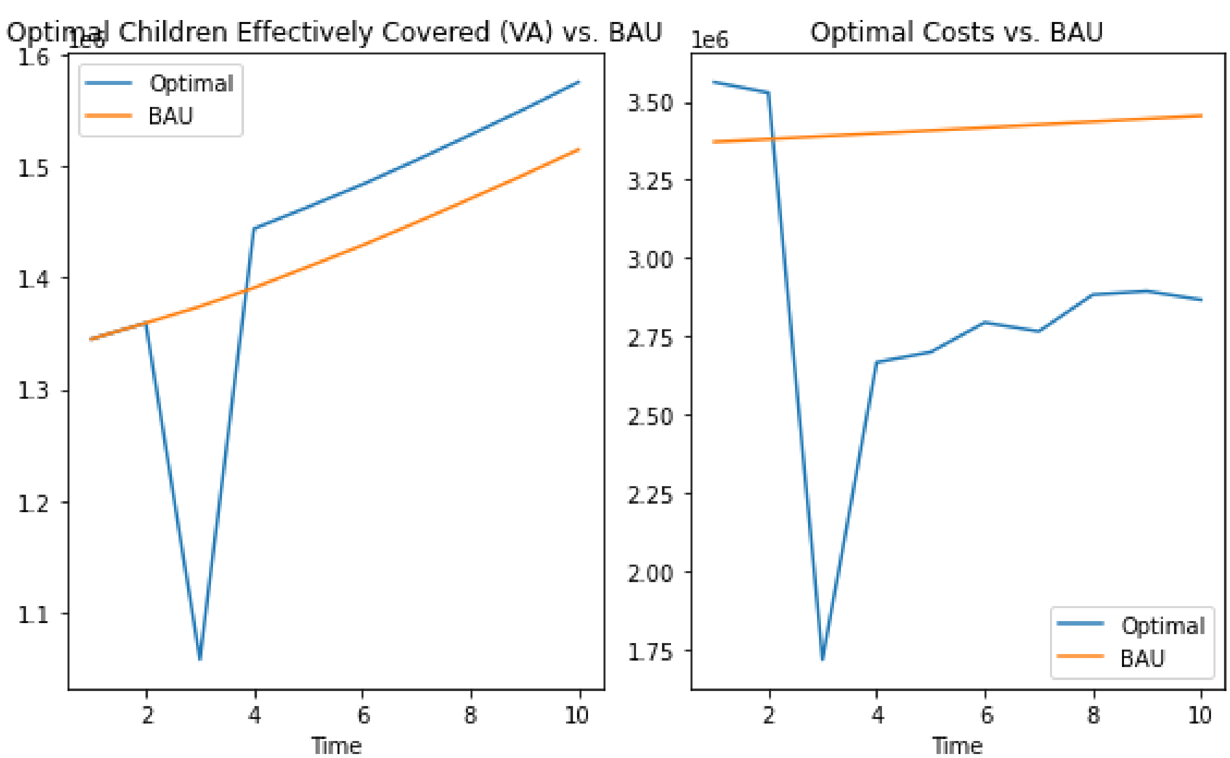



```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol
```

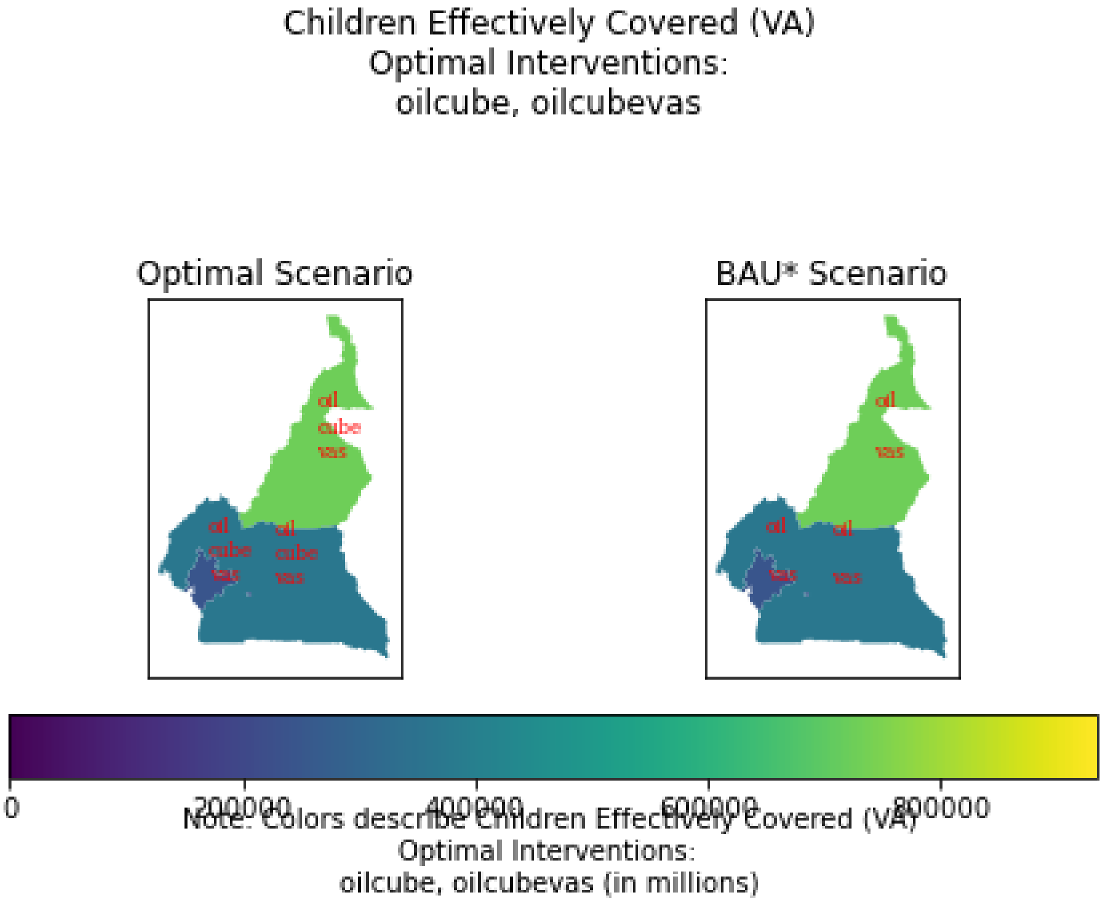
Children Effectively Covered (Zinc)
Optimal Interventions:
zflour



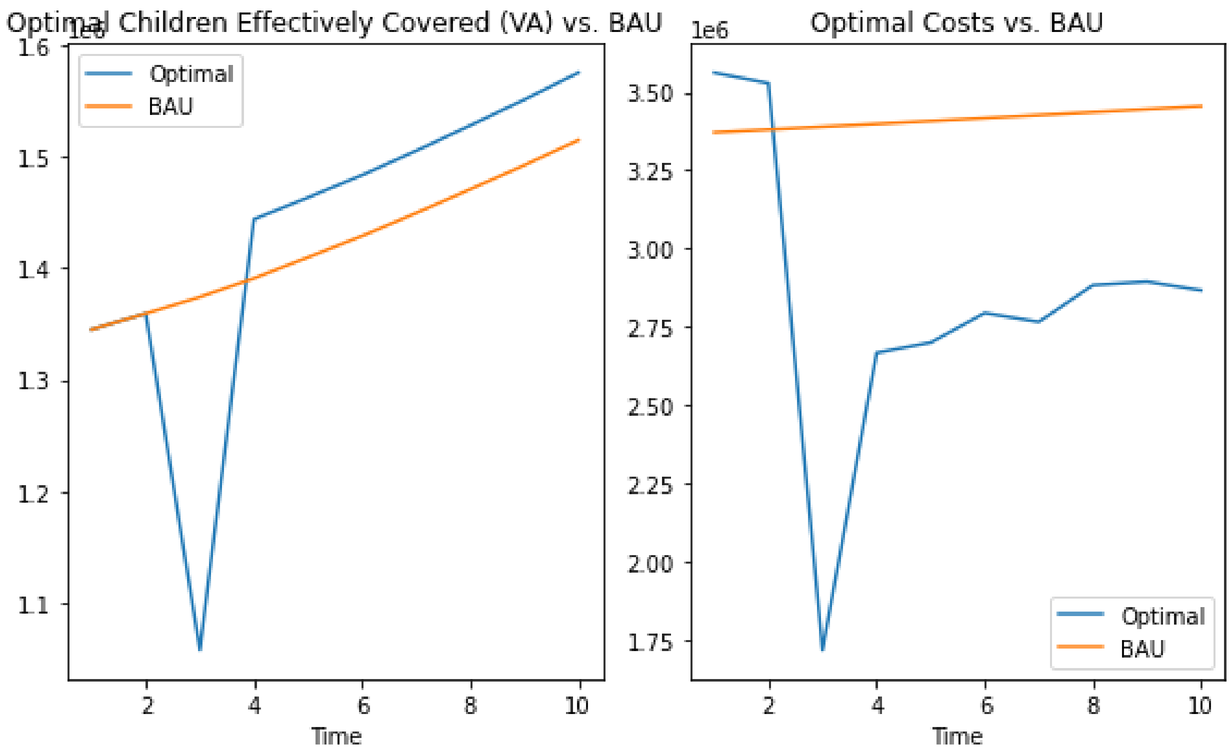
Note: Colors describe Children Effectively Covered (Zinc)
Optimal Interventions:
zflour (in millions)



```
interactive(children=(IntSlider(value=1, description='time', max=10, min=1), Dropdown(description='opti
```




```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol

interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver ol
```




```
interactive(children=(Dropdown(description='m', options={None: <minimod.solvers.costsolver.CostSolver o
```

