## Farm to Facility Model Guide to Files

### Python Setup

The first step to being able to work the model is to get python 3 and an IDE for your computer.

I believe the best way to utilize this model is through the Spyder IDE. Why? Well, this IDE allows us to explore variables in the computer, like R studio. This similarity makes it great for the development of a process model. Other IDEs like Google colab and Jupiter notebook are great for creating notebook reports.

### Anaconda Download

We will run python throughout anaconda. This is a python distribution platform. It allows us to access multiple programs such as Spyder to use python with.

Download and Install anaconda: <https://www.anaconda.com/products/distribution>

Once downloaded open the anaconda navigator and make sure Spyder is installed. Once you are in you can start using the Spyder IDE

### Model Important files

Important files will be in italics and bolded

#### MainModel3z

* This file is the main model file. It contains the main loop function that runs the process model.
* It is split into multiple sections
* The first section is running the data collection data frames from ***Dictionariez.py***
* The second section is the iteration module, in this module everything that occurs in the model is a linear flow of events as in a process.
  + PRCC Sensitivity analysis. Optional section for changing inputs when the PRCC sensitivity analysis is running. Options of ON and OFF can be changed in the ***SCInputz.py*** file
* Data frame Setup and Cont Scenarios
  + This first section creates the main model data frame (the field). The arguments for this specific data frame are in the ***InFunz.py*** file
  + Calendar

    Description automatically generated with medium confidence
  + Screenshot of the main data frame. This data frame will be the one updated as the model progresses. Other data frames mentioned earlier will collect data from this data frame to create other outputs such as contamination progression.
* The next step is to add the background contamination by using the *F\_systematic\_C* function from ***ContScen.py***
* The same is done to contaminate the field. The same function is used but the clustering parameters will change based on the contamination scenario
* Subsequently, you keep going down the document and encounter all the steps that define the process.
  + Microbial die-off is calculated as two sections
  + From Contamination event to PH Sampling
  + And from PH Sampling to Harvest.
* The processes continue. Each function generally takes in a data frame and returns a modified data frame or a list of data frames.

#### Funz

* One of the most important files of the document
* Is home to most functions used for this model
* Not all functions are used in the model, the functions that are used in the model are those solely established in the ***MainModel3z.py*** file
* Some functions here were created to try things out. I kept them in there because they are still useful functions in case, we want to develop something new in the future.

#### Dictionariez

* This module contains the basics from the creation of the data collection data frames
* First contains the name of the columns
* Second, it has functions that create data frames and collect data from these data frames.

#### ContScen

* These documents contain functions to contaminate the main data frame in different ways. The main function used here is the F\_systematic\_C. the advantage of this function is that can create multiple-sized clusters throughout the field. For the analysis, this was the only contamination function used.
* All other functions here work, but they were not used in the analysis.

#### InFunz

* This file serves on purpose, to store the function F\_InDF that creates the main data frame that will be modified through the model.

#### Inputz

* This file contains most of the stationary inputs that the model needs.
  + Contamination Scenario Inputs
  + Some other functions that need to be in this file for the creation of other intermediate inputz
  + For example, the function that defines the chlorine levels throughout the wash.

#### SCInputz

* This file contains fixed inputs for the model
* Sampling plan characteristics
* Iteration number
* Field size
* Weight
* Etc.

### Running the scenario analysis:

#### MarginalGains\_Baseline. Py contains the file that runs the analysis

* This file contains a function named scenario\_function.
* In this function interventions and sampling strategies, and contamination scenarios can be tuned to generate any combination of scenarios.
* To change the number of iterations that the model will run go to SCInputz.py, and change the parameter “N\_Iterations”. In a core i5, 32 Gb. ram computer 10,000 iterations will take approximately 1.4 hrs. per scenario.
* This function generated three outputs
  + A data frame: with important outputs collected from the process:
    - CFU accepted/rejected each sampling plan
    - Weight accepted/ rejected for each sampling plan
  + A data frame with contamination progression for every iteration throughout the process
  + A data frame with the prevalence of contamination throughout the process.
* Then there is a chunk with important functions used to produce the output data frames used for the manuscript
* The chunk titled “Effect of Individual Interventions” runs scenarios for the effect of individual interventions
* The chunk titled “Running the scenario Analysis” runs the scenario analysis for the 147 combinations from the manuscript. If you run these one by one this will take days to run.
* Subsequent chunks create output data frames used for each analysis. Saves the CSV files

#### Final plots in R

* From the CSV files obtained from marginalgains\_baseline.py.
* Different chunks create plots and tables found in the analysis.
* Each chunk has a title with the different outputs it creates.

#### Sensitivity Analysis.py

* Simple code that creates the outputs of the PRCC sensitivity analysis

#### Sensitivity Analysis.rmd

* R markdown file with code that runs the PRCC sensitivity analysis

#### Factor Sensitivity Analysis.py

* Runs the factor sensitivity analysis.
* Has a main function that creates the same outputs as the previous analysis
* Chunk running all the different analysis combinations
* Chunk creating the output that will be used to create plots.

## Run the model piece by piece

If you want to see each part of the model in action to assess the functions, to do a basic knowledge of python is recommended for troubleshooting. This is raw code is not optimized for user interaction.

follow these instructions:

* Open the file MainModel3z.
* Run Chunk 1 libraries
  + Make sure to add the path of the folder where this and all the other files are to the paths.
* Run chunk 2 libraries
  + If the path was setup correctly this chunk should run without any errors
* Go to” chunk 3 the process”
* Run lines 47-57: this loads data frames that track intermediate outputs through the process
* In line 61, remove the # sign from line #i = 1. You can use 1 or any number you want, it just sets the seed.
* Run lines 61-65: this will set up the iteration number we are running.
* Skip all the way to line 108 “BEGINNING OF THE MODEL”
* Step 0:
  + creates the total mass and sets up background contamination
  + you can run this whole chunk.
  + You will be tracking the “df” data frame. You can see this in variable explorer.
* Step 1: preharvest
  + This section applies to die-off at preharvest
  + Conducts preharvest sampling if ON
  + Rejects product
  + Adds contamination if contamination happens after preharvest sampling.
  + Each function is labeled with comments:
* Step 2: Harvest
  + Computes dieoff
  + Computes sampling
  + Rejects based on sampling
  + Comments before every function
* Step 3: Receiving
  + Growth between Harvest and Precooling
  + Precooling steps
  + Growth during receiving storage at the new temperature
  + Palletization: adds pallets to df based on weight per pallet
  + Receiving sampling
* Step 4 value addition
  + Gb2. Is the list of data frames: splits the main data frame into 4 data frames to simulate 4 processing lines in the system
  + Preliminary wash, based on efficiency
  + Shredding cross-contamination has a sanitation option built-in
  + Conveyor belt cross-contamination has a sanitation option built-in
  + Washing
    - Makes lists into data frames that are split into washing rates 100 lb/min
    - Performs washing based on washing function Funz.F\_Washing\_ProcLines3
  + Shaker table
  + Dewatering centrifuge
* Step 5 packing and mixing
  + The product is packed into 5 lb packs
  + FPS happens here
  + Filtering out the rejected product
* Step 6 post-processing steps
  + Growth or die off during post-processing storage
  + Transportation to customer
  + Customer sampling
  + Rejection rule
* Step 7
  + Collecting the process outputs.

For reference. Each function contains a source file for example

The sampling function is in yellow, the source file is in red: Funz.F\_Sampling\_2

* To get to the source file where the function is defined go to the Funz.py document.
* Look for the F\_Sampling\_2 function to see the details of the sampling function.