**Paid Leave Microsimulation Model:**

**Python Version User Manual**

Draft

June 4, 2020



**Paid Leave Microsimulation Model:**

**Python Version Model User Guide**

Draft

June 4, 2020

**DOL-IMPAQ PAID Leave microsimulation model**

**Python Model User Guide**

This document provides a step-by-step guide for users of the paid leave microsimulation model developed by IMPAQ International (IMPAQ) and Institute for Women’s Policy Research (IWPR) for the Chief Evaluation Office at the United States Department of Labor (DOL). With this guide, users should be able to set up the needed computing environment, properly specify model parameters, launch the microsimulation program, interpret the model output, and understand the potential of model extension.

**Setting up the computing environment**

Hardware requirements – Current version of the model has been tested on mainstream workplace and home computers with Intel i5 and i7 multicore processors, resulting in manageable runtime (within an hour) even for large states from the American Community Survey (ACS) 5-year public use microdata sample (PUMS) such as California. Runtime is less than 5 minutes for small states such as Rhode Island. Minimum RAM tested is 8GB which is sufficient to handle large ACS states (California data is less than 2GB), although we recommend 16GM RAM or higher for better runtime performance. To store ACS data for all states, a disk space of 25GB is required, as the original ACS state household files and state person files (based on state of residence) have a total file size of about 12GB, while user would also need another 12GB to store the place-of-work based ACS state files, which are an alternative partition of all ACS persons in the country. The file sizes of the other two input datasets, the Family and Medical Leave Act (FMLA) Employee Survey and the Current Population Survey (CPS) data sets are minimal compared to ACS thus have limited impact on the hardware requirements.

Software requirements – Current Python model is coded in Python 3, and the Graphic User Interface (GUI) can be launched from terminal by running Microsimulator.py. Windows users would need to have Python 3 as well as the necessary packages installed before running the model. Users can quickly install the needed packages by following these steps:

1. Open the Windows Command Prompt or PowerShell.
2. Navigate to the *Microsimulator/* directory.
3. Type the command *pip install –r requirements.txt* and then hit enter.

Users who already have any of the required packages installed but are unable to update them to the latest versions can use the command *pip install <package>*. Replace *<package>* with one of the packages below.

|  |  |
| --- | --- |
| * + cycler==0.10.0   + kiwisolver==1.1.0   + matplotlib==2.2.3   + mord==0.5   + numpy==1.17.4   + pandas==0.23.0 | * + pyparsing==2.4.5   + python-dateutil==2.8.1   + pytz==2019.3   + scikit-learn==0.20.1   + scipy==1.3.3   + six==1.13.0 |

Advanced Python users who might not be able to make changes to their environment can install the packages to a virtual environment. Follow the steps below if you have these restrictions:

1. Open the Windows Command Prompt or PowerShell.
2. Navigate to the *Microsimulator/* directory.
3. Type the following commands, hitting enter after each command.
   * *python -m venv microsim-env*
   * *microsim-env\Scripts\activate.bat* (from Command Prompt) or *microsim-env\Scripts\activate.ps1* (from PowerShell)
   * *pip install –r requirements.txt*

For the greatest simplicity, we recommend installing [Anaconda](https://www.anaconda.com/), which can greatly facilitate Python package installation for Windows users. The tool will work with the Anaconda 5.3.0 distribution. Users with older releases can update their packages by running the command *conda update --all*. The only package not included in Anaconda distributions is *mord*, which can be installed using the command *conda install –c mord*.

* Dataset requirements
  + ACS – Current model has been tested on 5-year ACS PUMS for periods 2012-2016, 2013-2017, and 2014-2018. For original ACS data files, user should download the desired state files from [Census](https://factfinder.census.gov/faces/tableservices/jsf/pages/productview.xhtml?pid=ACS_pums_csv_2012_2016&prodType=document). Person files should be placed in *./data/acs/[year]/person\_files*, while household files should be placed in *.data/acs/[year]/household\_files*, where [year] should be replaced by the ending year of the 5-year ACS period (i.e., 2016, 2017, or 2018). User should ensure file names of these Census data files follow the format below for proper data reading by the program.

|  |  |
| --- | --- |
| * ACS household files: | *ss[yr]h[st].csv*, where *[yr]* = last two digit of ending year of the 5-year ACS, and *[st]* = 2 digit FIPS state code. |
| * ACS person files: | *ss[yr]p[st].csv*, where *[yr]* = last two digit of ending year of the 5-year ACS, and *[st]* = 2 digit FIPS state code. |