Table 1: Optimally Chosen Interventions vs. BAU*. {#tbl:df_int_opt}

	BAU*	Optimal Intervention Chosen	Interventions Not Chosen
Zinc (Children Eff. Cov.)	Zinc Flour (95 mg/kg)	Zinc Flour (95 mg/kg)	Zinc Cube 600 mg/kg
VA (Children Eff. Cov.)	VA Oil 9 mg/kg + VAS-CHD	VA Oil 12 mg/kg + VAS-CHD (partial) + VA Cube (80 mg/kg)	VA Oil 12 mg/kg, VAS Routine
Folic Acid (WRA Eff. Cov.)	Folic Acid Flour 1.65 mg/kg	Folic Acid Flour 1.65 mg/kg	Folic Acid Cube 100 mg/kg
All (Lives Saved)	VA Oil 9 mg/kg + VAS-CHD + Zinc Flour 95 mg/kg + Folic Acid Flour 1.65 mg/kg	VA Oil 9 mg/kg + Zinc Flour 95 mg/kg + Folic Acid Flour 5 mg/kg + Folic Acid Cube 100 mg/kg	Zinc Cube 600 mg/kg, VA Oil 12 mg/kg, VAS Routine, Folic Acid Cube 100 mg/kg
All (Lives Saved, Alt.)	VA Oil 9 mg/kg + VAS-CHD + Zinc Flour 95 mg/kg + Folic Acid Flour 1.65 mg/kg	VA Oil 9 mg/kg + VAS-CHD (partial) + Zinc Flour (95 mg/kg) + Folic Acid Cube (100 mg/kg) + Folic Acid Flour (5 mg/kg)	Zinc Cube 600 mg/kg, VA Oil 12 mg/kg, VAS Routine, Folic Acid Cube 100 mg/kg

@tbl:df_int_opt shows the set of optimal interventions chosen for each micronutrient simulation as well as their Business as Usual Scenario (BAU).

The BAU* was chosen for each simulation so that it would be consistent across effective coverage and lives

Table 2: Total Benefits and Costs Across Space

		% North	% South	% Cities	National
Zinc (Children Eff. Cov.)	CB	0.26	0.33	0.41	8,923,189.41
	CC	0.24	0.37	0.39	6,069,548.82
VA (Children Eff. Cov.)	CB	0.64	0.20	0.16	13,069,585.95
	CC	0.64	0.20	0.16	23,736,675.25
Folic Acid (WRA Eff. Cov.)	CB	0.18	0.33	0.49	17,096,476.30
	CC	0.24	0.37	0.39	1,550,946.68
All (Lives Saved)	CB	0.37	0.35	0.28	26,614.00
	CC	0.27	0.33	0.40	11,511,343.00
All (Lives Saved, Alt.)	CB	0.45	0.30	0.25	32,849.00
	CC	0.42	0.26	0.31	14,577,066.00

Note: TB= Total Benefits and TC= Total Costs

saved simulations. In the case of the micronutrient effective coverage simulations, the optimal solution is the BAU* scenario for zinc and folic acid. For vitamin A, in contrast to the BAU* scenario, fortified boullion cube is also chosen.

When we consider all interventions together, however, the optimal choice becomes different than the BAU* (which is just a composite of the micronutrient scenarios' BAU).[1] *In contrast to the BAU*, the optimally chosen set of interventions may not include VAS campaign, and may include an addition of folic acid fortified boullion cube. Folic acid fortified flour is included at 1.65 mg/kg, not 5.0 mg/kg.

Note that, although some of the interventions chosen optimally are the same as in the BAU*, the timing and spatial distribution of each intervention may not be the same. Since MINIMOD chooses the optimal set of interventions across space and time, it may be that the appearance of an intervention may happen earlier or later in time than others and may only occur in certain parts of the country.

The BAU* scenarios all assume a constant set of interventions across space and time.

We can also see the differences in how each region is affected after 10 years in terms of accumulated benefits and costs.

*@tbl:opt_space shows the accumulated benefits and costs for each simulation across space. Note that the units of the benefits are dependent on the simulation.

We can also see how benefits accumulate over time, in +@tbl:opt_time.

To illustrate how different these outcomes are to the BAU* scenarios, +@fig:bau_comp shows the difference in cost-per child on a bar graph for each simulation, side by side.

```
import matplotlib.pyplot as plt

cpb = (
    df_all
    .loc[(slice(None), 10), :]
    .sum()
    .unstack()
    .assign(cost_per_benefit = lambda df: df['CC']/df['CB'])
    [['BAU* Cost per Benefit', 'cost_per_benefit']]
    .rename({'cost_per_benefit' : 'Optimal Cost per Benefit'}, axis=1)
    )

```

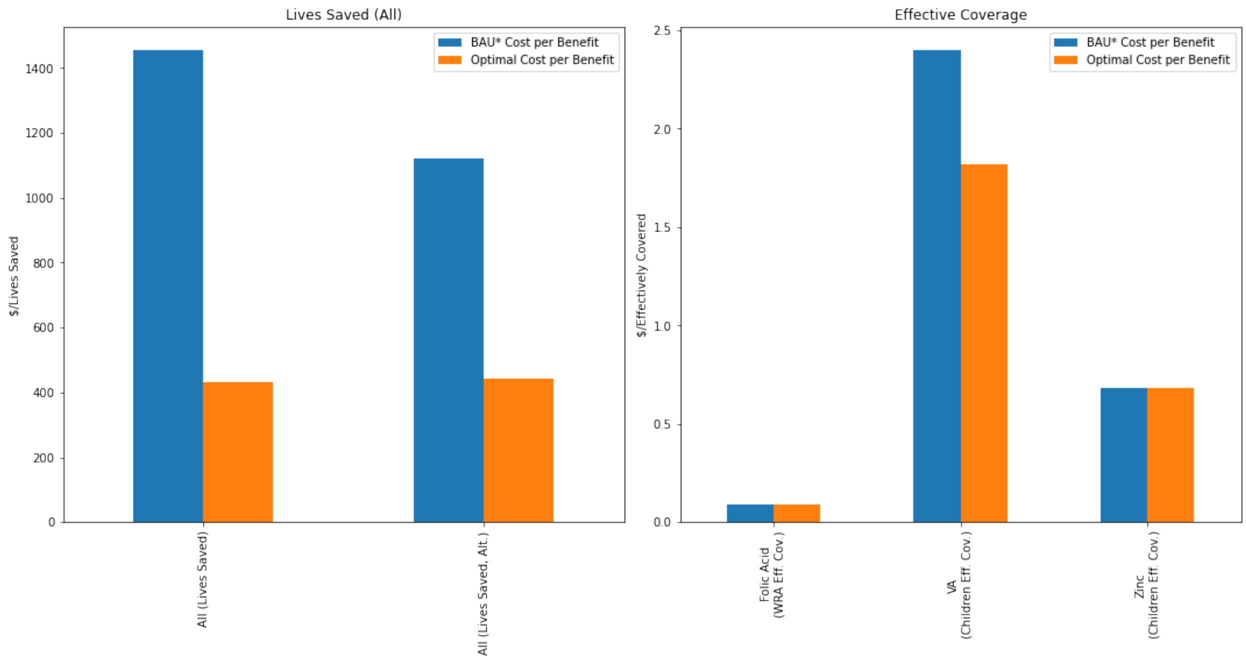
[1] Notice that there are two alternative definitions for the lives saved estimates of each nutritional intervention. The resulting optimally chosen interventions are similar, but the alternative specification includes VAS campaign as well.

Table 3: Optimal Cumulative Benefits over Time

Time	Zinc (Children EC) CB	VA (Children EC) CB	Folic Acid (WRA EC) CB	All (LS) CB	All (LS, Alt.) CB
1	833,571	6,011,853	0	2,290	3,741
2	1,679,604	7,371,118	0	4,611	7,524
3	2,538,338	2,521,598	1,928,972	6,962	11,350
4	3,410,396	3,965,198	3,915,128	9,658	14,299
5	4,296,229	5,428,330	5,958,749	12,390	17,287
6	5,195,240	6,911,502	8,061,227	15,160	20,317
7	6,107,563	8,416,423	10,225,700	17,967	23,387
8	7,033,162	9,944,016	12,452,854	20,811	26,498
9	7,971,739	11,494,698	14,743,015	23,695	29,653
10	8,923,189	13,069,586	17,096,476	26,614	32,849

Note: CB= Cumulative Benefits over a 10 year period, EC= Effective Coverage and LS=Lives Saved.

```
fig, (ax1,ax2) = plt.subplots(1,2, figsize= (15,8))

all_ls = (
    cpb
    .loc[cpb.index.str.contains('All')]
    .assign(perc_change= lambda df: (df['BAU* Cost per Benefit'] - df['Optimal Cost per Benefit'])/df['
    )
all_ls[['BAU* Cost per Benefit', 'Optimal Cost per Benefit']].plot.bar(ax=ax1)
eff_cov = cpb.loc[~cpb.index.str.contains('All')].assign(perc_change= lambda df: (df['BAU* Cost per Bene
eff_cov[['BAU* Cost per Benefit', 'Optimal Cost per Benefit']].plot.bar(ax=ax2)

ax1.set_title("Lives Saved (All)")
ax2.set_title("Effective Coverage")

ax1.set_ylabel("$/Lives Saved")
ax2.set_ylabel("$/Effectively Covered")

ax1.set_xticklabels(ax1.get_xticklabels(), rotation = 90)
ax2.set_xticklabels(ax2.get_xticklabels(), rotation = 90)

fig.text(.5,-.1, "Note: EC= Effective Coverage", ha='center')


plt.tight_layout()

plt.savefig("optimization_work/demewoz_lives_saved/reports/multi_mn_plus_lives_saved/bauvsopt.png", dpi
```



