varargin as a Function Parameter

back to Fan's Intro Math for Econ, Matlab Examples, or Dynamic Asset Repositories

Call Function with Two Parameters and Defaults

Call function below without overriding

```
ff_varargin(1.1, 2)
```

```
fl_a = 1.1000
it_b = 2
mt_data = 3×4
    0.6965    0.5513    0.9808    0.3921
    0.2861    0.7195    0.6848    0.3432
    0.2269    0.4231    0.4809    0.7290
ar_st_colnames = 1×4 string array
"col1"       "col2"       "col3"       "col4"
ar_st_rownames = 1×4 string array
"row1"       "row2"       "row3"       "row4"
st_table_name =
"Table Name"
it_table_ctr = 1021
```

Override Subset of Varargin

```
rng(789);
mt_data_ext = rand(5,2);
ar_st_colnames = ["col1", "col2"];
ar_st_rownames = ["row1", "row2", "row3", "row4", "row5"];
ff_varargin(param_map, support_map, mt_data_ext, ar_st_colnames, ar_st_rownames);
```

```
fl_a =
  Map with properties:

       Count: 2
     KeyType: char
   ValueType: any
it_b =
  Map with properties:

       Count: 1
     KeyType: char
   ValueType: any
mt_data = 5×2
    0.3233    0.7589
    0.2302    0.0106
    0.7938    0.0247
    0.6244    0.1110
    0.9754    0.5381
ar_st_colnames = 1×2 string array
"col1"       "col2"
ar_st_rownames = 1×5 string array
"row1"       "row2"       "row3"       "row4"       "row5"
st_table_name =
"Table Name"
it_table_ctr = 1021
```

Function with varargin as Inputs

Basic default structure with varargin.

```
function ff_varargin(fl_a, it_b, varargin)
% This is an example of how to use varargin:
% 1. includes array matrix
% 2. includes array
% 3. includes scalar
% 4. includes string
% 5. includes cell array

%% Catch Error
cl_params_len = length(varargin);
if cl_params_len > 5
    error('ff_mat2tab:TooManyOptionalParameters', ...
          'allows at most 5 optional parameters');
end

%% Default Folder Parameters
% by default all go to Sandbox folder with sub folders by dates
rng(123);
mt_data = rand(3,4);
% String array requires double quotes
ar_st_colnames = ["col1", "col2", "col3", "col4"];
ar_st_rownames = ["row1", "row2", "row3", "row4"];
% Others
st_table_name = "Table Name";
it_table_ctr = 1021;
cl_params = {mt_data ar_st_colnames ar_st_rownames ...
                  st_table_name it_table_ctr};

%% Parse Parameters
% numvarargs is the number of varagin inputted
[cl_params{1:cl_params_len}] = varargin{:};
% cell2mat(cl_params(1)) works with array
mt_data = cell2mat(cl_params(1));
% The structure below works with cell array
ar_st_colnames = cl_params{2};
ar_st_rownames = cl_params{3};
% Others
st_table_name = cl_params{4};
it_table_ctr = cl_params{5};

% Build Basic Matlab Table
% Suppose we want to store matrix results in a table,
% there are Q columns and N rows, The Q columns each is a different variable.
fl_a
it_b
mt_data
ar_st_colnames
ar_st_rownames
st_table_name
it_table_ctr
```

```
end
```