1 Taxonomy of UI Inputs

UI Version 1.0 is designed to handle the following inputs related to the system behavior and analytics. Input is to be arranged in six separate csv sheets as described in various sections below. These sheets are to be named as below:

1. System Preamble: **CS-SP**
2. System Parameters: **CS-SR**
3. System Dynamics: **CS-SD**
4. Initial Conditions: **CS-SI**
5. Simulation Performance: **CS-SF**
6. System Analytics: **CS-SA**

The above identifiers are also used for cross-referencing variables appearing in these sheets.

2 System Preamble: CS-SP

The purpose of the preamble csv sheet is to set up the simulation by defining the nature and size of the system. It contains:

- Numerical values of some key parameters that help set up the simulation, e.g. size of the dynamic system and parameters of time integration.
- Names and sizes of all variables related to the dynamic system that appear in the remaining sheets (CS-SR, CS-SD, CS-SI, CS-SF and CS-SA).

The preamble must provide the data listed in Table (2).

A. Dynamic System Type. This input, identified as **SysType** in the preamble defines the form of the dynamic system and the way it is to be entered by the user. The current version allows dynamic system types shown in the list below. The identifier associated with each system type is identified below, as well as in Table (2). Additional dynamic system categories will be added in future versions of the software.

1. Ordinary Differential Equations (ODEs), specified as direct (typed) input at the User Interface.
   Identifier: ODE_A
2. Ordinary Differential Equations (ODEs), specified in a csv sheet.
   Identifier: ODE_B
3. Ordinary Differential Equations (ODEs), specified in a MATLAB©or Python function.
   Identifier: ODE_C
4. Ordinary Difference Equations (ODFs), specified as direct (typed) input at the User Interface.
   Identifier: ODF_A
5. Ordinary Difference Equations (ODFs), specified in a csv sheet.
   Identifier: ODF_B
6. Ordinary Difference Equations (ODFs), specified in a MATLAB©or Python function.
   Identifier: ODF_C

7. Stochastic Differential Equations (SDEs), specified as direct (typed) input at the User Interface.
   Identifier: SDE_A
8. Stochastic Differential Equations (SDEs), specified in a csv sheet.
   Identifier: SDE_B
9. Stochastic Differential Equations (SDEs), specified in a MATLAB©or Python function.
   Identifier: SDE_C
10. Hidden State Black Box (HBOX), specified as a function or package with predefined inputs and outputs. No "state" is specified in this option. The user provides a system simulation package that receives defined inputs and provides defined outputs to the uncertainty forecasting software in the backend. All forecasting analytics is to be performed in terms of these "observable" inputs and outputs. The actual state of the system is either too complicated or proprietary, and is never revealed (it is *hidden*).
    Identifier: HBOX

Validation checks: Input must match names provided in Table (2), case sensitivity not required.
Cross-Referencing: Some inputs to follow in the preamble are required for only a particular type of dynamic system. If not dealing with such a system, the fields next to such inputs must be left blank or any provided data must be ignored.

B. System Dimensions. There are four inputs possible here, depending on the choice of dynamic system under **SysType**. The following inputs are included:

1. Dimensionality of State. This input is required for system types ODE_{A,B,C}, ODF_{A,B,C}, SDE_{A,B,C} and defines the number of state variables in the dynamic system. It is not needed for dynamic system type HBOX and any provided input must be ignored. This input must be named *NS* in the preamble. It identifies the number of independent states of the dynamic system. It assumes a positive integer value.
Validation checks: $NS \in \mathbb{Z}^+$[1]
Cross-Referencing: No direct appearance expected in other sheets. This input assists in counting state variables.

2. Dimensionality of Noise Terms. This information is required for system type SDE_{A,B,C}. It is not needed for other system types and will be ignored if a value is provided. This input must be named *NN* in the preamble.
Validation checks: $NN \in \mathbb{Z}^+$. Mandatory input only for system type SDE_{A, B, C}.
Cross-Referencing: No direct appearance expected in other sheets. This input assists in counting the number of noise terms driving systems of the type SDE_{A,B,C}.

3. Dimensionality of Inputs. This input is required for system type HBOX. It is not needed for other system types and will be ignored if a value is provided. This input must be named *NU* in the preamble.
Validation checks: $NU \in \mathbb{Z}^+$. Mandatory input only for system type HBOX.
Cross-Referencing: No direct appearance expected in other sheets. This input assists in counting input variables for system type HBOX.

[1] Set of positive integers

4. Dimensionality of Outputs. This information is required for system type HBOX. It is not needed for other system types and will be ignored if a value is provided. This input must be named NY in the preamble.
Validation checks: $NY \in \mathbb{Z}^+$, Mandatory input only for system type HBOX.
Cross-Referencing: No direct appearance expected in other sheets. This input assists in counting output variables for system type HBOX.

C. System Variable Names. This set of inputs defines the names of key variables appearing in the dynamic system. Depending on the nature of dynamics, some inputs may not be needed and can be left empty. The following information is requested:

1. State Name. This input, identified as **SName** in the preamble defines the letter name used for state variables, e.g. SName = x. If $NS = 3$, the three states are identified as $x1$, $x2$ and $x3$. The user must only identify the "parent" state name (in this example, that's "x"). The derivative names, i.e. $x1, x2, \ldots, xNS$ are not to be defined. Name of state is a required input for dynamic systems of type ODE_{A, B, C}, ODF_{A, B, C} and SDE_{A, B, C}. It is not needed for dynamic system type HBOX and any provided input must be ignored.
Validation checks: No special characters. Choice of symbols is from among the 52 letters of the English alphabet (26 lower case, 26 upper case).
Cross-Referencing: Each state (every element of the set $\{x1,\ x2, \ldots, xNS\}$) must appear in **CS-SD**. Partial state may appear in **CS-SI**. Partial state will appear in **CS-SF**, **CS-SA**.

2.. Time Name. This input, identified as **TName** in the preamble defines the letter name to be used for "time", e.g. TName = t. This name is not case sensitive. This input is mandatory for all dynamic system types.
Validation checks: No special characters. Choice of symbols is from among the 52 letters of the English alphabet (26 lower case, 26 upper case). This name should be unique, i.e. not used for another item, e.g. State Name.
Cross-Referencing: This variable may appear in **CS-SD**, although it is not necessary. It may also appear in **CS-SF**, **CS-SA**.

3. Input Variables Names. This information, identified as **UName** in the preamble defines the names used for all input variables. It is only a mandatory input for dynamic systems of type HBOX and will be ignored for othersystem types. All input names must appear in separate fields of the UName row, for example

$UName$	$Temp$	$Speed1$	$Speed2$

The above example assumes that $NU = 3$. This input is mandatory only for dynamic systems of type HBOX and will be ignored for other types.
Validation checks: No special characters. Choice of names must be unique and not conflict with states, time, or other parameters. Mandatory input only for system type HBOX.
Cross-Referencing: The full list of input names must appear in the description of the HBOX dynamic system: see **CS-SD**.