Note: EC= Effective Coverage

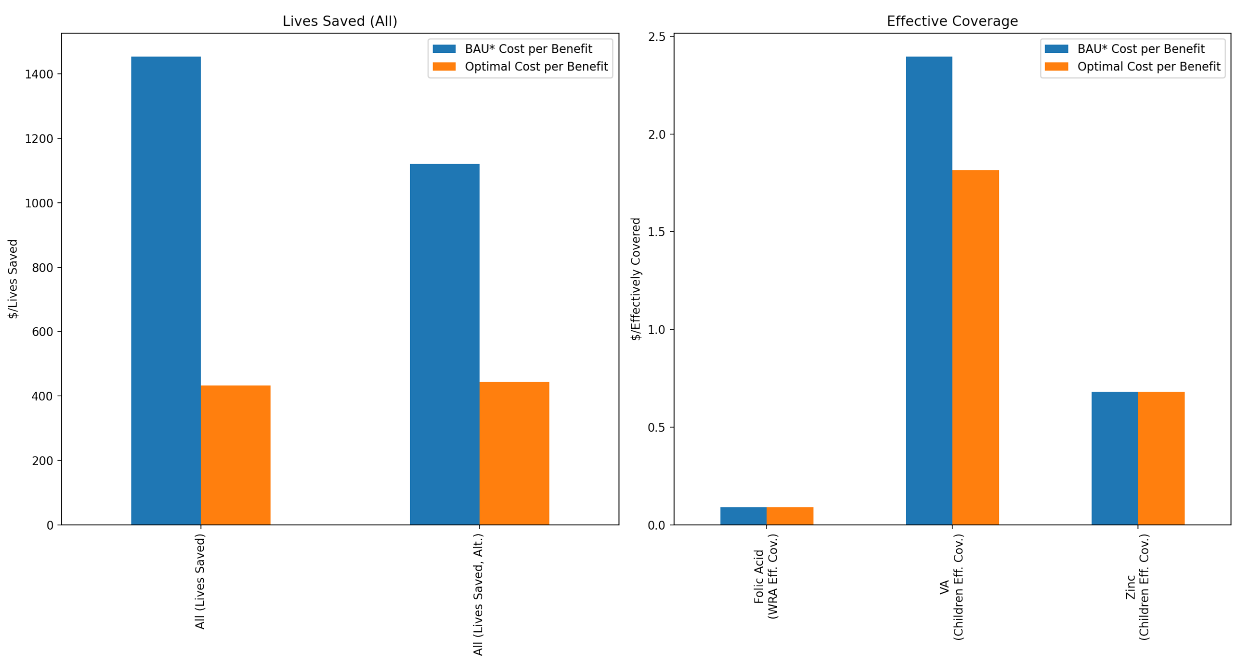


Figure 1: Cost per Benefit Comparison of BAU* vs. Optimal

In the following sections, we will look at relevant maps that will show us how accumulated benefits evolve through time. For these sections we will use shorthand for interventions, outlined in the following table:

Intervention	Abbreviation
Zinc Flour (95 mg/kg)	zflour
VA Oil 9mg/kg	oil
VAS-CHD	vas
VA Cube (80 mg/kg)	cube
Folic Acid Cube (100 mg/kg)	fcube
Folic Acid Flour (5 mg/kg)	fflour
Folic Acid Flour (1.65 mg/kg)	fflour33

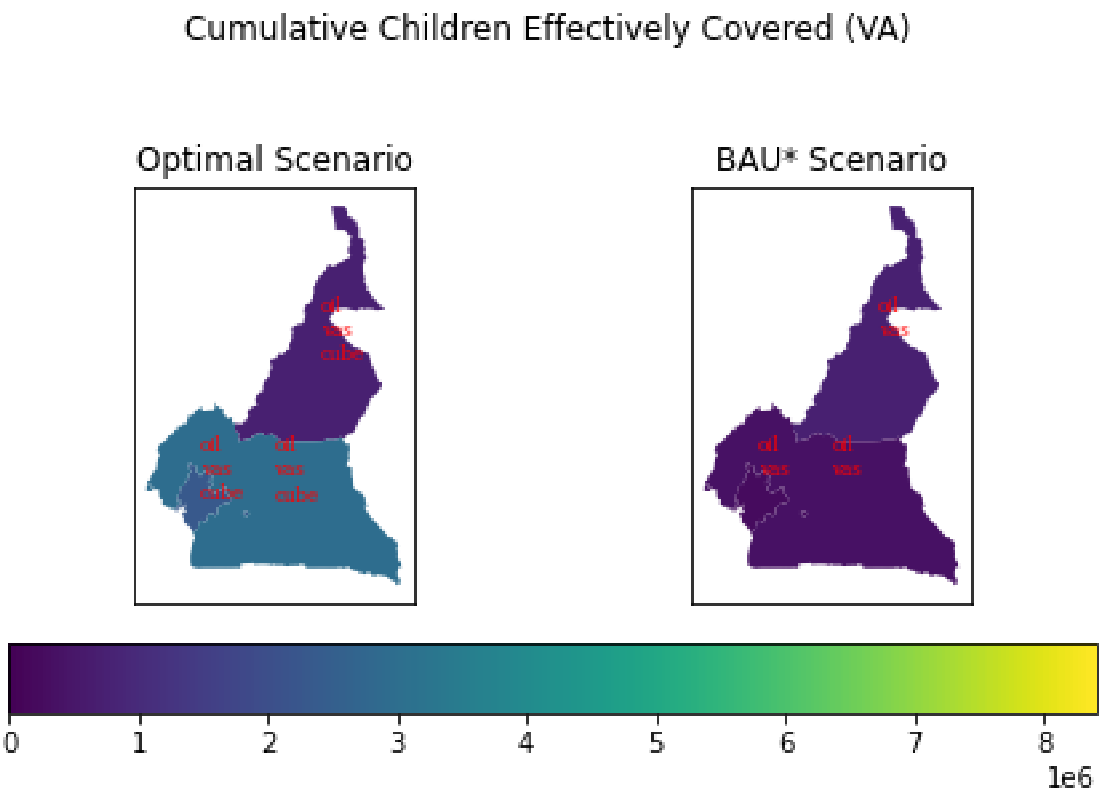
Effective Coverage Simulations

Vitamin A

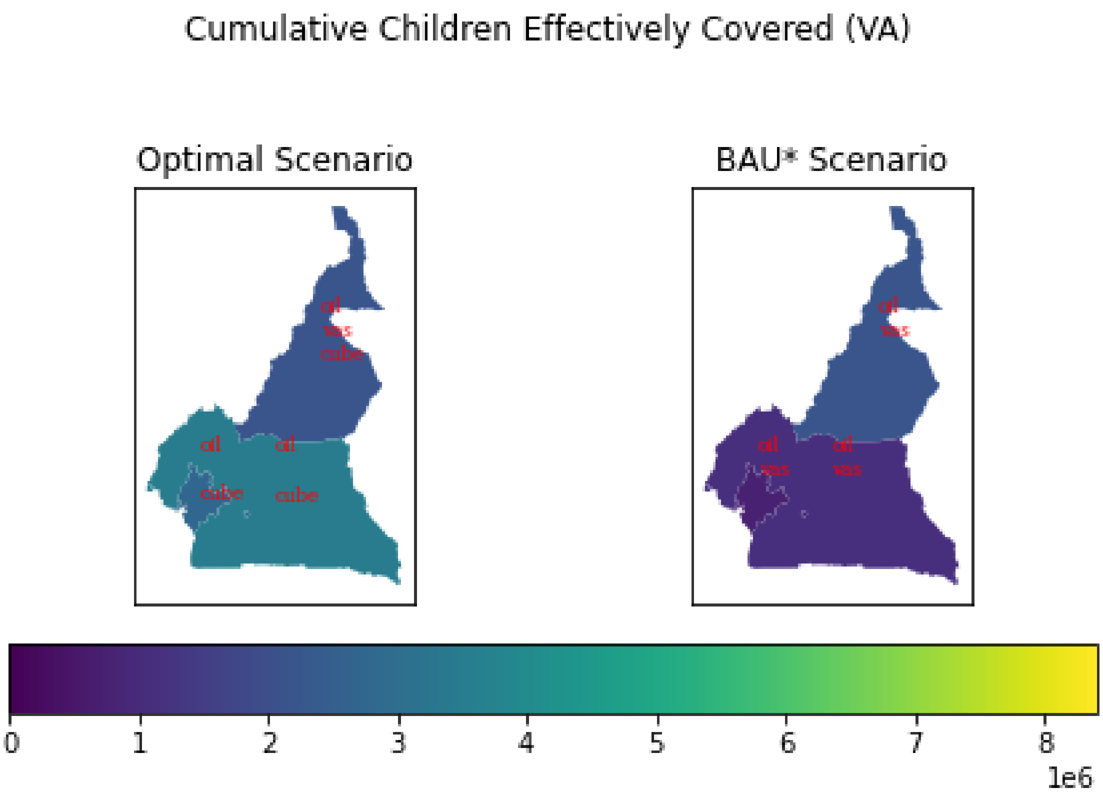
```python
# Load data
geo_df = gpd.read_file("examples/data/maps/cameroon/CAM.shp")

# Now we create the boundaries for North, South and Cities
# Based on "Measuring Costs of Vitamin A..., Table 2"
north = r"Adamaoua|Nord|Extreme-Nord"
south = r"Centre|Est|Nord-Ouest|Ouest|Sud|Sud-Ouest"
cities= r"Littoral" # Duala
# Yaounde is in Mfoundi
geo_df.loc[lambda df: df['ADM1'].str.contains(north), 'space'] = 'North'
geo_df.loc[lambda df: df['ADM1'].str.contains(south), 'space'] = 'South'
geo_df.loc[lambda df: df['ADM1'].str.contains(cities), 'space'] = 'Cities'
geo_df.loc[lambda df: df['ADM2'].str.contains(r"Mfoundi"), 'space'] = 'Cities'

# Now we aggregate the data to the `space` variable
agg_geo_df = geo_df.dissolve(by = 'space')


vas.models[None].plot_map_benchmark(intervention = None,
time = 1,
optimum_interest = 'cb',
bench_intervention = 'oilvas',
map_df = agg_geo_df,
merge_key = 'space',
intervention_in_title = False,
intervention_bubbles= True,
intervention_bubble_names = ['oil', 'vas', 'cube'],
bau_intervention_bubble_names = ['oil', 'vas', 'cube'],
save = 'optimization_work/demewoz_lives_saved/reports/multi_mn_plus_lives_saved/vas1.png')

plt.gcf().text(0.5,-.05, 'Note: Effective coverage (in millions)', ha='center')

vas.models[None].plot_map_benchmark(intervention = None,
time = 3,
optimum_interest = 'cb',
bench_intervention = 'oilvas',
map_df = agg_geo_df,
merge_key = 'space',
intervention_in_title = False,
intervention_bubbles= True,
```

```
intervention_bubble_names = ['oil', 'vas', 'cube'],
bau_intervention_bubble_names = ['oil', 'vas', 'cube'],
save = 'optimization_work/demewoz_lives_saved/reports/multi_mn_plus_lives_saved/vas3.png')

plt.gcf().text(0.5,-.05, 'Note: Effective coverage (in millions)', ha='center')

vas.models[None].plot_map_benchmark(intervention = None,
time = 10,
optimum_interest = 'cb',
bench_intervention = 'oilvas',
map_df = agg_geo_df,
merge_key = 'space',
intervention_in_title = False,
intervention_bubbles= True,
intervention_bubble_names = ['oil', 'vas', 'cube'],
bau_intervention_bubble_names = ['oil', 'vas', 'cube'],
save = 'optimization_work/demewoz_lives_saved/reports/multi_mn_plus_lives_saved/vas10.png')

plt.gcf().text(0.5,-.05, 'Note: Effective coverage (in millions)', ha='center')

fig, (ax1, ax2)  = plt.subplots(1,2, figsize=(10,6))

vas.models[None].plot_bau_time('b', ax=ax1)
vas.models[None].plot_bau_time('c', ax=ax2)
ax1.set_ylabel("Millions Effectively Covered")
ax2.set_ylabel("USD (in millions)")

plt.tight_layout()
plt.savefig('optimization_work/demewoz_lives_saved/reports/multi_mn_plus_lives_saved/vas_b_c.png', dpi=
```