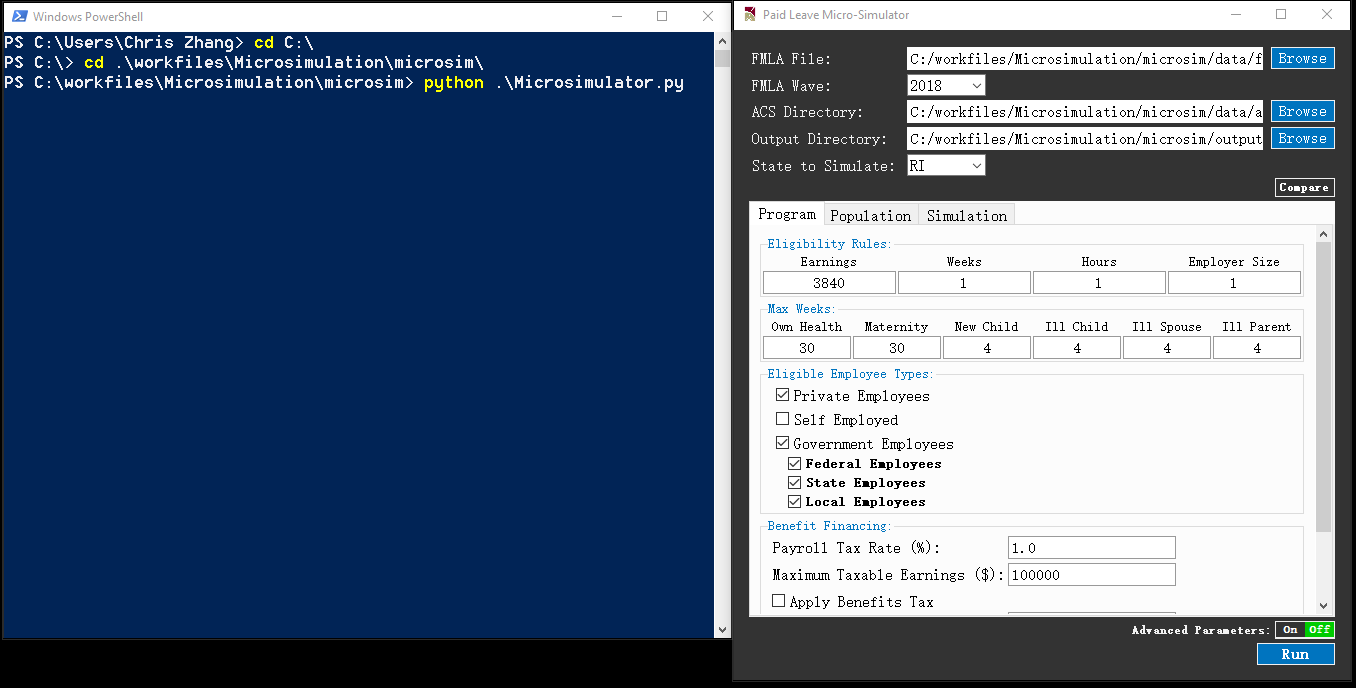
For place-of-work based ACS state data, we have generated the state person and household files which are placed in *./data/acs/[year]/pow\_person\_files*, and *./data/acs/[year]/pow\_household\_files* respectively. File names within these folders should not be changed.

* + FMLA – Current model uses either the FMLA 2012 or FMLA 2018 data, and should be placed in *./data/fmla\_[year]*, where *[year]* is either 2012 or 2018, representing the two survey waves.
  + CPS – the CPS microdata are used for auxiliary simulation of a few covariates needed for simulation in both FMLA and ACS data samples. Current model uses CPS March Annual Social and Economic (ASEC) Supplement in 2014, 2015, or 2016, the middle year corresponding to the three available 5-year ACS PUMS respectively. Data files containing the needed CPS data columns should be named as *cps\_clean\_[year].csv*, where *[year]* is the year of the CPS data, and should be placed in *./data/cps*.