4. Output Variables Name. This information, identified as **YName** in the preamble defines the names used for all output variables. It is only a mandatory input for dynamic systems of type HBOX and will be ignored for othersystem types. All output variable names must appear in

separate fields of the YName row, for example

YName	*Pres*14	*Dens*21	*Pres*28	*QFlux*41

The above example assumes that $NY = 4$.
Validation checks: No special characters. Choice of names must be unique and not conflict with states, time, or other parameters. Mandatory input only for system type HBOX. Ignore for other system types.
Cross-Referencing: The full list of output names must appear in the description of the HBOX dynamic system: see **CS-SD**. A partial list of output variables is also expected to appear in sheets **CS-SA** and **CS-SF**.

D. Non-Dimensionalization of System Variables. Some applications require various system variables to be non-dimensionalized before use in simulation and analysis. In this set of preamble inputs, binary information is requested to indicate whether each type of system variable is to be non-dimensionalized or not. Only some of the following information is mandatory, depending on the specification of the dynamic system.

1. State Non-Dimensionalization. This binary input, named **NonDimS**, indicates whether states must be non-dimensionalized before they are used in quantities of interest or other analytic metrics. NonDimS = 1 implies that state non-dimensionalization *is* needed, while NonDimS = 0 implies that it is not needed. This input is mandatory for dynamic systems of type ODE_{A, B, C}, ODF_{A, B, C} and SDE_{A, B, C}. It is not needed for dynamic system type HBOX and any provided input must be ignored.
Validation checks: This is a binary scalar input that can either be 0 or 1.
Cross-Referencing: If **NonDimS** = 1, non-dimensionalization parameters for the state will be defined in the parameter sheet **CS-SR**.

2. Time Non-Dimensionalization. This binary input, named **NonDimT**, indicates whether time must be non-dimensionalized before being used in quantities of interest or other analytic metrics. NonDimT = 1 implies that time non-dimensionalization *is* needed, while NonDimT = 0 implies that it is not needed. This input is mandatory for all dynamic systems.
Validation checks: This is a binary scalar input that can either be 0 or 1.
Cross-Referencing: If **NonDimT** = 1, non-dimensionalization parameters for the state will be defined in the parameter sheet **CS-SR**.

3. Input Non-Dimensionalization. This binary input, named **NonDimU**, indicates whether input variables must be non-dimensionalized before being used. NonDimU = 1 implies that input non-dimensionalization *is* needed, while NonDimU = 0 implies that it is not needed. This information is mandatory for dynamic systems of type HBOX. This input is not needed for all other dynamic system types and any information, if provided, will be ignored.
Validation checks: This is a binary scalar input that can either be 0 or 1.
Cross-Referencing: If **NonDimU** = 1, non-dimensionalization parameters for the input variables will be defined in the parameter sheet **CS-SR**.

4. Output Non-Dimensionalization. This binary input, named **NonDimY**, indicates whether output variables must be non-dimensionalized before being used. NonDimY = 1 implies that output non-dimensionalization *is* needed, while NonDimY = 0 implies that it is not needed. This information is mandatory for dynamic systems of type HBOX. This input is not needed for all other

dynamic system types and any information, if provided, will be ignored.
Validation checks: This is a binary scalar input that can either be 0 or 1.
Cross-Referencing: If **NonDimY** = 1, non-dimensionalization parameters for the input variables will be defined in the parameter sheet **CS-SR**.

E. Time Unit: Dynamics. Define unit of time used in system dynamics. This is different from time unit used for system analytics. This input must be identified as **TDunit** and is allowed to take on self-descriptive unit names as shown in Table (2).
Validation checks: Input must match names provided in Table (2), case sensitivity not required.
Cross-Referencing: None required.

F. Time Unit: Analytics. Define unit of time used for presentation of system analytics. This input must be named **TAunit** and is allowed to assume the same self-descriptive unit names as the input **TDunit.** Note that **TDunit** and **TAunit** are allowed to be assigned differently.
Validation checks: Input must match names provided in Table (2), case sensitivity not required.
Cross-Referencing: None required.

G1. Maximum Integration Time Step. Define maximum allowable time step of integration for system dynamics. This input must be identified as **TSmax** and must be a scalar value. If this variable is irrelevant, use one of following options:

1. Set it to infinity, as "inf".
2. Define **TSmax** without any data associated with it.

Validation checks: This scalar input must be strictly positive and real ($\in \Re^{+}$). Also, if both are specified, its numerical value must be strictly greater than the numerical value of **TSmin**.
Cross-Referencing: This input is relevant only for dynamic systems of type ODE_{A, B, C}. If system is not one of these type, it must either be be empty, or any provided data must be ignored (with an appropriate warning message).

G2. Minimum Integration Time Step. Define minimum allowable time step of integration for system dynamics. This input must be named **TSmin** and must be a scalar value. If this input is irrelevant to the application, use one of following options:

1. Set it to infinity, as "0".
2. Define **TSmin** without any data associated with it.

Validation checks: This scalar input must be positive and real (≥ 0). Also, if both are specified, its numerical value must be strictly less than the numerical value of **TSmax**.
Cross-Referencing: This input is relevant only for dynamic systems of type ODE_{A, B, C}. If system is not one of these type, it must either be be empty, or any provided data must be ignored (with an appropriate warning message).

G3. Time Step. This scalar input is relevant only for dynamic system types ODF_{A, B, C} and SDE_{A,B,C}. However, even if the dynamic system is one of these types, this input is not required. The integration time step can be defined as one of the parameters of the dynamic system (see Items **K.** and **L.** below).

G4. Simulation End Time. This scalar input is not needed. Its value is extracted from Item **I.** below (time vector for analytics).

~~[H. Time Vector for Analytics.] This vector defines specific time instances at which quantities of interest and/or states of health must be monitored. This is different than the integration time step. User may either provide an explicit row vector containing time instances, or, a triple, containing the initial time, frequency and final time, as follows: initial:frequency:final. This input must be identified as **TAvec** in the preamble.~~
~~Validation checks: Each element of input must be strictly positive and real ($\in \Re^+$).~~
~~Cross-Referencing: No mandatory cross-referencing is required. This input may appear in **CS-SF** and/or **CS-SA**.~~

I. Parameter Types. System dynamics usually contain parameters that capture various aspects of the underlying physics. Provide the list of parameter types that appear in system dynamics. This input is identified by the header **ParamType**. It is not required to provide a count of the number of parameters of each type. The user must only define the types of parameters that will appear in input sheets **CS-SD**, **CS-SA**, and **CS-SF**. Permitted parameter types are defined in Table (2). Below is a description of each parameter type identifier:

- **param_c**: constant parameter. The numerical value of such a parameter does not change over time.
- **param_sd**: state-dependent parameter. The numerical value of such a parameter depends on the current value of the state through an explicit functional form. This category includes time-varying parameters, i.e. parameters that change as an explicit function of time.
- **param_f**: parameter defined through a function call. Each parameter name of this type must be preceded by an "@" symbol (function identifier). The inputs of this function may include time, state variables and other parameters and these inputs must be defined. See Sec.(4).
  Cross-Referencing: User must provide a function file associated with each parameter defined under this type.
- **param_tab**: parameter is specified in tabular form dependent on time, state variables and other parameters.
  Cross-Referencing: User must provide a table associated with each parameter defined under this type. In some instances, there may be more than one table needed for each parameter name. E.g. let "k" be a matrix parameter of size 2×3, defined under parameter type "param_tab". Then, $2 \times 3 = 6$ tables must be provided to determine the value of each of its components.
- **param_rand**: parameter is specified as a random variable with known distribution (continuous or discrete).