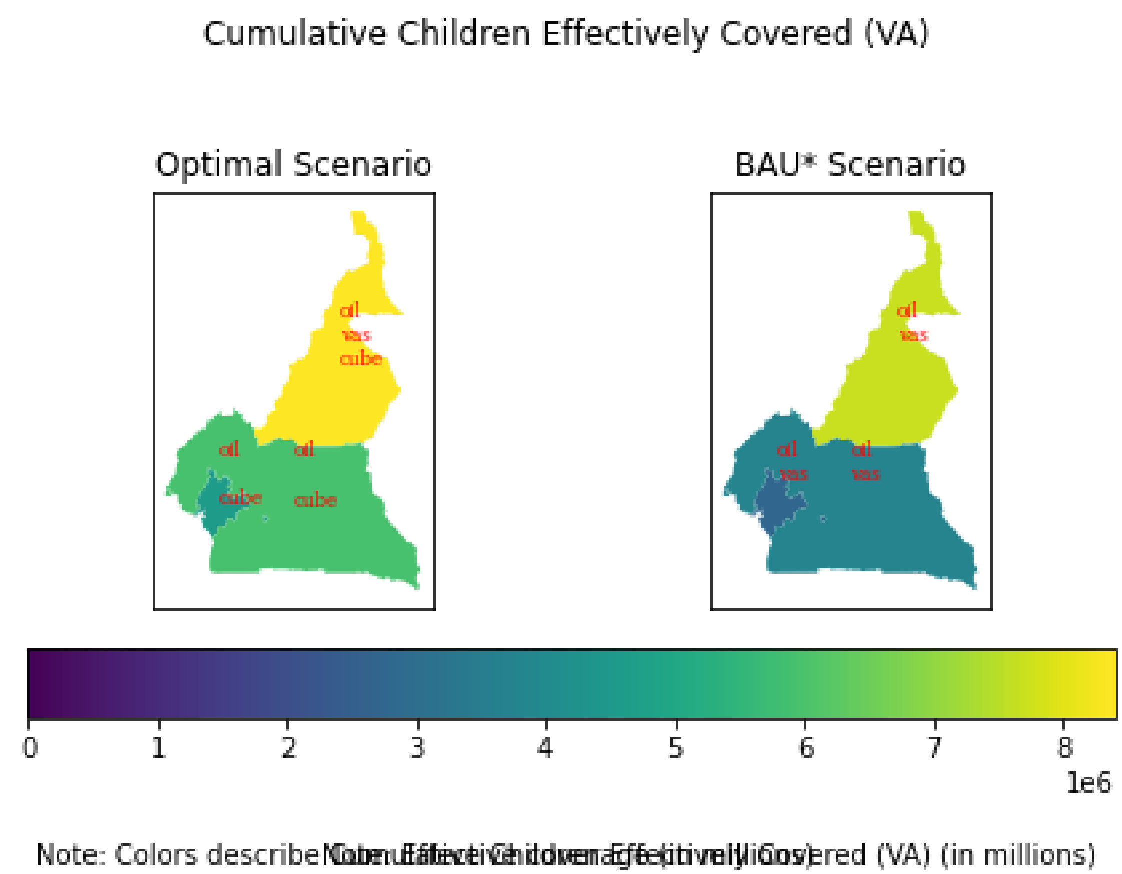
```
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
/home/lordflaron/Documents/minimod/minimod/utils/plotting.py:250: MatplotlibDeprecationWarning: The 's'
  df_bubble_final.apply(lambda x: ax.annotate(s = x.bubble_name,
```

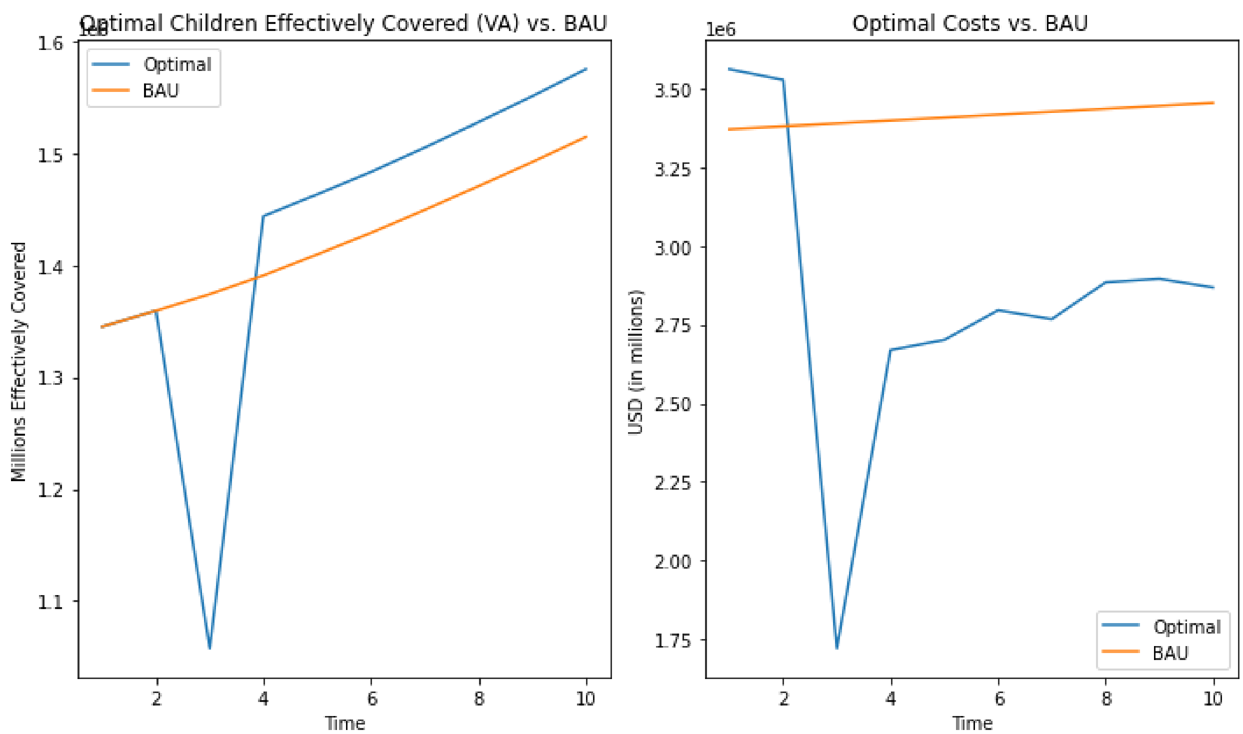


Note: Colors describe Cumulative Children Effectively Covered (VA) (in millions)



Note: Colors describe Cumulative Children Effectively Covered (VA) (in millions)





For the Vitamin A simulations, we start with fortified oil (at 75% coverage), fortified cube and a VAS routine as can be seen in +@fig:vas1. By Period 4 (in +@fig:vas3), the south and cities stop VAS campaigns, but proceed with cube and oil interventions. This continues until the last period. If we compare this to the BAU* scenario, accumulated benefits have reached around the same levels in the South and cities, but with higher benefits in the North.