**Running the model**

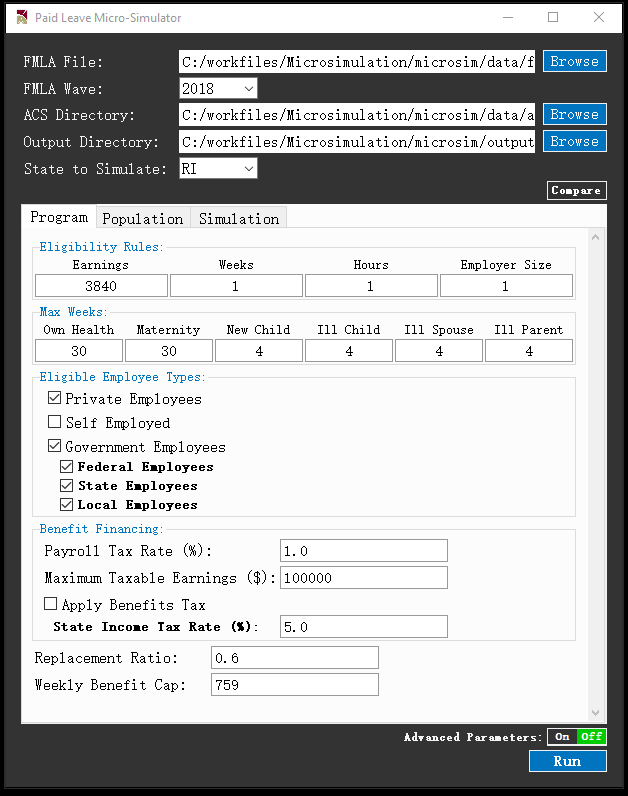
* Launching the model – After Python 3 and above-listed Python packages are installed, the model can be launched in a terminal (e.g. a Windows command line window). User should first change directory to the *microsim* folder, and run the command python *Microsimulator*.*py*. The GUI will then be launched, as shown in **Exhibit 1** below.

**Exhibit 1: Launching the Model**



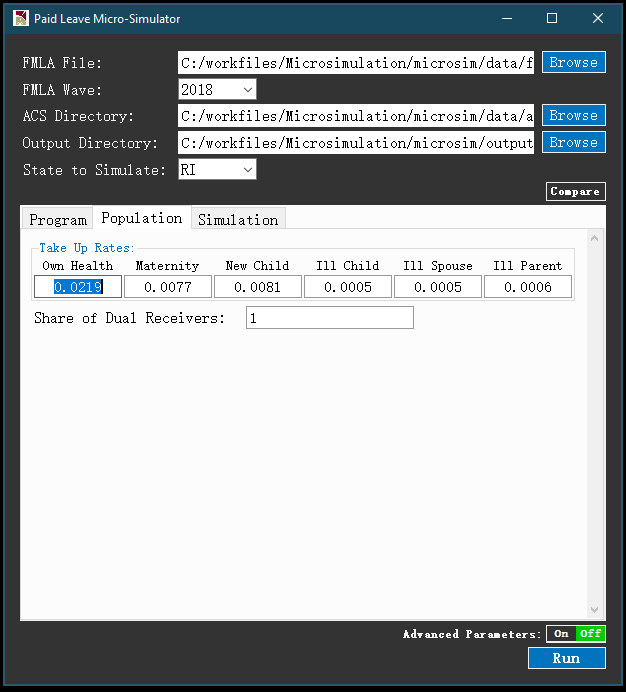
* Specifying input data and parameters – Once the above GUI is displayed. User can specify data sources and model parameters.
  + FMLA File – File path to the FMLA dataset. Current default is to use the FMLA 2018 data located at *./data/fmla\_2018/*
  + FMLA Wave – Wave year of the FMLA dataset. User should specify either 2012 or 2018, and verify that the wave year is consistent with the file path specified in FMLA File.
  + ACS Directory – Directory of ACS datasets. Current default is *./data/acs/* which contains 3 subdirectories *2016*, *2017*, and *2018*, representing 3 different 5-year ACS PUMS: ACS 2012-2016, ACS 2013-2017, and ACS 2014-2018. Within each year subdirectory, there are 4 subdirectories: *household\_files, person\_files, pow\_household\_files,* and *pow\_person\_files*. The former 2 subdirectories contain ACS state PUMS datasets that will be used if user choose to uncheck *State of Work*. The latter 2 subdirectories contain ACS state PUMS datasets that will be used if *State of Work* is checked.
  + Output Directory – Directory where output files will be stored upon completion of simulation.
  + State to Simulate – ACS state PUMS dataset to use as underlying worker population. The dropdown menu contains 50 states plus DC.
  + *Program* tab – This tab contains the following parameters (a full list of *Program* parameters is shown in **Exhibit 2**):
    - Eligibility Rules – These are minimum requirements on annual earnings (in dollars), number of weeks worked over a year, number of hours worked over a year, and number of employees at workplace for a worker to be eligible to receive leave benefits from the program.
    - Max Weeks – These are maximum number of weeks for which an eligible worker can receive leave benefits from the program. User can set different value of maximum number of weeks for each of the 6 leave types.
    - Eligible Employee Types
      * Private Employees – If checked, workers working for private business employers will be included as eligible workers.
    - Benefit Financing – These are parameters of the payroll tax on eligible workers. *Payroll Tax* is an integer value representing percentage points of tax rate. *State Average Tax Rate* is the estimated average state income tax, which would be applicable if the *Benefit Tax* checkbox below is checked, in which case the state leave program benefits would also be subject to payroll taxation, allowing state to recoup tax revenue from paid program benefits. *Maximum Taxable Earnings Per Person* is an integer value that places a cap on annual taxable earnings that are subject to this payroll tax.
    - Replacement Ratio – Share of wage that would be replaced by program benefits during leave. The ratio should be a positive value between 0 and 1.
    - Weekly Benefit Cap – Maximum weekly benefits in dollars for each leave type. Current model assumes a uniform cap for all 6 leave types.