Validation checks: Input must match names provided in the list above, case sensitivity not required. It is possible that no parameter types are defined.
Cross-Referencing: If no parameter types are defined, there cannot be any variables other than system variables (Item **C.**) in **CS-SD**, **CS-SF** and **CS-SA**.

J. Parameter Names and Sizes. Names of all parameters for each parameter type must be defined here. Next to each parameter type, parameter names must appear as a vector of strings. No numerical values are to be provided here. Numerical specification (value, table, function, etc.) will be input in a separate sheet (**CS-SR**: see Sec.(3)). An example is provided below in Table (1). Note that parameter

names are not considered case sensitive.
Each parameter could be scalar or vector valued. The dimensionality of every parameter name must be specified. Size specifications are to be made directly under its name. See Table (1): it is evident that K_1, mu_earth and S are scalars while rho is a 3×1 vector and g is a 2×4 matrix.
Validation checks: Warning display if repeated parameter names appear. Parameter names are case sensitive, e.g. k and K are different; Cd and CD are different. Parameter sizes can only contain strtictly positive integer values ($\in \mathbb{Z}^+$).
Cross-Referencing: These will appear in **CS-SD**, **CS-SF** and/or **CS-SA**. However, each sheet may only contain a partial list of defined parameters.

param_c	K_1	S	C_D	mu_earth
	1	1	1	1
param_sd	g	ref		
	[2,4]	[1,2]		
param_tab	rho			
	[3,1]			
Parameter names are case sensitive.				

Table 1: Naming Parameters and Their Sizes Under Each Parameter Type

Item ID	Item Name	Input Data Type	Taxonomy: Item	Data
A.	**Dynamic system type**	name identifier: string input	SysType	ODE_{A, B, C} ODF_{A, B, C} SDE_{A, B, C} HBOX
B.	**State space dimension**	scalar positive integer	*NS*	Numerical value
	Input space Dimension	scalar positive integer	*NU*	Numerical value
	Output space dimension	scalar positive integer	*NY*	Numerical value
	Process noise Dimension	scalar positive integer	*NN*	Numerical value
C.	**State name**	letter identifier: string input	SName	{letter}
	Time name	letter identifier: string input	TName	{letter}
	Input name	name identifiers: vector string input	UName	{names}
	Output name	name identifiers: vector string input	YName	{names}
D.	**State non-dimensionalization**	binary	NonDimS	0/1
	Time non-dimensionalization	binary	NonDimT	0/1
	Input non-dimensionalization	binary	NonDimU	0/1
	Output non-dimensionalization	binary	NonDimY	0/1
E.	**Time unit: Dynamics**	name identifier: string input	TDunit	sec min hour day
F.	**Time unit: Analytics**	name identifier: string input	TAunit	same options as TDunit
G1.	**Max integration timestep**	scalar real value	TSmax	Numerical value
G2.	**Min integration timestep**	scalar real value	TSmin	Numerical value
H.	**Time vector: Analytics**	scalar positive integer	TAvecLen	Numerical value
I.	**Parameter types**	row vector defining parameter types: string inputs	ParamType	param_c param_sd param_f param_tab param_rand
J.	**Names and Sizes of Parameters for each Defined Parameter Type**	Row 1: Names (vector string input) Row 2: Size of each parameter	param_{id}	{names } 1, or, [m,n]
parenthetical names ({· }) in taxonomy are not case sensitive. Rest must appear as shown.				

Table 2: Taxonomy for System Preamble

3 System Parameters: CS-SR

The purpose of the Parameter sheet, **CS-SR** is to provide numerical specification of each parameter defined in the preamble. These parameters are expected to appear in System Dynamics (**CS-SD**), Initial Conditions (**CS-SI**), Simulation Performance (**CS-SF**), System Analytics (**CS-SA**), or even within other parameters defined in System Parameters (**CS-SR**). As described in the preamble, there are six possible types of parameters. Numerical specification of each is described next. For each type of parameter that must be defined, its identifier must first be specified. All parameters of a common type must be grouped together, as shown in examples below.

Constant Parameters: The category name identifier for this parameter type is **param_c**. All parameters of this type must be defined in separate rows. An example is provided below.

param_c	$K1$	1	0.6
	$S2$	11.5	2.4
		5.1	1.9
		0.5	12.4
	rho	0.671	

(Constant Parameters)

Validation checks: All numerical entries must be real numbers. There should be no "broken matrices". For example, when a matrix input is expected, all rows must have the same number of columns.
Cross-Referencing: All parameter names defined under this parameter type in the preamble (**CS-SP**) must appear in the list. Their sizes should also match the size defined in the preamble.

Time/State Dependent Parameters: The category name identifier for this parameter type is **param_sd**. All parameters of this type must be defined in separate rows. An example is provided below.

param_sd	g	$-mus*x1/r^3$	$-mus*x2/r^3$	$-mus*x3/r^3$
		$-mue*x1/r^3$	$-mue*x2/r^3$	$-mue*x3/r^3$
	r	$sqrt(x1^2+x2^2+x3^2)$		
	rho	$rho0*exp(-h/h0)$		

(State Dependent Parameters)

Validation checks: Terms contained in the parameter expressions must only include from among the following four categories of terms:

1. System variables (states, inputs or outputs),
2. Time variable, t,
3. Other parameters, whose names, types and sizes were defined in the preamble, **CS-SP**.
4. Special terms with predefined meanings, e.g. "pi", "sin", "cos", "exp", ∧, "sqrt", etc.

For instance, in the definition of "rho" in above example, the variable h is intended to be a system variable (input) and will be found in the preamble. Similarly, in this example, the variables $rho0$ and $h0$ are intended to be constant parameters. Also, there should be no "broken matrices". For example, when a matrix input is expected, all rows must have the same number of columns.
Cross-Referencing: All parameter names defined under this parameter type in the preamble (**CS-SP**) must appear in the list. Their sizes should also match the size defined in the preamble.

Function Call Parameters: The category name identifier for this parameter type is **param_f**. All parameters of this type must be defined in separate rows. For each parameter, the function to be called must be named, starting with the function identifier, "@". In the cells to follow, input variables for the function name must be defined. These input variable names can include system variables (state, time, inputs, outputs) and/or other parameters. Note that for each parameter named under this category, only one row of input is needed, irrespective of its size. The (matrix) size of the parameter was defined in the preamble. An example is provided below.

param_f	*CTemp*	*@CTemp*	*x*1	*t*	*mue*
	M	*@MachN*	*T*	*R*	*Temp*

(Function Call Parameters)

Validation checks: Each parameter name must have eaxctly one row of input. Immediately following the parameter name under this categorry (cross-reference with preamble), a function name must appear preceded by the "@" symbol. The function name may not be the parameter name, e.g. see the second example above (parameter name: M, function name: $@MachN$). Note that it is possible that the parameter $CTemp$ was defined in the preamble **CS-SP** being of size $[1,3]$ (or another size), and yet only one row of input is needed, as described in the example above. The input variables for each row can only include system variable names (states, time, inputs, outputs) and/or other defined parameters.
Cross-Referencing: All parameter names defined under this parameter type in the preamble (**CS-SP**) must appear in the list. As mentioned above, there is no need to match the size of input here with their size defined in the preamble.