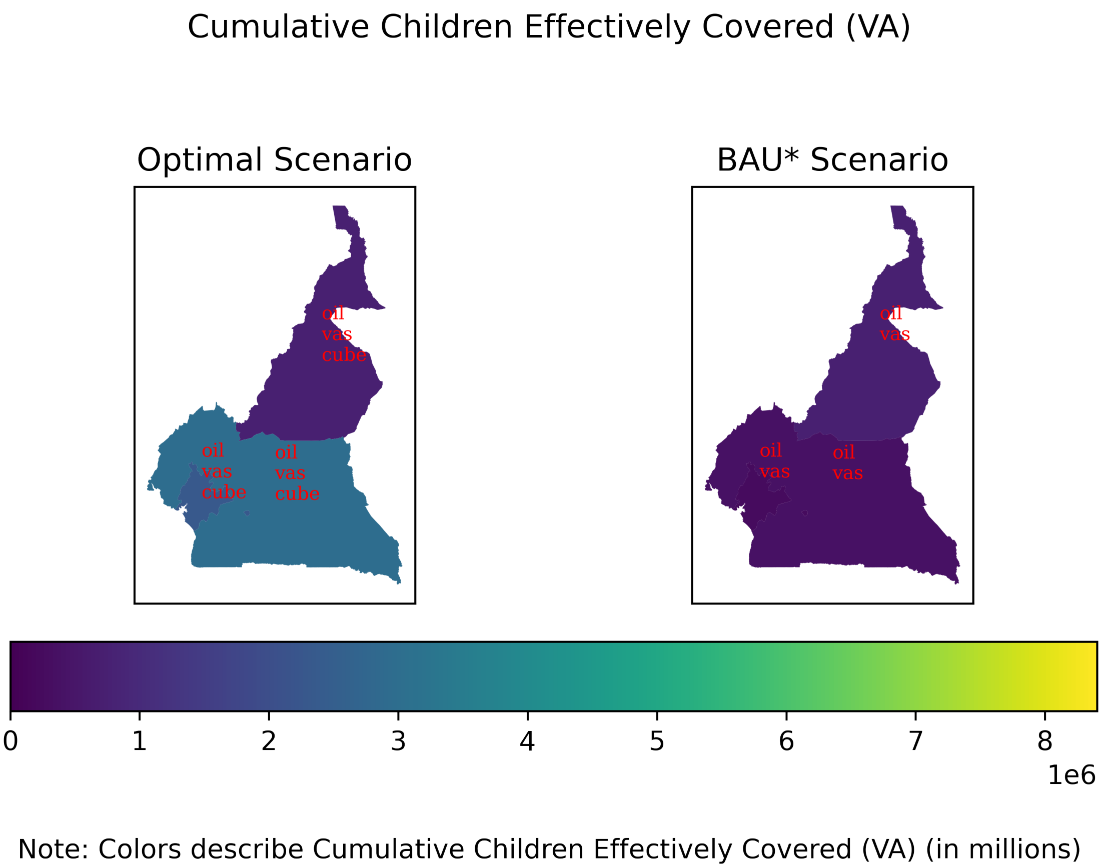


Figure 2: Vitamin A Accumulated Benefits, T=1

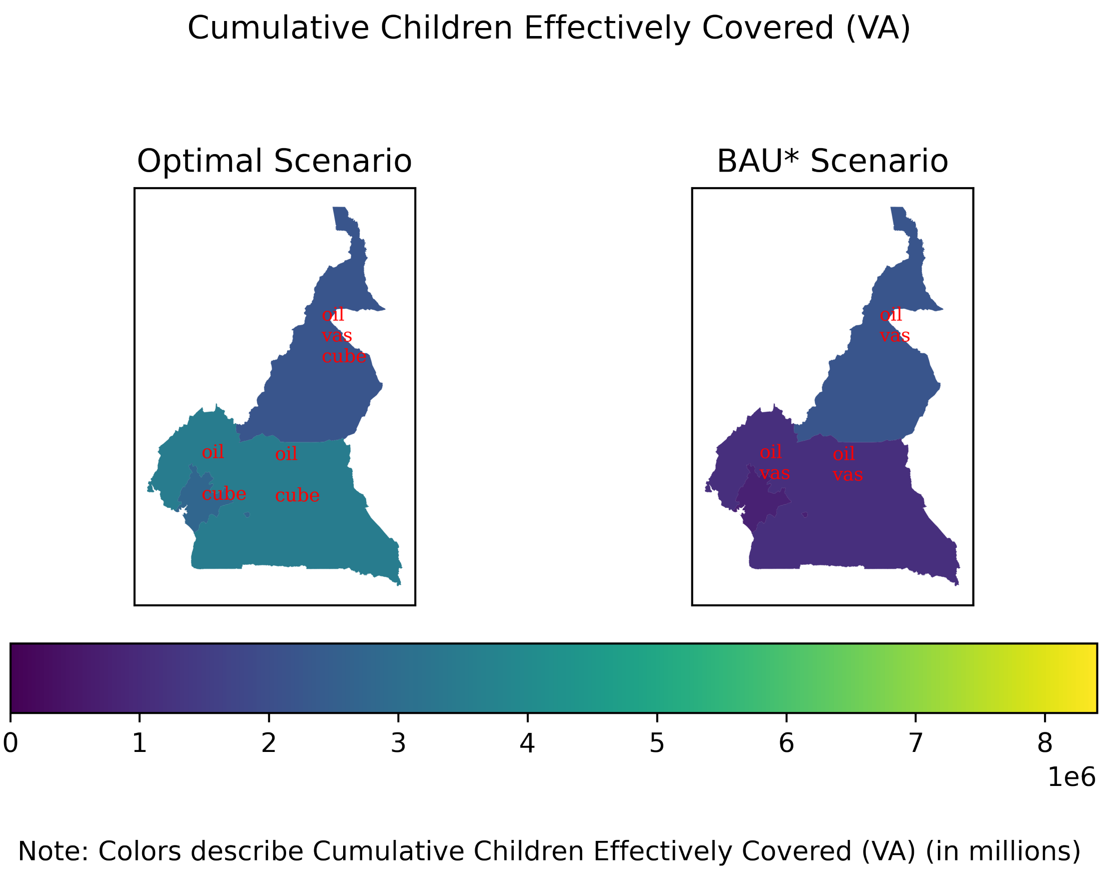


Figure 3: Vitamin A Accumulated Benefits, T=3

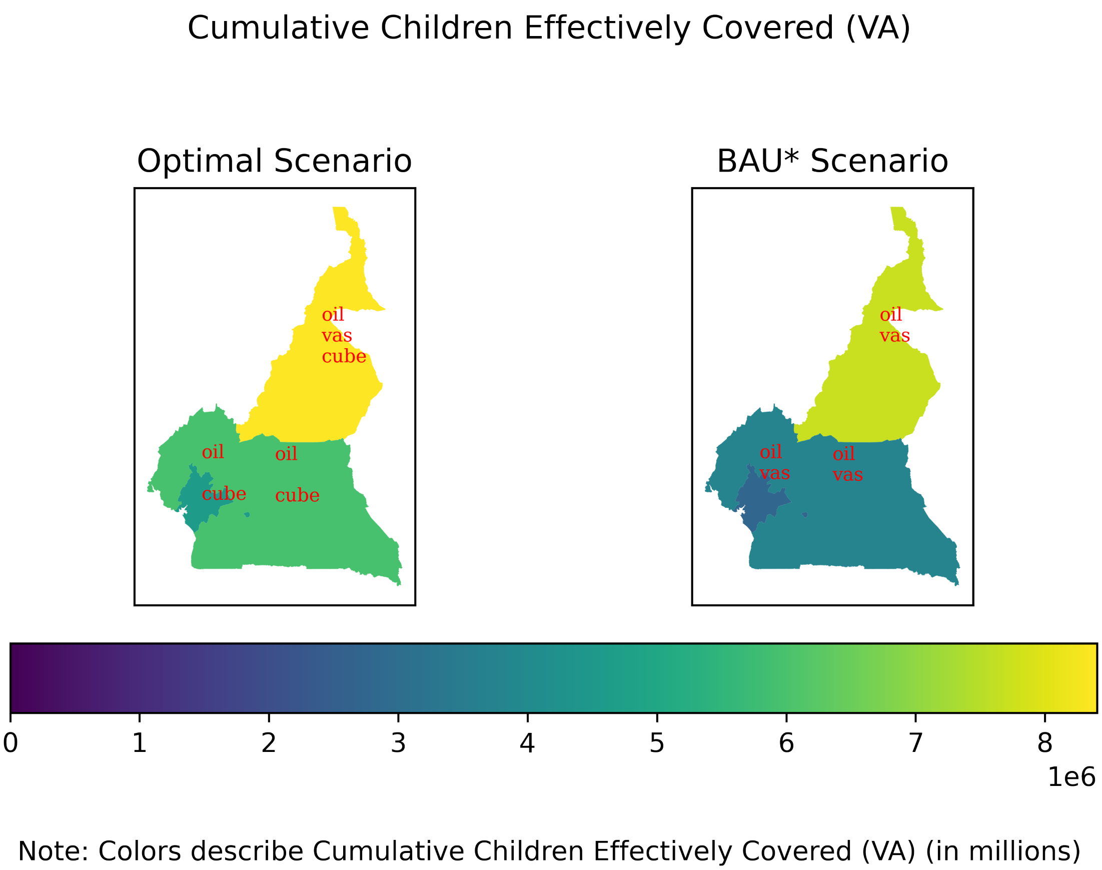


Figure 4: Vitamin A Accumulated Benefits, T=10

To illustrate the differences between benefits and costs between the optimal and BAU* scenarios, we compare the two in +@fig:vas_b_c.