**Exhibit 2: Model Parameters under Program Tab**



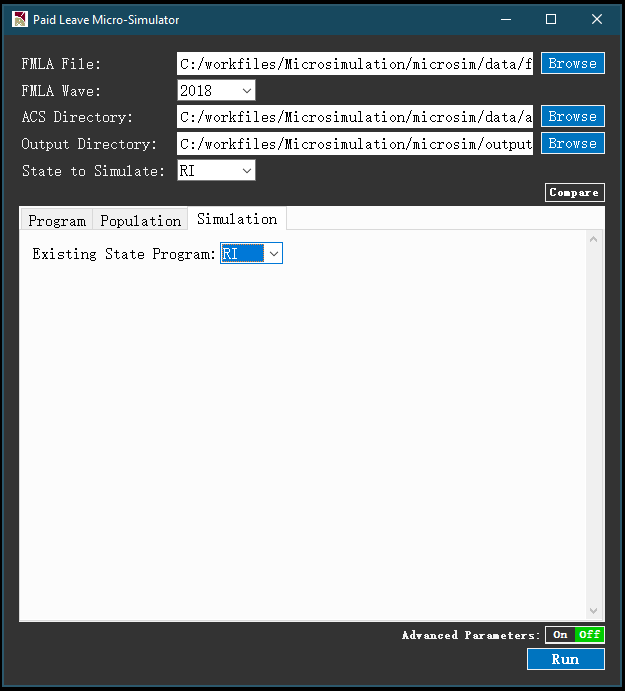
* + *Population* tab – This tab contains the following parameters (a full list of *Population* parameters is shown in **Exhibit 3**):
    - Take Up Rates – These are take up rates of the benefit for each leave type among all eligible workers in the state. Namely, take up rate is total number of actual leave-taking workers under program (thus equivalent to all cases approved in model) divided by total number of eligible workers in the state.
    - Share of Dual Receivers – Share of eligible workers who can receive leave benefits simultaneously from both employer and state program, out of all eligible workers who receive any leave pay benefit from employer. This share should be a value between 0 to 1.

**Exhibit 3: Model Parameters under Population Tab**



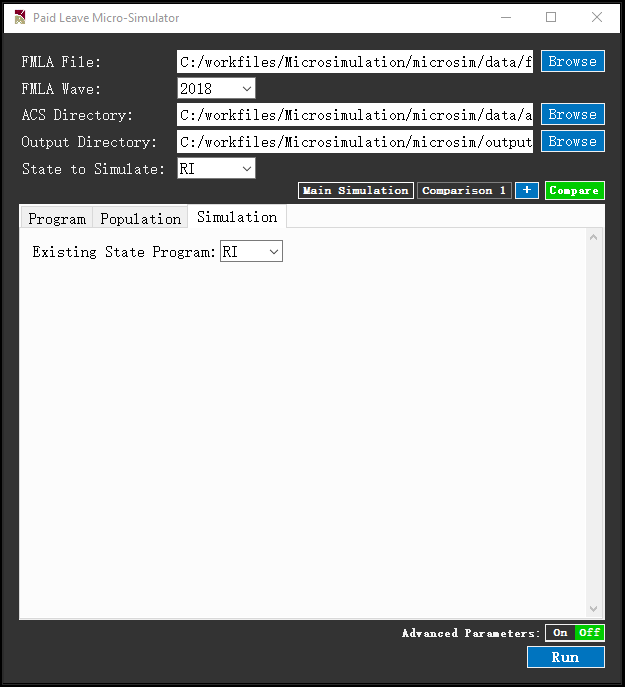
* + *Simulation* tab – This tab contains the following parameter:
    - Existing State Program – Existing state leave program parameters to use. If a state is selected, parameters under *Program* tab will be overridden by a set of pre-determined parameters that best represent the leave program of the selected state, and parameters under *Population* tab will be overridden by a set of empirical estimates, including take up rates for each leave type estimated from historical state program data.