Tabular Parameters: The category name identifier for this parameter type is **param_tab**. Each parameter of this type has a table associated with it. The table(s) can either be explicitly defined in the csv sheet or provided as a separate data file. Examples of each type of definition are given below.

Consider first a tabular parameter with an explictly listed table in the csv sheet. An example is provided below. Following the name of the parameter, the cell that follows immediately should contain the name of the independent variable. The next cell again should carry the name of the parameter. The independent variable can only include system variables (states, time, input, output) and/or other parameter names. Note that only two-column tables can be defined this way. Tables that have more than one independent variable are significantly more difficult to interpolate and must be defined as a function-parameter. In the rows to follow, numerical values in the table must appear under appropriate column names. Note that the numerical values under the independent variable must be ordered in ascending or descending order.

If the parameter is vector valued, the index of the parameter should be identified under its name. For example, is $Coff1$ is size $[1,2]$, there are two tables needed, followed by one another, with the index identified directly under the parameter name.

param_tab	$Coff1$	$K1$	$Coff1$
		0.5784	0.4889
		0.7694	1.0347
		1.3499	0.7269
		3.0349	−0.3034
		3.7254	0.2939
		4.0631	−0.7873
		4.7147	0.8884
		6.2050	−1.1471

(Tabular Parameter: Ex 1)

It is possible that some tabular parameters come with table sizes that are very big. Such tables can be specified as data files (help with file types) that contain the table. For parameters of this type, the definition contains the name of the data file, which is to be distinguished by the identified "#". In the cells adjacent to the file name, column headings must be defined. An example is given below. Column headings can include system variables (state, time, inputs, outputs) and/or other parameters. If the parameter is matrix sized, a data file can be named for each element of the matrix, with its index identified as the last entry in the row. See example below.

param_tab	$Coff2$	$\#CTemp1$	T	$Coff2$	$[1,1]$
	$Coff2$	$\#CTemp2$	$x2$	$Coff2$	$[1,2]$

(Tabular Parameter: Ex 2)

It is also possible to have a mixed definition (explicit table and data file), as seen in a simple example below:

param_tab	$Coff3$	K	$Coff3$		
	$[1,2]$	0.5784	0.4889		
		0.7694	1.0347		
		1.3499	0.7269		
		3.0349	−0.3034		
		4.7147	0.8884		
		6.2050	−1.1471		
	$Coff3$	$\#Coff31$	T	$Coff3$	$[1,1]$

(Tabular Parameter: Ex 3)

In the above example, the parameter $Coff3$ was defined of size $[1,2]$ in the preamble, **CS-SP**. Note that parameters that required multiple tables to be used for adjustment (e.g. drag coefficient adjusted for flight Mach Number) must be defined as *function-call parameters*. Similarly, piece-wise defined parameters must also be defined as function-call parameters.

Validation checks: For explicitly defined tables, only two columns are allowed: one for an independent variable and the second for the named parameter. The data under the independent variable must be arranged in order, either ascending or descending. Data file names must be preceded by the "#" symbol. The file-name may not be the same as the parameter name, e.g. see the second example above (parameter name: $Coff2$, data-file names: $\#CTemp1$ and $\#CTemp2$). The input variables for each row can only include system variable names (states, time, inputs, outputs) and/or other defined parameters.
Cross-Referencing: All parameter names defined under this parameter type in the preamble (**CS-SP**) must appear in the list. The size of the parameter must match the definition provided in this sheet.

Random Parameters: Random parameters must be specified by their probability distribution. They can be one of many types. See Section (5) for a detailed description. In this version of the Software, random parameters are allowed to assume the forms itemized below. These forms are described in detail in Sec.(5).

A. Standard Probability Distribution: Continuous. The parameter name must be followed by, in the adjacent cell, the name of its standard distribution. In the next rows, the parameters associated with the identified distribution must be defined, in the same manner as in Sec.(5). Note that the size of distribution parameters must be consistent with the vector size of the parameter. Parameters of this type are currently not allowed to be matrices - they can however, be vectors. All currently supported standard random distributions are listed in Sec.(5). An example is given below for a parameter named *CR*:

param_rand	*CR*	*Normal*	
	mean	0	0
	covariance	1.1	0
		0	0.4

(Random Parameter: Ex 1)

Note that from the description above, it is apparant that the vector size of CR is $[2,1]$. Validation checks: All validation checks listed in Sec.(5) must be performed. Parameters of this type are only allowed to be scalars or vectors.

B. Probability Mass Function. The parameter name must be followed by, in the adjacent cell, the keyword identifier **pmf**. In the following rows, a table must be provided that defines the probability mass function of the parameter. Only vector sized random parameters of this type are allowed. If the size of the parameter is $[M,1]$, the pmf table must contain $(M+1)$ columns. The $(M+1)^{th}$ column contains the value of the pmf for each row (realization). Please refer to Sec.(5):**(Item C.)** for validation and other checks.

4 System Dynamics: CS-SD

The system dynamics is spelled out in this csv sheet, which must be named **CS-SD**. The current version of the forecasting software admits continuous and discrete time dynamic systems governed by ordinary differential equations, ordinary difference equations, and stochastic difference equations. Each case is treated below. For purposes of illustration, we assume that the state is named x and time is named t. The following types of dynamic systems can be defined (name identifiers can be cross-referenced with Table (2)).

ODE_A. Ordinary Differential Equations: Direct UI Input. This option allows the user to directly type in ordinary differential equations (ODEs) on the user interface.
Validation checks: *NS* fields must be provided, arranged as rows, as shown below in Eqs.(1). Recall that *NS* was defined in the preamble **CS-SP** as the dimension of the state space. Each row corresponds to one state, in order. I.E., the first row is meant for state $x1$ and the last row for state xNS. Each row must contain the complete expression appearing in the right hand side of the dynamic equation for the corresponding state: for example, compare Eqs.(1) and (2). Letter variables used in these expressions must be consistent with name definitions for the states, parameters and time, as outlined in the preamble. Example:

$$\boxed{x2} \tag{1a}$$

$$\boxed{-k*x1+a*x1\wedge 3+b*x2} \tag{1b}$$

The above input will be interpreted as

$$\dot{x1} = x2 \tag{2a}$$

$$\dot{x2} = -k*x1+a*x1\wedge 3+b*x2 \tag{2b}$$

Therefore, only the right-hand-sides of the dynamic model (Eqs.(2)) are to be input by the user. These inputs are understood to be in order, i.e. first entry ((1a)) corresponds to $x1$ and the second entry ((1b)) corresponds to $x2$.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODE_A.

ODE_B. Ordinary Differential Equations: CSV. In this option, the ordinary differential equations are to be provided as a csv input, containing the right-hand-sides of each ODE in order.
Validation checks: There must be *NS* expressions. Each expression must appear in a separate row. Each expression is supposed to represent the right hand side of the ODE system. The expressions will be assumed to be in order; i.e., it will be assumed that the first expression corresponds to $x1$ and the last to xNS. All terms contained in the expressions include only the following four categories of terms:

1. State variables: $x1$, $x2$, ..., xNS (assuming the letter identifier for the state is x),
2. Time variable, t
3. Parameters, whose names, types and sizes were defined in the preamble, **CS-SP**.
4. Special terms with predefined meanings, e.g. "pi", "sin", "cos", "exp", $\wedge$, "sqrt", etc.

Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODE_B.

ODE_C. Ordinary Differential Equations: Function Call. The user is required to provide the name of the function that contains the system dynamics, along with all input variables required by this function. Therefore, only one row of input is needed. The first field of this row contains the function name.

The function name must be preceded by an "@" symbol (function identifier). All remaining fields contain inputs to this function, one input name in each field. The input variables must be arranged in the following order:

$$(t,\ x,\ \text{parameter1},\ \text{parameter2},\ \ldots)$$

where, parameters were defined in the preamble **CS-SP**. See example in Eq.(3).

$$\boxed{@lorenz96}\ \boxed{t}\ \boxed{x}\ \boxed{mu} \tag{3}$$

Validation checks: The function name must be preceded at the "@" symbol. Following this identifier, function name can only contain upper/lower case letters and numbers. The sequence of input variables must begin with time, followed by the state name, and then relevant parameters. It is possible that there are no input parameters specified. Only time and state names are mandatory inputs.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODE_C.

ODF_A. Ordinary Difference Equations: Direct UI Input. This option allows the user to directly type in ordinary difference equations (ODFs) on the user interface.
Validation checks: *NS* fields must be provided, arranged as rows, as shown below in Eqs.(4). Recall that *NS* was defined in the preamble **CS-SP** as the dimension of the state space. Each row corresponds to one state, in order. I.E., the first row is meant for state $x1$ and the last row for state xNS. Each row must contain the complete expression appearing in the right hand side of the dynamic equation for the corresponding state: for example, compare Eqs.(4) and (5). Letter variables used in these expressions must be consistent with name definitions for the states, parameters and time, as outlined in the preamble. Example:

$$\boxed{DT*x2} \tag{4a}$$

$$\boxed{DT*(-k*x1+a*x1\wedge 3+b*x2)} \tag{4b}$$

The above input will be interpreted as

$$x1(i+1) = x1(i)+DT*x2(i) \tag{5a}$$

$$x2(i+1) = x2(i)+DT*(-k*x1(i)*+a*x1(i)\wedge 3+b*x2(i)) \tag{5b}$$

Therefore, only the right-hand-sides of the dynamic model (Eqs.(5)) are to be input by the user. These inputs are understood to be in order, i.e. first entry ((4a)) corresponds to $x1$ and the second entry ((4b)) corresponds to $x2$.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODF_A.

ODF_B. Ordinary Difference Equations: CSV. In this option, the ordinary difference equations are to be provided as a csv input, containing the right-hand-sides of each ODF in order.
Validation checks: There must be *NS* expressions. Each expression must appear in a separate row. Each expression is supposed to capture the right hand side of the ODF system. The expressions will be assumed to be in order; i.e., it will be assumed that the first expression corresponds to $x1$ and the last to xNS. All terms contained in the expressions include only the following four categories of terms:

1. State variables: $x1$, $x2$, …, xNS (assuming the letter identifier for the state is x),

2. Time variable, t
3. Parameters, whose names, types and sizes were defined in the preamble, **CS-SP**.
4. Special terms with predefined meanings, e.g. "pi", "sin", "cos", "exp", $\wedge$, "sqrt", etc.

Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODF_B.

ODF_C. Ordinary Difference Equations: Function Call. The user must provide the name of the function that contains the system dynamics, along with all input variables required by this function. Therefore, only one row of input is needed. The first field of this row contains the function name. The function name must be preceded by an "@" symbol (function identifier). All remaining fields contain inputs to this function, one input name in each field. The input variables must be arranged in the following order:

$$(t,\ x,\ \text{parameter1},\ \text{parameter2},\ \ldots)$$

where, parameters were defined in the preamble **CS-SP**. See example in Eq.(6).

$$\boxed{@abcdyn}\ \boxed{t}\ \boxed{x}\ \boxed{G}\ \boxed{P} \tag{6}$$

Validation checks: The function name must be preceded at the "@" symbol. Following this identifier, function name can only contain upper/lower case letters and numbers. The sequence of input variables must begin with time, followed by the state name, and then relevant parameters. It is possible that there are no input parameters specified. Only time and state names are mandatory inputs.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to ODF_C.

SDE_A. Stochastic Differential Equations: Direct Input This option allows the user to directly type in stochastic differential equations (SDEs) on the user interface. The user must provide the right-hand-sides of each SDE in the system, in order.
Validation checks: NS rows must be provided for input. Recall that NS is the dimension of the state space, i.e. the number of independent states. Each row must contain $(NN+1$ fields, where NN is the dimensionality (number) of noise terms: see Sec.(2). The first field is for the "deterministic" part of the dynamics, i.e. terms that do not contain noise forcing. Note that this field is allowed to contain random parameters (i.e. parameters of type "param_rand"). Fields 2 through $(NN+1)$ contain coefficients/expressions that multiply each of the NN noise forces, in order. See example below. Letter variables used for states and various parameters must be consistent with those defined in the preamble.

Consider the following example with $NS=3$ and $NN=2$: resulting in 3 rows, each with 3 fields:

$$\boxed{x2}\ \boxed{0}\ \boxed{0} \tag{7a}$$

$$\boxed{-k*x1+a*x1\wedge 3+b*x2}\ \boxed{g}\ \boxed{0} \tag{7b}$$

$$\boxed{-x2+Q*x1+b*x2*x1}\ \boxed{h}\ \boxed{1} \tag{7c}$$

The above input will be interpreted as

$$dx1\ =\ x2dt \tag{8a}$$

$$dx2 = (-k*x1 + a*x1\wedge 3 + b*x2)dt + g*d\xi_1 \quad (8b)$$
$$dx3 = (-x2 + Q*x1 + b*x2*x1)dt + hd\xi_1 + d\xi_2 \quad (8c)$$

Therefore, only the right-hand-sides of Eqs.(8) are to be input by the user. The inputs are understood to be in order, i.e. row corresponds to state variable $x1$ and NS^{th} row corresponds to state variable xNS. Similarly, the second field in any given row depicts the coefficient of the noise term ξ_1, and the $(1+NN)^{\text{th}}$ field depicts the coefficient of the noise term ξ_{NN}.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to SDE_A.

SDE_B. Stochastic Differential Equations: CSV. In this option, the SDE must be defined in a csv sheet. The user must provide the right-hand-sides of each SDE in the system. Therefore, there must be NS rows of input, one for each state, in order. Each row must contain $NN+1$ fields, such that the first field contains the "deterministic part" of the dynamic equation and the remaining fields contain the coefficients of each of the noise forcing terms, in order. See the example above for system type SDE_A. The expressions in each field of this example should appear in the same manner, except, now, in a csv sheet.
Validation checks: NS rows must be provided. Recall that NS is the dimension of the state space, i.e. the number of independent states. Each row must contain $(NN+1$ fields, where NN is the dimensionality (number) of noise terms: see Sec.(2). The first field in each row is for the "deterministic" part of the dynamics, i.e. terms that do not contain noise forcing. Note that this field is allowed to contain random parameters (i.e. parameters of type "param_rand"). Fields 2 through $(NN+1)$ contain coefficients/expressions that multiply each of the NN noise forces, in order. Letter variables used for states and various parameters must be consistent with those defined in the preamble.

Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to SDE_B

SDE_C. Stochastic Differential Equations: Function Call. The user must provide the name of the function that contains the system dynamics, along with all input variables required by this function. Therefore, only one row of input is needed. The first field of this row contains the function name. The function name must be preceded by an "@" symbol (function identifier). All remaining fields contain inputs to this function, one input name in each field. The input variables must be arranged in the following order:

$$(t,\ x,\ \text{parameter1},\ \text{parameter2},\ \ldots)$$

where, parameters were defined in the preamble **CS-SP**. See example in Eq.(9).

$$\boxed{@logistic}\ \boxed{t}\ \boxed{x} \quad (9)$$

Validation checks: The function name must be preceded at the "@" symbol. Following this identifier, function name can only contain upper/lower case letters and numbers. The sequence of input variables must begin with time, followed by the state name, and then relevant parameters. It is possible that there are no input parameters specified. Only time and state names are mandatory inputs.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to SDE_C.

HBOX. Hidden State Black Box Model. CSV. In this option, the user provides a black box model with inputs and outputs. There are no states involved. If this system type was chosen, the user must provide three rows of information:

1. The first row contains the name of the function/package. The function name must be preceded by an "@" symbol (function identifier).
2. The second row provides the names, in order, of the input variables to the package. These input names were defined in the preamble, **CS-SP**.
3. The third row provides the names, in order, of the output variables for the system package. These names were defined in the preamble, **CS-SP**.

Following the UName and YName examples in Sec.(2) ($NU = 3$ and $NY = 4$), the following is an example

$@npss$			
$Speed1$	$Speed2$	$Temp$	
$Dens21$	$Pres14$	$Pres28$	$QFlux41$

Validation checks: The function name must be preceded at the "@" symbol. The input names must match those provided in the preamble. All defined input names must appear, i.e. the number of fields in the second row must equal NU. Output names must also match those provided in **CS-SP**. All defined output names must appear, i.e. the number of fields in the second row must equal NY.
Cross-Referencing: The **SysType** input in the preamble (**CS-SP**) should be set to HBOX. Cross reference NU, NY, UName and YName.

Note (1): When HBOX is selected, the user should include any model dependencies, libraries, or external resources referenced by the executable. i.e.: Governing models such as a high pressure compressor related to a propulsion system should be uploaded alongside the overall program executable if the program requires these models to function properly.

Note (2): As the HBOX can be an executable whose source code can be any programming language, the time dependent output variables are required to be formatted as a comma separated text file with each column corresponding to a unique output variable and time. That is, using the example above, the text file should be formatted as follows:

t	$Dens21$	$Pres14$	$Pres28$	$QFlux41$
0.0	1.123	256.75	105.65	575.05
0.1	1.096	257.50	103.53	570.15

5 Initial Conditions: CS-SI

The following input types are supported for the specification of initial conditions of a dynamic system.

A. Standard Probability Distributions: Continuous Case. The initial state is fully specified in terms of its continuous probability density function (pdf). See appendix for the definition of a pdf.

Taxonomy: IC_A_{pdf name}

The following pdf names are supported.

- **Gaussian.** A N-dimensional Gaussian pdf is fully specified through its mean vector and covariance matrices. Allowed pdf names: Gauss, Gaussian, Normal. Name is not case sensitive. This initial condition input must identify the pdf name, followed by the mean and covariance parameters in the rows below. See Example (Initial Condition: Gaussian)

IC_A_Gauss			
mean	1	0	−1
covariance	0.01	0	0.002
	0	0.04	0.0025
	0.002	0.0025	0.009

(Initial Condition: Gaussian)

  Validation checks: If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. The size of the mean vector must be $N \times 1$. The size of the covariance matrix must be $N \times N$. In addition, the covariance matrix is required to be symmetric and positive definite.
  Cross-Referencing: UI must cross-reference this input with system type (**SysType**). The size of the mean vector must equal either NS of NU, depending on **SysType**, as described above. Similarly, the size of covariance matrix must match up with the size of the mean vector.

- **Uniform.** A N-dimensional Uniform pdf is fully specified through the finite sized domain. Only hypercuboidal domains are admissible. Allowed pdf names: Uniform. Name is not case sensitive. This initial condition input must identify the pdf name, followed by the domain size in the rows below. See Example (Initial Condition: Uniform)

IC_A_uniform		
domain	−1	2
	−2	4
	10	25

(Initial Condition: Uniform)

  Validation checks: The domain input must of size $N \times 2$. If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. The first numerical value in each row must be strictly less that the second numerical value of that row. I.E. $domain(i,1) < domain(i,2)$, $i = 1,2,\ldots,N$.
  Cross-Referencing: UI must cross-reference this input with system type (**SysType**). The size of the domain input will be determined by **SysType**, as described above.

- **Poisson.** The state distribution is assumed to be uncorrelated, each with its own rate parameter. The N-dimensional Poisson is thus assumed to be a product of the univariate marginals. The pdf is fully specified through a rate vector, specifying the rate parameter for each state. Allowed pdf names: Poisson, Rate. Name is not case sensitive. This initial condition input must identify the pdf name, followed by the rate vector in the row below. See Example (Initial Condition: Poisson)

IC_A_Poisson		
rate	0.1	1

(Initial Condition: Poisson)

Validation checks: The rate input vector must of size $N \times 1$. If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. Each element of the rate vector must be strictly positive.
Cross-Referencing: UI must cross-reference this input with system type (**SysType**). The size of the rate input will be determined by **SysType**, as described above.

- **t-Distribution.** A N-dimensional t-distribution is fully specified through its degrees of freedom (scalar positive integer), location vector and scale matrix. Allowed pdf names: Student, T. Name is not case sensitive. This initial condition input must identify the pdf name, followed by the degrees of freedom, location vector and scale matrices in the rows below. See Example (Initial Condition: Student)

IC_A_T		
dof	2	
location	1	0
scale	0.01	−0.004
	−0.004	0.02

(Initial Condition: Student)

Validation checks: The location input vector must of size $N \times 1$. If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. The size of the scale matrix must be $N \times N$. In addition, the scale matrix is required to be symmetric and positive definite.
Cross-Referencing: UI must cross-reference this input with system type (**SysType**). The size of the location and scale inputs will be determined by **SysType**, as described above.

- **Cauchy.** This is a special case of the t-distribution, with one degree of freedom. It is to be specified as such, i.e. by selecting the t-distribution case and setting the number of degrees of freedom to unity.

B. Custom Probability Density Function. Taxonomy: IC_B.

User must provide a function that takes a state/system input value ($x/u \in \Re^N$) as input and outputs the value of the probability density function (pdf) at the input point. Allowed function base: .m (MATLAB©), .py (Python). There are no restrictions on what the internal constitution of the function looks like. Options (non-exhaustive) include, function evaluation and interpolation. This option requires one row of input. The first cell must identify the pdf name, preceded by the "@" symbol

(function identifier), followed by the name of the input variable to the function. This must either be the state name or the System Input name. See example in Eq.(10)

IC_B	
@initialpdf	*u*

(10)

Validation checks: The function name must be preceded by the function identifier, "@". If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, the input to the function must match the state name (defined under **SName** in **CS-SP**. If **SysType** is HBOX, the input to the function must match the system input name (defined under **UName** in **CS-SP**.
Cross-Referencing: UI must cross-reference this input with system type (**SysType**). The name of input variable to function will be determined by **SysType**, as described above.