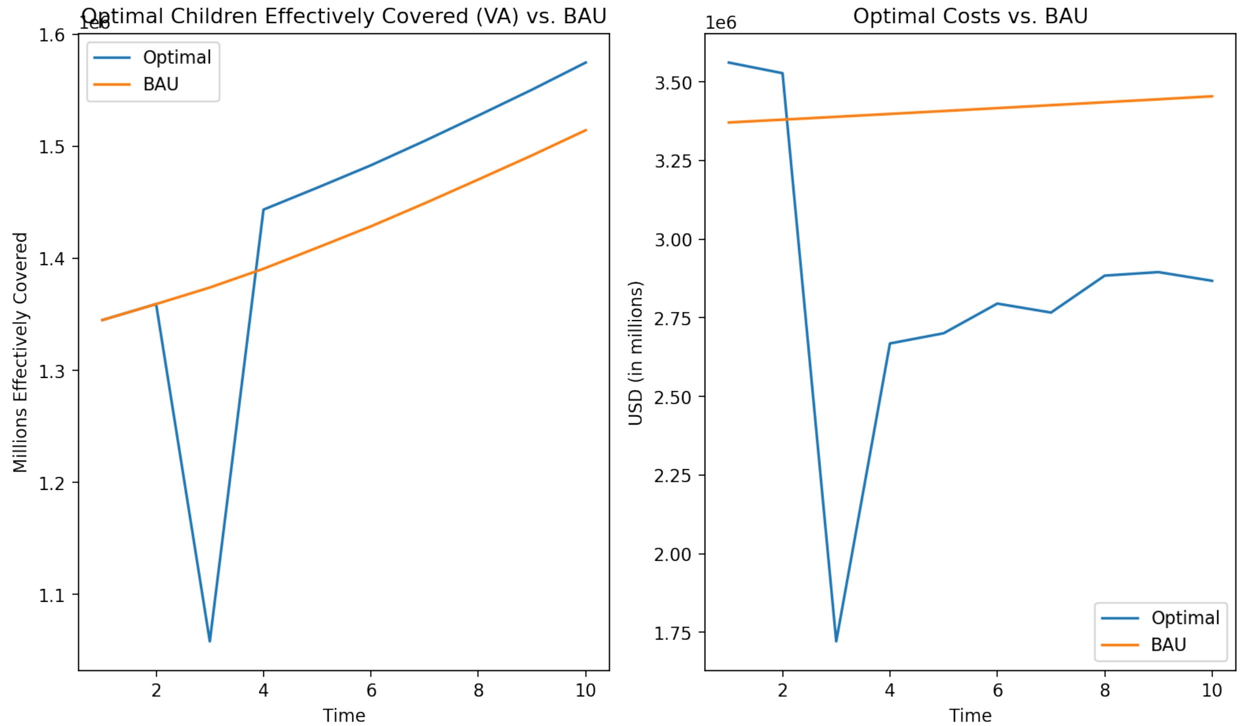


Figure 5: Vitamin A Per-year Benefits and Costs across Time

Zinc

For zinc interventions, we see that zinc fortified flour is chosen, which is the same as the BAU* scenario. This leads to the highest benefits being in the cities, followed by the south and then the North.

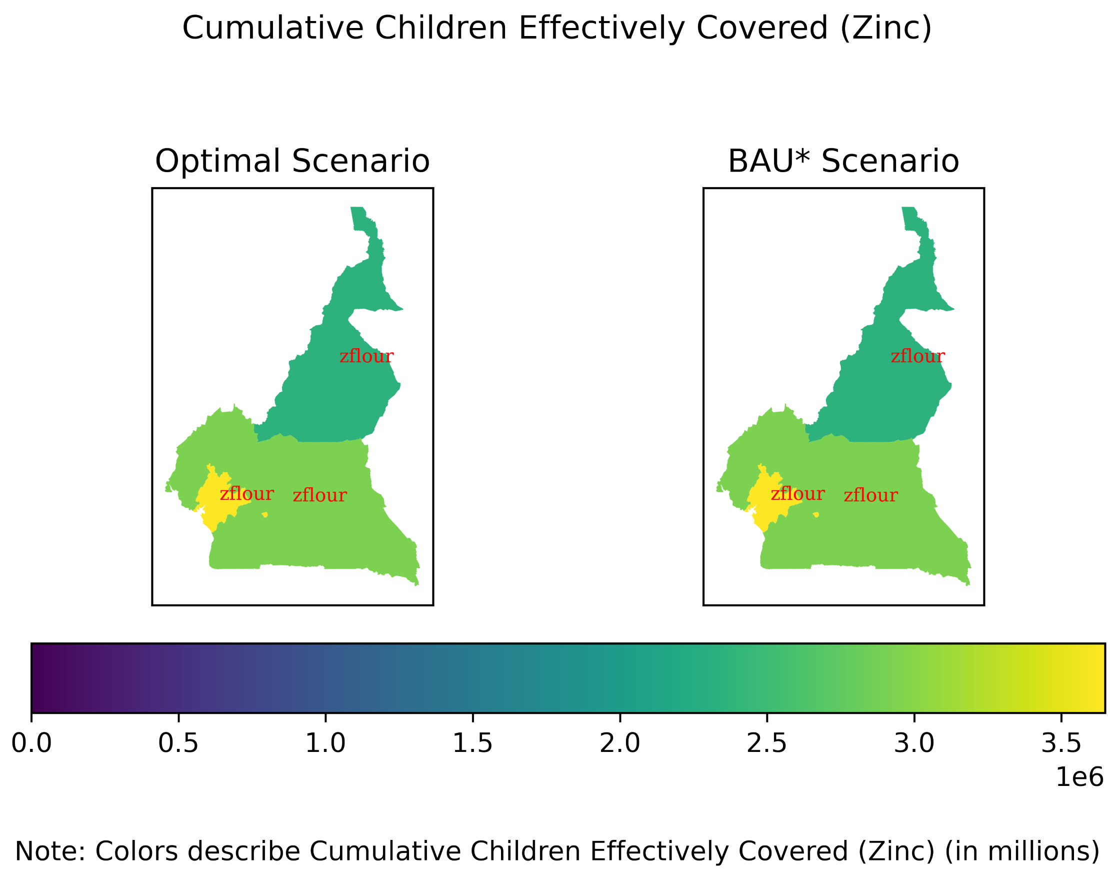


Figure 6: Zinc Accumulated Benefits, T=10

Folic Acid

Folic Acid is the same as the zinc interventions, in that the BAU* intervention is chosen as the optimal intervention. Cities are disproportionately affected, compared to the South and the North.

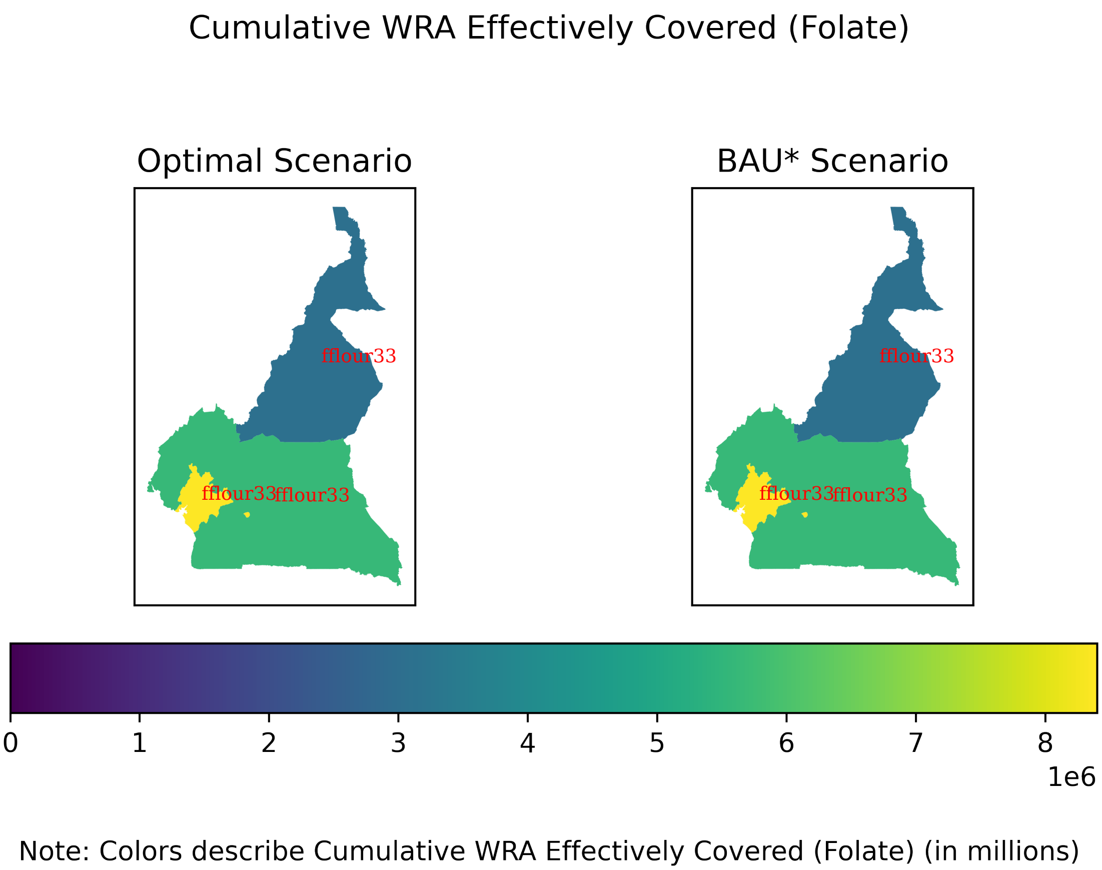


Figure 7: Folic Acid Accumulated Benefits, T=10

Lives Saved

For Lives Saved, we find that the same intervention is chosen for all periods everywhere leading to accumulated benefits that are more proportionately distributed across the country, while the BAU* scenario has higher benefits in the North.