**Exhibit 4: Model Parameters under Simulation Tab**



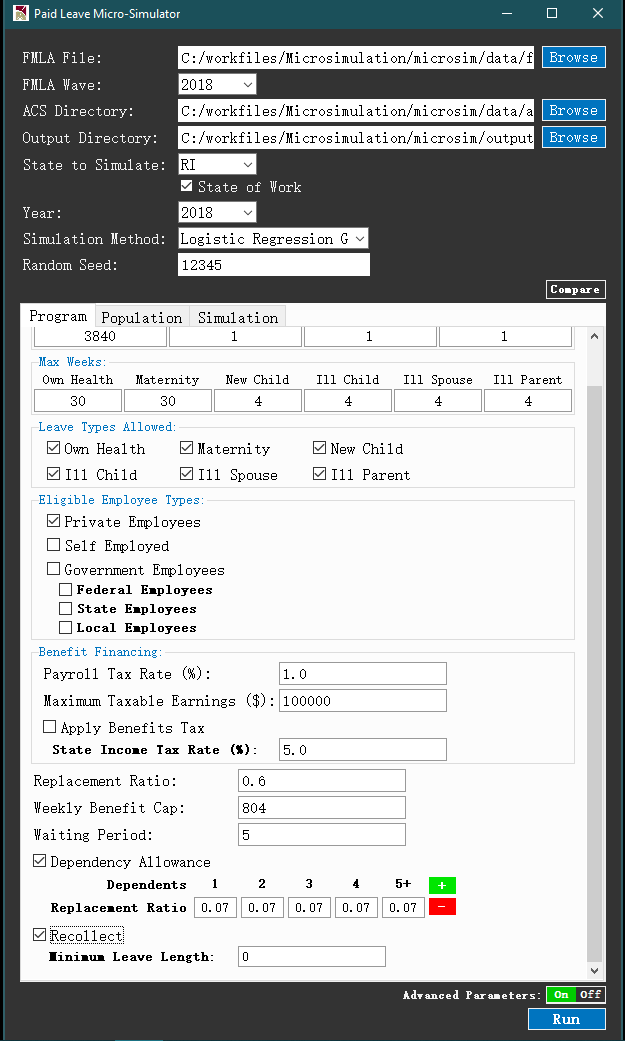
* + Compare Button – When clicked, this button will show the full list of parallel simulation to run for comparison, next to the *Compare* button. Each parallel simulation will be performed using an alternative set of program parameters from a selected state, based on user choice of state in a separate *Existing State Program* dropdown list for that parallel simulation. For example, in **Exhibit 5**, after clicking *Compare*, a new button ‘Comparison 1’ is added next to the default ‘Main Simulation’. User can then click the ‘Comparison 1’ button which will lead to a *Simulation* tab which asks for *Existing State Program* parameter for *Comparison 1*. Likewise, user can further add more parallel simulations by clicking the *plus* button next to *Compare*, resulting in *Comparison 2*, *Comparison 3*, etc. The comparison results will be eventually displayed in GUI result window, as well as saved in the output folder.