C. Probability Mass Function. Taxonomy: IC_C

Initial state uncertainty is specified through a table containing possible realizations of uncertainty and the frequency of occurrence of each realization. The data is to be presented as a table, where number of rows (M) equals the number of potential realizations and number of columns equals system dimensionality plus one ($N+1$). The final column (i.e. $(N+1)^{\text{th}}$ column) contains the value of the pmf for each realization of uncertainty. There are two options.

- **Table Written in CSV.** In this case, the table is written out in the csv input sheet. The first row must identify all variable names. The first N cells contain system variable names (states or inputs). The final, $(N+1)^{th}$ cell in the first row can contain a custom name, or be left empty. The input in the $(N+1)^{th}$ cell of the first row is not important.

- **Table Provided as a Data File.** The user may provide the input table as a data file, especially if the table is very large in size. In this case, the user must specify the name of the data file preceded by the data-file identifier, "#". There is only one row of input needed: following the name of the data file, the adjacent rows must identify, in order, the variable names associated with the first N columns of the table. The name for the $(N+1)^{th}$ column is not important, as in the previous case. An example input is provided below.

IC_C					
#initialpmf	*Pres*14	*Dens*21	*Pres*28	*QFlux*41	*pmf*

(11)

In the above example, it is assumed that $N = NU = 4$ and **SysType** is HBOX with input names *Pres*14, *Dens*21, *Pres*28 and *QFlux*41. The user chooses to use the name *pmf* for the final column.

Validation checks: If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. The variable names of the first N columns must match system variable names (inputs if **SysType** is HBOX and states otherwise). All entries of the final column must be non-negative (zero is allowed, although not expected). Also, the sum of entries in the final column must equal unity (1). While the number of columns is predetermined ($N+1$), the number of rows (M) is not. The "end of table" is indicated by an empty row and helps determine M.
Cross-Referencing: UI must cross-reference the input table size (specifically, the number of columns) with system type (**SysType**). The number of columns will be determined by **SysType**, as described above.

D. Full Ensemble. Taxonomy: IC_D
Initial state uncertainty is specified through an ensemble, containing a finite number of realizations of the system uncertainty. An underlying assumption is that each realization has equal probability of realization (difference from a probability mass function). The data is to be presented as a csv table, where number of rows is the size of the realization (number of particles, or, ensemble size: M) and number of columns equals system dimensionality (N). There are two options.

- **Table Written in CSV.** In this case, the table is written out in the csv input sheet. The first row must identify all variable names. The first N cells contain system variable names (states or inputs).

- **Table Provided as a Data File.** The user may provide the input table as a data file, especially if the table is very large in size. In this case, the user must specify the name of the data file preceded by the data-file identifier, "#". There is only one row of input needed: following the name of the data file, the adjacent rows must identify, in order, the variable names associated with the first N columns of the table. An example input is provided below.

IC_D					
#initialens	*Pres*14	*Dens*21	*Pres*28	*QFlux*41	*pmf*

(12)

In the above example, it is assumed that $N = NU = 4$ and **SysType** is HBOX with input names *Pres*14, *Dens*21, *Pres*28 and *QFlux*41.

Validation checks: If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. The variable names of the first N columns must match system variable names (inputs if **SysType** is HBOX and states otherwise). While the number of columns is predetermined (N), the number of rows (M) is not. "End of table" is indicated by an empty row and helps determine M.
Cross-Referencing: UI must cross-reference the input table size (specifically, the number of columns) with system type (**SysType**). The number of columns will be determined by **SysType**, as described above.

E. Partial State Ensemble + Marginal pdf. Taxonomy: IC_EA_{pdf name}, IC_EB, IC_EC
Partial initial uncertainty, $x_1 \in \Re^{N_1}$ (or, $u_1 \in \Re^{N_1}$) , $N_1 < N$ is specified as an ensemble (see IC_D). The remainder, i.e. $x_2 \in \Re^{N_2}$ (or, $u_2 \in \Re^{N_2}$), such that $N_1 + N_2 = N$, has a prescribed distribution. For x_2 (or u_2), options IC_A, IC_B and IC_C are allowed. All rules for these options must be followed.

Validation checks: If the system type is ODE_{A,B,C}, ODF_{A,B,C}, or SDE_{A,B,C}, N must equal NS. If **SysType** is HBOX, N must equal NU. All other validation checks relevant to IC_A, IC_B, IC_C apply.
Cross-Referencing: UI must cross-reference the input table size (specifically, the number of columns) with system type (**SysType**). All other cross-referencing checks relevant to IC_A, IC_B, IC_C apply.

A summary of initial condition inputs is provided in Table (3).

Case Label	Case Name	Input Data	Taxonomy: Case	Taxonomy: Parameters
A.	**Standard pdf (Continuous)**	pdf parameters	IC_A_{pdf name}	
	Gaussian	mean, covariance	IC_A_gauss IC_A_gaussian IC_A_normal	mean covariance
	Uniform	hypercuboidal domain	IC_A_uniform	domain
	Poisson	rate vector	IC_A_poisson	rate
	t-distribution	Degrees of freedom, location, scale	IC_A_t IC_A_student	DOF location scale
	Cauchy	Set deg. freedom $= 1$ in t-distribution	IC_A_t IC_A_student	DOF = 1 location scale
B.	**Custom pdf**	computer code (.m or .py): must output pdf value	IC_B	function
C.	**Probability Mass Function**	csv table of size $M \times (N+1)$: last column contains pmf value	IC_C	table
D.	**Full State Ensemble**	csv table of size $M \times (N)$ containing realizations	IC_D	table
E.	**Partial State Ensemble** $+$ **Marginal Distribution**	csv table for partial state $\mathbf{x}_1 \in \Re^{N_1}$, options A., B. or C. for $\mathbf{x}_2 \in \Re^{N_2}$, $N_1 + N_2 = N$	IC_EA_{pdf name} IC_EB IC_EC	table, parameters table, function tableE, tableC
parenthetical names ($\{\cdot\}$) in taxonomy are not case sensitive. Rest must appear as shown in taxonomy column.				

Table 3: Taxonomy for State Initial Condition Input

6 Simulation Performance

Simulation performance has the following key elements:

A. Quantities of Interest: This includes functions of the state (or input variables for systetm type HBOX) and corresponding error thresholds.

B. Minimum Candidate Particle Pool Size for Simulation Optimization.

C. Suggested Candidate Batch Size for Simulation Optimization.

D. Particle Removal Option.

Description of each of the above items and related inputs are provided below.

A. Quantities of Interest. The user must define all quantities of interest (QOI) and their corresponding upper and lower error bounds (thresholds). Lower error bounds are optional. Two types of QOI definitions are possible:

i. Explicit function expression. In this option, the QOI is expressed as an explicit function of system variables (**CS-SP, Item C.**). For each QOI of this type, four input cells are required, arranged in a single row. The first cell must define the QOI name. The QOI name should not include any special characters. See Eq.(13) for an example. The second cell must contain the functional expression of the QOI. This expression can only contain System Variable Names (states, time, inputs, outputs: **CS-SP, Item C.**) and/or parameter names and/or special terms (e.g. "pi", "sin", "cos", etc.). The third cell contains the upper error bound for this QOI. This input is mandatory. The fourth (and final) cell contains the lower error bound on this QOI. This input is optional. In the second example (Eq.(14)), the fourth cell is left empty, denoting that there is no specified value for the lower error threshold.