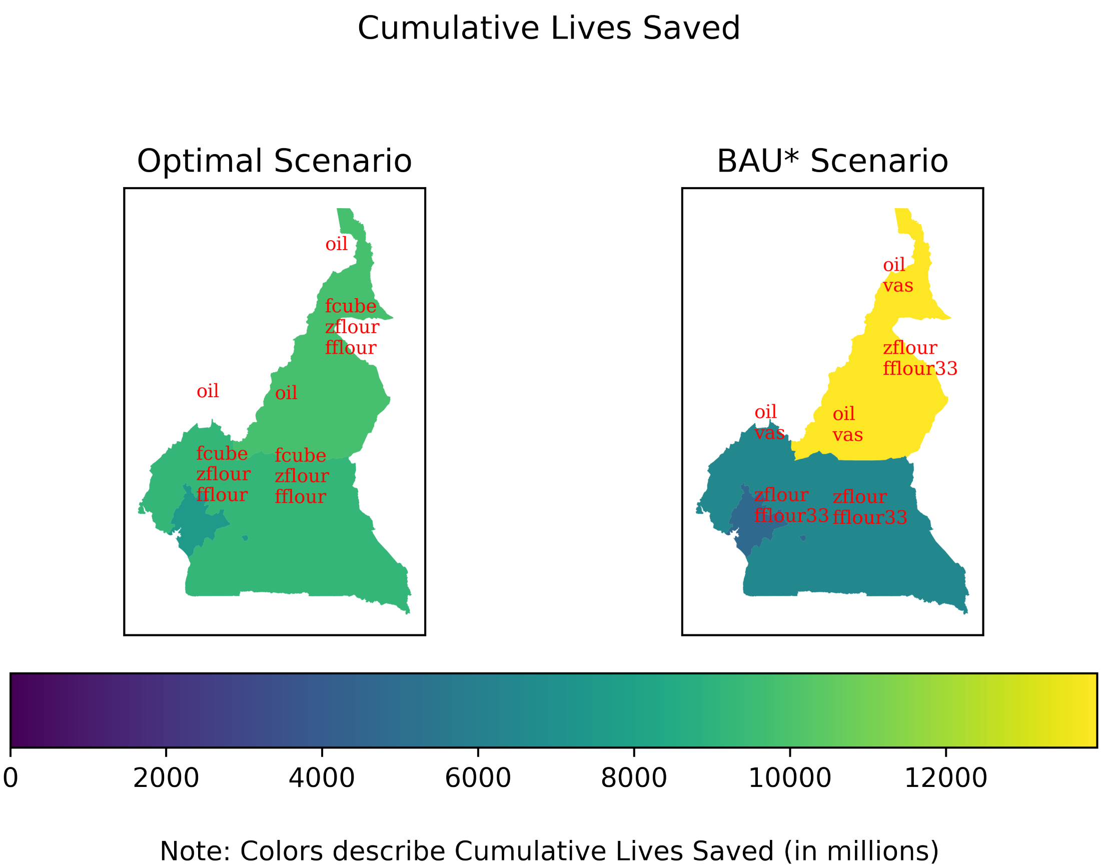


Figure 8: Lives Saved Accumulated Benefits, T=10

We also show the difference in per-year benefits and costs in +@fig:ls_b_c.

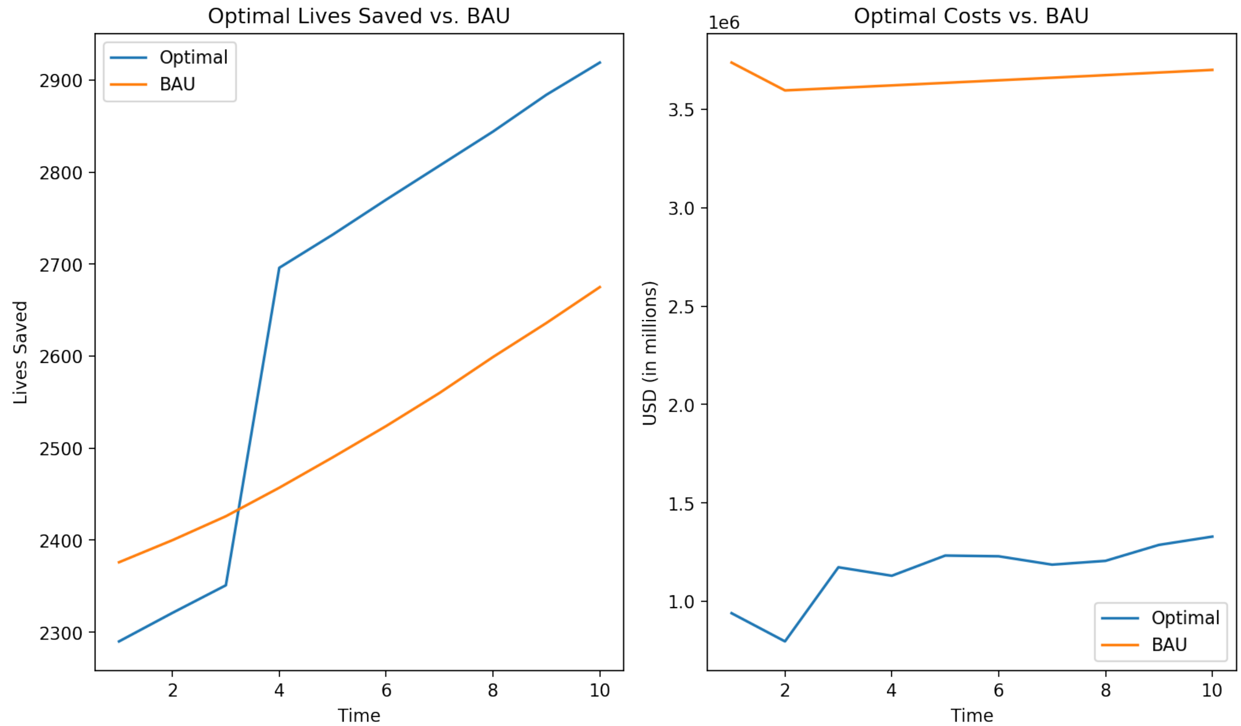


Figure 9: Lives Saved Per-Year Benefits and Costs

Lives Saved Alternative Definition

For the alternative definition of lives saved, we find that the same interventions are chosen across the country, apart from VAS campaigns in the north in periods 1-3 (+@fig:lshigh1). By period 4, we find that VAS campaigns stop and for the rest of time, the same interventions are used for all of the country. This leads to a similar geographic distribution by the end (+@fig:lshigh10) as the other lives saved definition, but with relatively higher benefits for the North.