**Exhibit 5: Comparison Button**



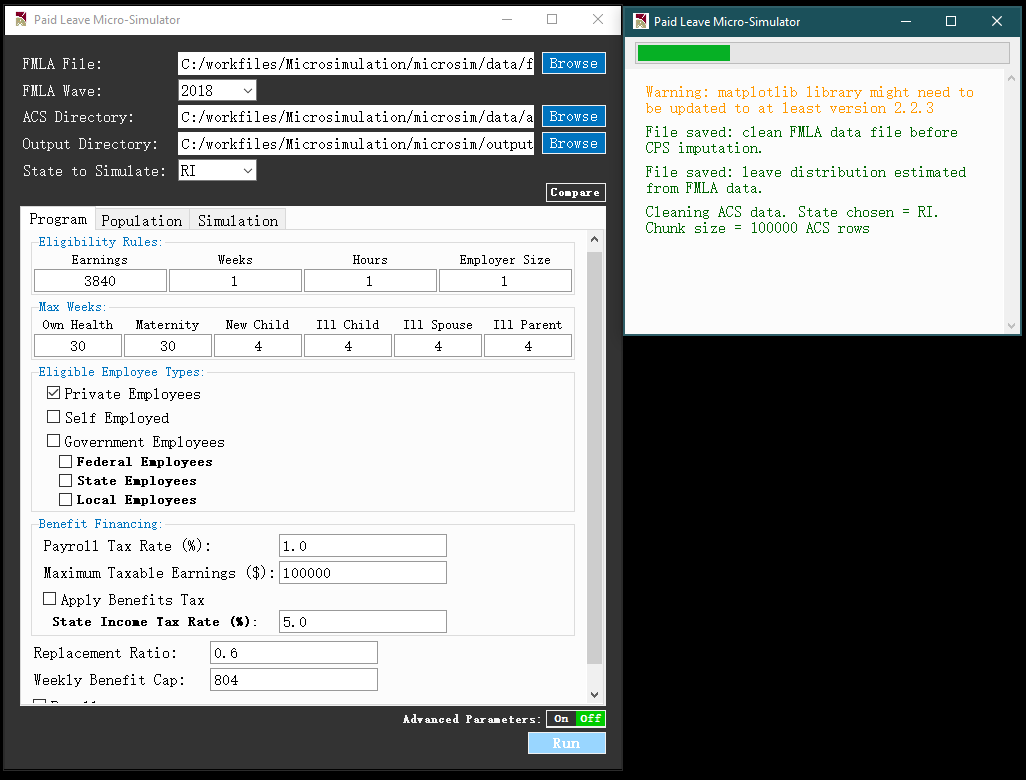
* + *Advanced* button – When clicked, this button will display advanced parameters in GUI as shown in **Exhibit 6**. Current model has following advanced parameters:
    - State of Work – If checked, state of workers will be determined by state of workplace. If unchecked, state of workers will be determined by state of residence. This parameter is checked as default, following common practice of currently implemented state programs.
    - Simulation Method – This dropdown menu in the GUI main panel allows user to specify the classifier to be used for simulation. Current model has implemented Logistic Regression under General Linear Model (GLM), Regularized Logistic Regression, Ridge Classifier, K Nearest Neighbor, Naïve Bayes, Support Vector Machine, Random Forest, and XGBoost.
    - Random seed – If Random Seed is checked, then each run of the model will correspond to a machine-generated pseudo-random state, such that difference between runs should be attributed to randomness if all parameters and input data are held constant.
    - Leave Types Allowed – Under the Program tab, user can choose to include/exclude leave types allowed by the program.
    - Wait Period – Under the Program tab. This is the number of days applicants need to wait from approval until receiving benefits.
    - Dependency Allowance – Under the Program tab. If checked, user can specify number of spouse and child dependents and the associated *incremental* wage replacement ratio offered by the program. For example, in **Exhibit 6**, the dependency allowance profile means that the program would offer an additional 7% wage replacement for each additional spouse or child dependent of the applicant, up to a maximum of 5 dependents.
    - Recollect – Under the Program tab. If checked, approved applicants can recollect program benefits incurred during the above-specified waiting period. The Minimum Leave Length value under Recollect is the minimum number of program-paid leave days required for recollection.
    - Minimum Leave Length Applied – Under the Population tab. This is the minimum number of leave day applied by applicant under the program. Default is 5 work days, consistent with existing program statistics that imply all claims and benefits are processed in unit of work weeks.
    - Alpha – Under the Population tab. This is a hyperparameter of the simulation model that represents how workers with long leave needs would be more likely to take up the program. Alpha = 0 means probability of program take up is independent from number of leave days needed. The link between take up and leave need is stronger as value of Alpha increases. Alpha can take any positive values (including decimal values), subject to bounding conditions imposed by ACS sample size. Model calibration based on existing program statistics suggests that Alpha can be as low as 0, and as high as 2.
    - Clone factor – Under the Simulation tab, user can specify an integer value to clone the ACS sample, thus be able to obtain more granular simulation results for more ACS persons, with the population weight of each person reduced proportionally to ensure proper aggregation. This feature can be useful for smaller ACS samples (such as smaller states) where simulation results may not reach a desirable level of sample variation.
  + *Run* button – After configuring all parameters above, user may click the *Run* button to execute the simulation program

**Exhibit 6: Advanced Model Parameters, Main Panel and Program Tab**



* Executing the model
  + Runtime display – After *Run* button being clicked, a runtime window will be displayed as in **Exhibit 7**. The runtime display shows a progress bar that represents estimated progress of current execution, and a series of runtime messages that indicates completion of key intermediate steps. At the beginning, warning message will be displayed if user’s Python environment has modules for which update is recommended.