Angmom-2	$sqrt(x1 \wedge 2 + K * x2)$	0.1	0.01	(13)
Energy	$sqrt(x1 \wedge 2 + x2 * x1 \wedge 2)$	0.01		(14)

Validation checks: The upper and lower error bounds (call it (U and L) must both be strictly positve, i.e. $U \in \Re^+$ and $L \in \Re^+$ and, $U > L$.

ii. Function call. In this option, the QOI is evaluated through a function call. The name of this function and its input variables must be defined. For each QOI of this type, two rows of inputs are needed:

- **Row 1.** Four input cells are required in the first row. The first cell must define the QOI identifier. No special characters should be used in the QOI name: See QOI: Example 3. The second cell must contain the function name that evaluates the QOI. The name of this function must be preceded by an "@" symbol. The third cell contains the upper error bound for this QOI. This input is mandatory. The fourth (and final) cell contains the lower error bound on this QOI. This input is optional. In QOI: Example 4, the fourth cell is left empty, denoting that there is no specified value for the lower error threshold.
- **Row 2.** This row must identify the input variables needed for the QOI function. Each cell must contain one input variable. Input variables can only contain System Variable Names (states, time, inputs, outputs: **CS-SP, Item C.**) and/or parameter names defined in the preamble (**CS-SP, Item I.**). QOI-Example 3 is given below

HFlux	$@qoiflux$	0.1	0.01
	$x1$	$x3$	Q

(QOI: Example 3)

QOI: Example 4 appears below

Mach Number	*@machnum*	0.01	
	x2	*T*	

(QOI: Example 4)

Validation checks: The upper and lower error bounds (call it (U and L) must both be strictly positve, i.e. $U \in \Re^+$ and $L \in \Re^+$ and, $U > L$. If the QOI expression is a function name (carries an "@" prefix), then there must be a non-empty second row of inputs that identify the input variables for the function. The input variables must be valid, e.g. if **SName** is x and $NS = 2$, then $x3$ is an invalid input variable.

The following example contains a mixed QOI input (explicit function expression and function call):

AngMom-2	$sqrt(x1 \wedge 2 + K * x2)$	0.1	0.01
Mach Number	*@machnum*	0.01	
	x2	*T*	

(QOI: Example 5)

The UI will identify the above input as TWO quantities of interest, the first one identified as "AngMom-2" and the second as "Mach Number". AngMom-2 has an explicit functional form while Mach Number is evaluated through a function call.

B. Minimum Candidate Particle Pool Size for Simulation Optimization. Optimal particles to be added to the simulation are chosen from a candidate pool. The user has the option of defining a minimum size for this pool. The actual candidate pool used in the simulation can be much bigger than this value. This information must be identified as **CPoolSize**, with its numerical value stated in the adjacent cell. This information is optional. See example below.

CPoolSize	5000

(15)

Validation checks: This information is optional. If provided, **CPoolSize** $\in \mathbb{Z}^+$. The user may not provide any entries with the header **CPoolSize**. Alternatively, the header may be provided but there may be no accompanying numerical value. In both cases, the input must be interpreted as "no input provided".

C. Suggested Candidate Batch Size for Simulation Optimization. A key ingredient in ensuring fast completion of forecasting with QOI error guarantees is parallel computing for rapid adaptation of the forecasting ensemble. In the parallel computing framework, a batch of optimal particles are introduced at a time (as opposed to one particle at-a-time). The user is allowed to suggest a batch size for each ensemble update. The input is optional, but if provided, should be identified as **BatchSize**, along with a positive integer numerical value. See example below

BatchSize	25

(16)

Validation checks: This information is optional. If provided, **BatchSize** $\in \mathbb{Z}^+$. A "relatively small" integer value is expected. Must ensure that $BatchSize < CPoolSize$. The user may not provide any entries with the header **CPoolSize**. Alternatively, the header may be provided but there may be no accompanying numerical value. In both cases, the input must be interpreted as "no input provided".

D. Particle Removal Option. Accuracy of forecasts make sense only to an extent. Accuracy also comes at a price (longer run times). Therefore, it is not wise to demand arbitrarily accurate simulations. The forecasting platform has the ability to reduce simulation load if it determines the accuracy of QOI more than what makes sense for a given application. While achieving greater accuracy than desired is unlikely, it does happen. Setting the option to remove particles to unity allows the simulation platform to freeze/drop particles when they are not needed (in the interest of completing simulations faster). The option should be identified as **ParticleRemove**. Setting this to zero will not permit the platform to drop particles from the simulation at any stage. See example below:

$$\boxed{\textit{ParticleRemove}} \ \boxed{0} \tag{17}$$

Validation checks: **ParticleRemove** is a binary variable (0/1). This input is also optional. If this information is not provided by the user, tthe default is **ParticleRemove** $= 0$.

7 System Analytics

In some instances, the metrics being tracked to elicit system performance are not the same as the quantities of interest described in Sec.(6). In this section, the user must specify the functions of system variables that need to be displayed after the forecasting simulation is completed.

A. Display Functions: In this input, the user must define functions of system variables that are to be displayed as a function of time after completion of the forecasting simulation. In general, these functions will be different than the Quantities of Interest. Even if they are the same, they must be re-defined. Two types of definitions are possible. The Display Function name should not contain any speecial characters.

i. Explicit function expression. In this option, the display function is expressed as an explicit function of system variables (**CS-SP, Item C.**). For each DF of this type, two input cells are required, arranged in a single row. The first cell must define the DF identifier. No special characters must be used in display function names. See Eq.(18) for an example. The second cell must contain the functional expression of the Display Function. This expression can only contain System Variable Names (states, time, inputs, outputs: **CS-SP, Item C.**) and/or parameter names and/or special terms (e.g. "pi", "sin", "cos", etc.).

LComb	$x1 + K * x2$	(18)
Damp	$sqrt(x1 \wedge 2 + x2 * x1 \wedge 2)$	(19)

ii. Function call. In this option, the Display Function is evaluated through a function call. The name of this function and its input variables must be defined. For each DF of this type, one row of inputs is needed, see Example (DF: Example 3). The first cell must define the DF identifier. No special characters must be used in the Display Function identifier. The second cell must contain the function name that evaluates the DF. The name of this function must be preceded by an "@" symbol. The third cell-onwards contain the input variables needed for the DF function. Each cell must contain one input variable. Input variables can only contain System Variable Names (states, time, inputs, outputs: **CS-SP, Item C.**) and/or parameter names defined in the preamble (**CS-SP, Item I.**). DF-Example 3 is given below

DF-3	$@qoiflux$	$x1$	$x3$	Q	(DF: Example 3)

Validation checks: Since the DF expression is a function name (carries an "@" prefix), then there must be a non-empty set of inputs following the second cell that define function inputs. The input variables must be valid, e.g. if **SName** is x and $NS = 3$, then $x4$ is an invalid input variable.

The following example contains a mixed DF input (explicit function expression and function call):

LC	$x1 + K * x2$			(DF: Example 4)
Mach	$@machnum$	$x2$	T	

The UI will identify the above input as TWO display functions, the first one identified as LC and the second as Mach. LC has an explicit functional form while Mach is evaluated through a function call.