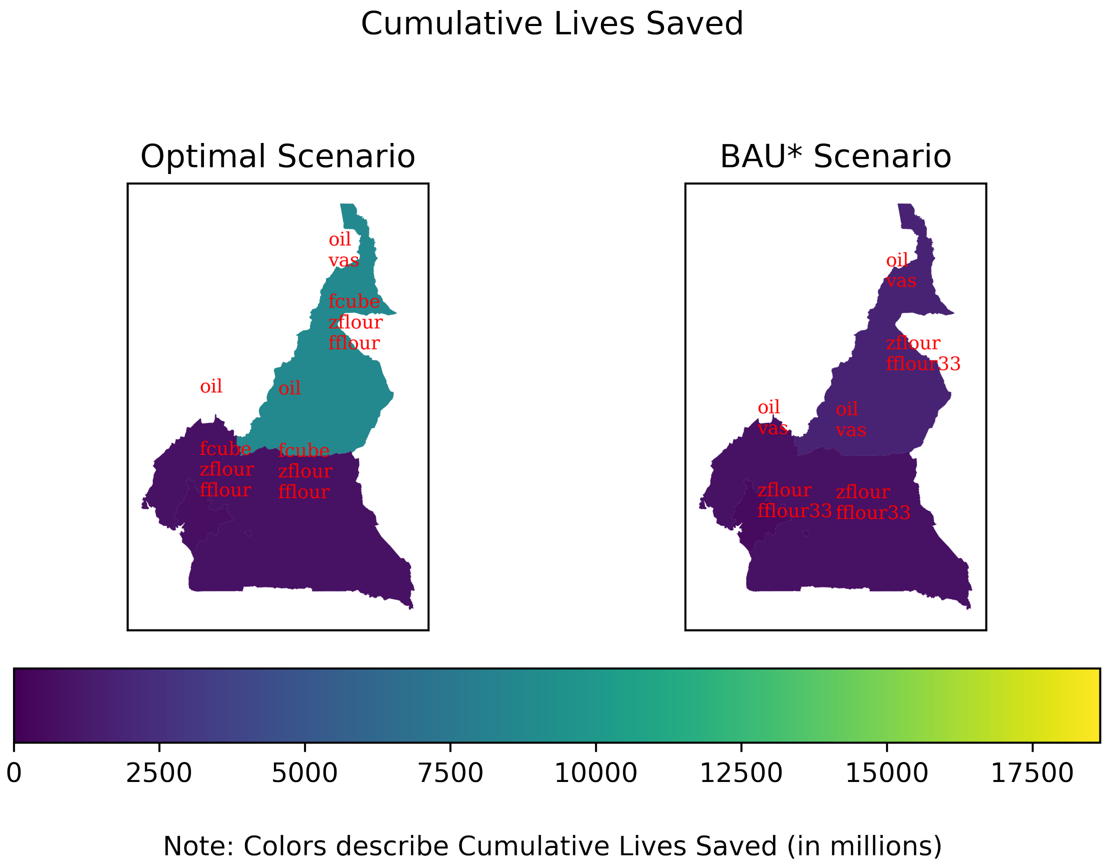


Figure 10: Lives Saved Alternative Specification Accumulated Benefits, T=1

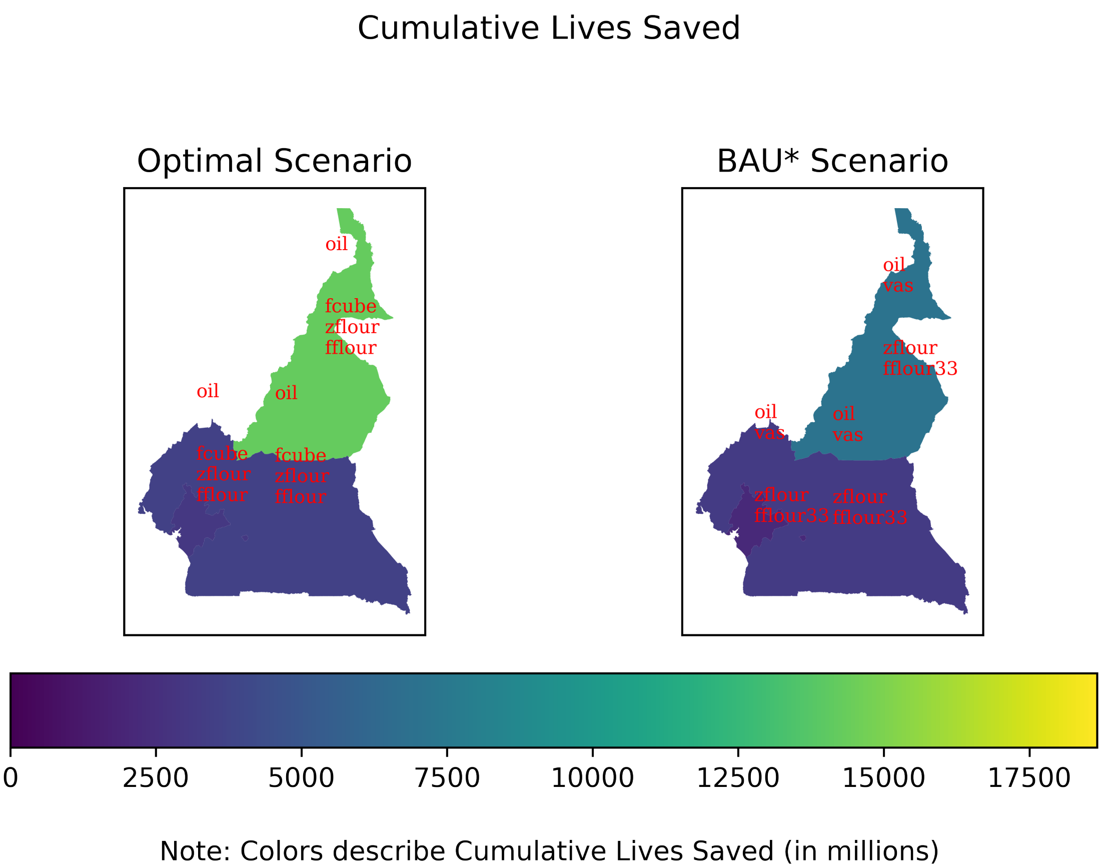


Figure 11: Lives Saved Alternative Specification Accumulated Benefits, T=4

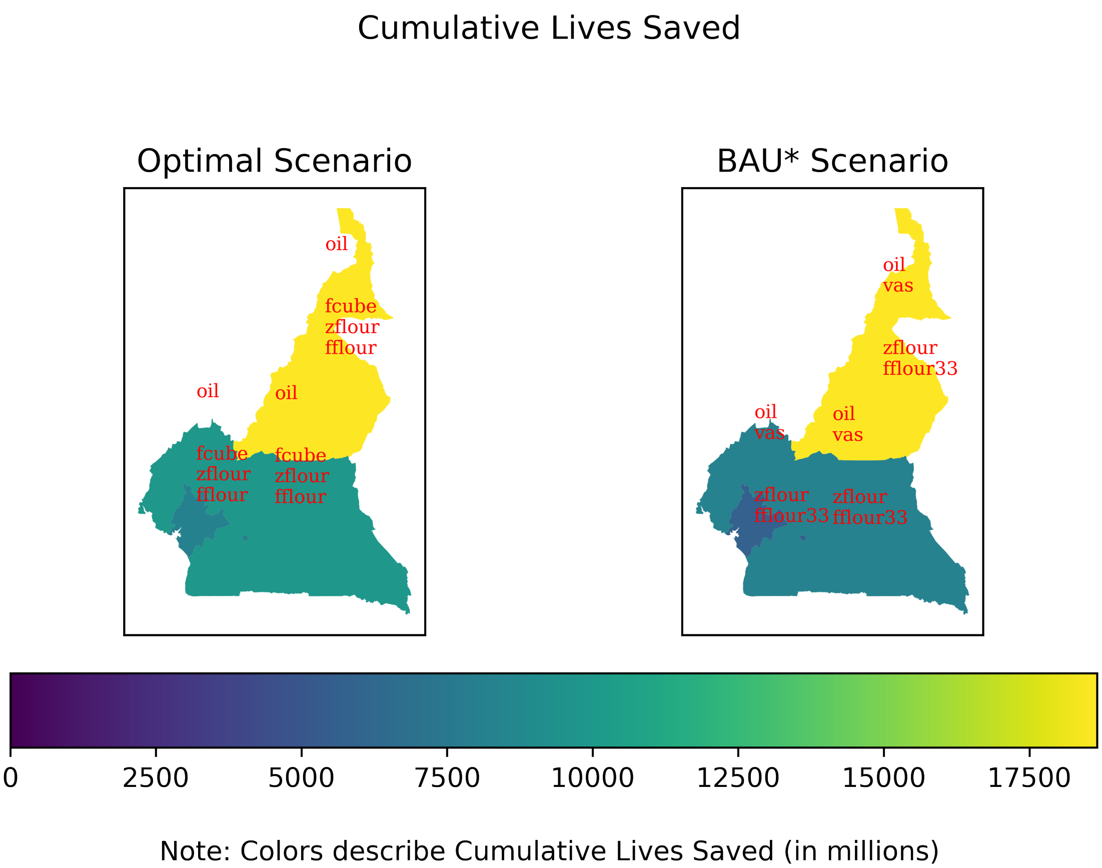


Figure 12: Lives Saved Alternative Specification Accumulated Benefits, T=10

We also show the difference in per-year benefits and costs in +@fig:lshigh_b_c.

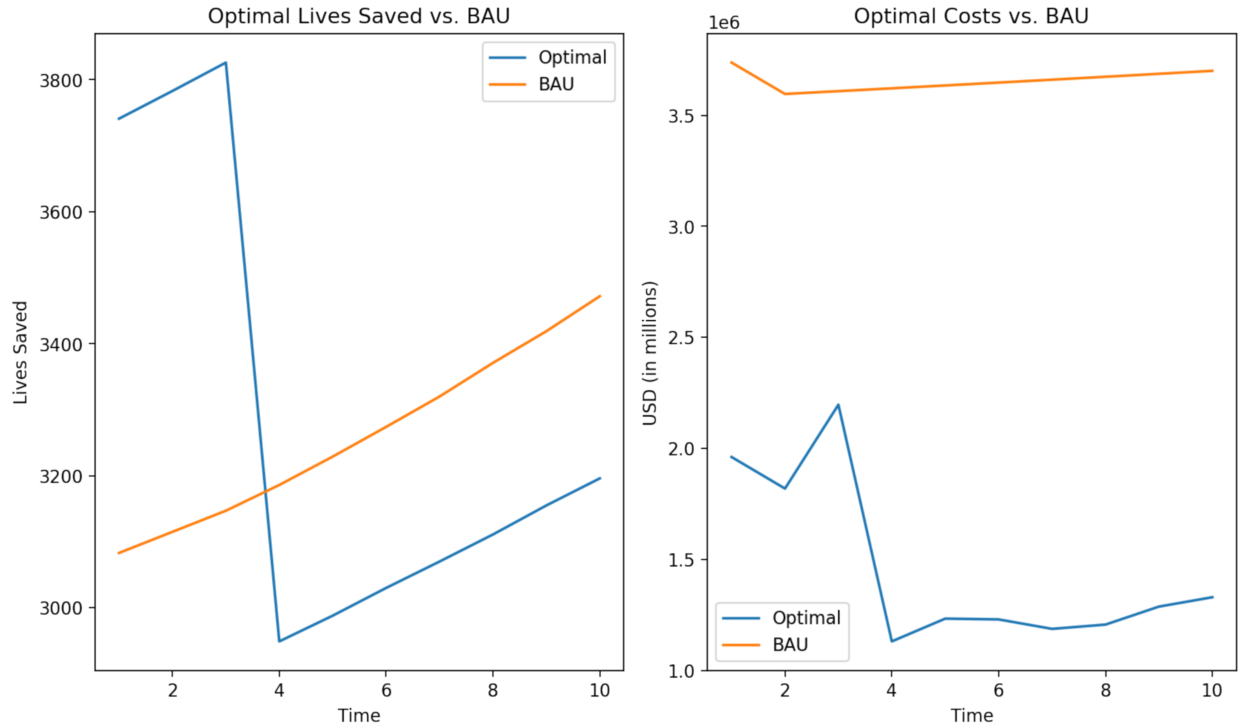


Figure 13: Lives Saved (Alt. Definition) Per-Year Benefits and Costs