**Exhibit 7: Runtime Display during Model Execution**



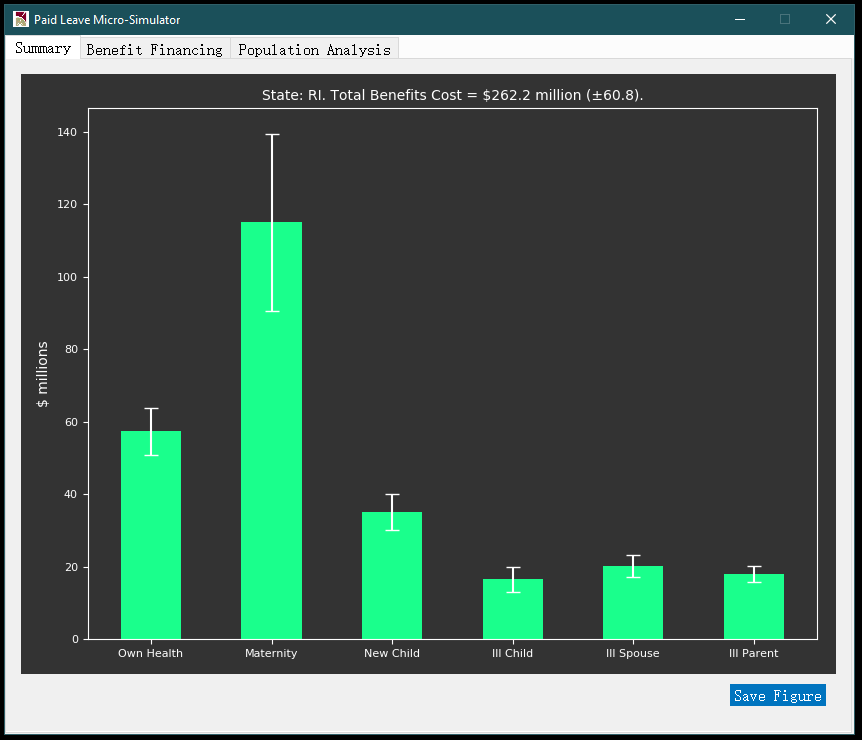
* + Runtime estimates – Time needed for completing the simulation is mainly affected by the size of the ACS state PUMS data. For smaller states with fewer rows, such as Rhode Island, executing the main simulation would take less than a minute on a Windows 10 machine with an Intel i7 processor and 16GB RAM. For the largest state California, executing the main simulation on the same machine would take about 14 minutes. These runtime estimates are all based on setting *Simulation Method* to *Logistic Regression*, which should be considered as the baseline classifier. For other classifiers, runtime may slightly differ. The exception is *Support Vector Machine*, a significantly more time-consuming method that relies on iteratively searching for a hyperplane in data features space that separates two classes (e.g. leave takers and non-takers). For example, when implemented with the Rhode Island population, runtime is 5 times slower under SVM compared to under logistic regression.

If user chooses to perform comparison simulations (i.e. comparing against existing program), runtime would increase as simulation steps will be performed multiple times on ACS data. However, the data cleaning step for FMLA and ACS data would only be performed once.

Post-simulation

* Simulation results in GUI – Upon completion of simulation, a result window will be displayed, with numerical and graphical results grouped in following tabs
  + *Summary* tab – As shown in **Exhibit 8**, this tab shows a graph that plots the estimates of program outlays for each of the six leave types. Total outlay estimate is displayed in the title of plot. There is a *Save Figure* button at bottom right corner for user to save the graph at a desired local directory.

**Exhibit 8: Summary Tab in Result Window**



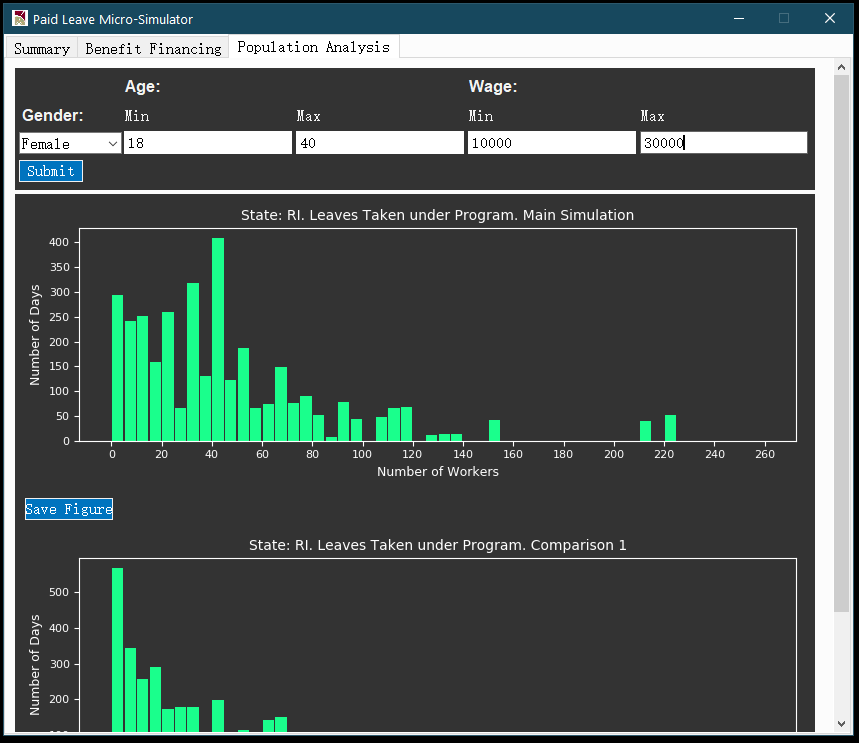
* + *Benefit Financing* tab – As shown in **Exhibit 9**, this tab displays a full set of results to illustrate how much program outlay can be financed given user-supplied Benefit Financing parameters, and how the tax revenues would be collected from different sub-populations, such as age groups, worker classes, and age groups. The top section of **Exhibit 9** shows that about $61 million would be generated by a 1% payroll tax, thus given the $262 million program outlay estimate in **Exhibit 8**, it can be estimated that the program would be funded at 61/262 = 23.3% by this payroll tax.

**Exhibit 9: Benefit Financing Tab in Result Window**

|  |  |
| --- | --- |
|  |  |

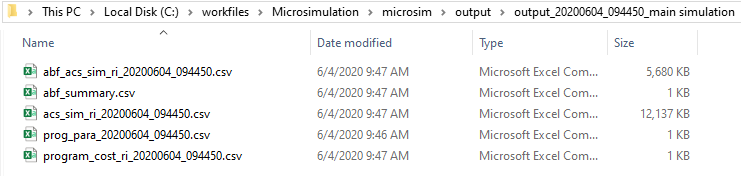
* + *Population Analysis* tab – As shown in **Exhibit 10**, this tab plots histogram of annual total leave length (in days) taken by eligible workers. Each histogram corresponds to a different simulation (e.g. Main/Comparison 1). On top of the panel, user can specify subpopulation of interest by selecting gender, age range, and range for annual wage income. Upon clicking the *Submit* button, the histograms below will be updated, showing the distribution of leave lengths for the specified subpopulation. Each histogram has its own *Save Figure* button should the user wish to export the graph to a local directory.

**Exhibit 10: Population Analysis Tab in Result Window**



* Simulation results in output folder – Besides results displayed in GUI, a set of analytical files will also be stored in the user-specified output folder (by default *./output*) should user have more customized analytical needs. **Exhibit 11** shows an example list of output files. Files generated from the same simulation run are stored in the same subdirectory */output\_[yyyymmdd]\_[hhmmss]\_[header]* where *yyyymmdd* and *hhmmss* respectively indicate the date and time stamp of model execution, and *header* is a unique identifier for parallel simulation under comparison, such as *main simulation*, *comparison 1*, etc.

**Exhibit 11: Files in Output Directory**



* + For each simulation header (which corresponds to a folder in the output directory), output files include the following
  + A master post-simulation ACS state PUMS data file – This is a dataset that contains all eligible ACS workers in the state chosen, with new columns generated from simulation attached.
  + A meta-data file that stores program parameters – This file allows user to keep track of the program parameters employed by the model for simulation.
  + A program cost summary file - This file summarizes program outlay across all six leave types and the total, as well as 95% confidence interval bounds for each outlay estimate.
  + A data file of ACS persons with key variables from the Administrative Budget Financing (ABF) module output.
  + An ABF summary file including key population level estimates from the ABF module output.