Supplementary Information: A Hybrid Mechanistic Machine Learning Approach to Model Industrial Network Dynamics for Sustainable Design of Emerging Carbon Capture and Utilization Technologies

August 30, 2023

1 Data Generation For Algal Biodiesel Industrial Network using Aspen Dynamics

We incorporated a simulation strategy to generate time series data for algal biodiesel industrial network by building mechanistic models in ASPEN for the network with 5 individual industrial nodes. We followed this approach as the data is often difficult to acquire from the industries due to its proprietary nature, however if the data from industries are available, it can be directly used in this approach. The 5 different industrial nodes in the modeled algal bio-diesel industrial network are algae growth, pretreatment, fermentation and extraction, purification, and anaerobic digestion. A distributed network approach facilitated in the ease of computational process simulation using ASPEN in dynamic mode, which was computationally challenging when done for whole network at one time. To automate the generation of data, we utilized a FORTRAN script for perturbing forcing variable (input/control variable) and run ASPEN in dynamic mode to generate time series for each of the state variables.

The models for each industrial node were first initialized at steady state to obtain the initial value for all state variables at time $t = 0$ (shown in Figure S7). Then the models were given a perturbation function to specific control variables (input variables), such as a RAMP function or Sine RAMP (SRAMP) function. The forcing functions for the control variables of whole network is also given in Figure S7 in this SI. Similar forcing functions were used for control variables of each node. Using the process models in ASPEN dynamic mode and forcing functions, we generated 10,000 data points for each node which was used for training and testing of surrogate models. In order to obtain data that captures the dynamics of system appropriately in response to the forcing functions, there are four key parameters set up in the simulation. These are a) Simulation time ; b) Dynamic Response Time; c) Loop Time and d) Sampling time, explained below :

(a) Simulation time – This is the total number of time steps (hours) that the dynamical model is run to generate dynamic data. Using a code script in FORTRAN users can specify the exact quantification of time parameter in hours for which the process node has to run. It can either be imitative with the experimental setup or can have specification as per the user requirement of the research conduct. Considering an instance, if the simulation time is set to be 100, this would lead to the process running for 100 hours spread out for each loop. The simulation time for each node was different (such as 500 hours, 400 hours or 1000 hours) as shown in respective node figures for dynamical data (Figures S12 - S17).

(b) Dynamic Response Time (DRT) : For generating dynamic data, each control variable is perturbed using a forcing function (RAMP or SRAMP). DRT is the total time in hours till which the control variable changes. After this time, if the simulation time is not complete, a new forcing function was imposed for new DRT.

(c) Loop Time : The loop time (LT) is the time set that allows for simulation to run for full simulation time based on the DRT and simulation time set by user. Once a loop has been completed, a new time starts with the same conditions still in place. The loop time runs up till the final simulation time specified by the user after which all the variables attains convergence and the simulation does not show any dynamics later onward. With the aim to capture the overall dynamics associated with the conditions, it is imperative to keep the loop time equal to or greater than the DRT. If the loop time has a lesser value assigned, that means a new dynamic run gets started without the system completely capturing the dynamics of a particular loop, leading to the extreme chaos in the dynamicity. Additionally, it is also important to not set DRT much less than the LT value since this will give rise to the dynamic construction of the state variables for the assigned DRT while the system will show steady state value for the time of DRT minus LT. Once the system portrays extra steadiness, the performance and precision of the model generated after the application of the white-box ML algorithm

reduces. The prime reason of such behavior can be attributed to the underlying concept that the algorithm is designed to capture the nonlinear dynamics, and does not perform well if there is presence of several steady state in variables throughout the total time run of the process. In our work, for each of the nodes, we have set the loop time to be slightly larger than the DRT so that steady state doesn't prevail for larger amount of time.

(d) Sampling Time : The sampling time parameter is assigned after the generation of the dynamical data points. This parameter is the user-required time gap between the two subsequent dynamic instances of the state variables which can be different from the default value of 0.01 hours. Taking an example, if the plant is run for 10 hours (X) with sampling time set to be 0.1 hour (Y) and loop time set as 1 hour (Z), this will lead to the generation of 100 data points overall (X*Z/Y). The sampling time for this research study has been systematically set in a way that leads to the generation of 10000 data points overall for each of the nodes, that can be utilized in the machine learning algorithm.

Forcing Functions for Control Variables To generate dynamic data we have used RAMP and Sine RAMP (SRAMP) functions to perturb the control variables for selected DRTs in each loop time. The input/control variable (in our study it is Fm, which is the flow rate of the stream) is typically chosen from the different properties of the input stream going into the process node (for instance flowrate, temperature, density, pressure, etc.) and then it is allowed to vary up till a particular value (Jump Value) with a user specified error tolerance. The conditionals provided in the script also details about the dynamic response time (DRT) that is also a user-defined parameter describing the hours till which the input has to be perturbed inside that particular condition. Finally, we need to specify the curve structure that can either be discrete or continuous (in majority of the use cases, the user prefers to keep the variation to be continuous). We have selected the variation to be continuous for all control variables.

2 Process flow diagrams of the Industrial Nodes in the Algal Biodiesel Industrial Network

In this section, we provide the process flow diagrams for five nodes of the algal bio-diesel industrial network.

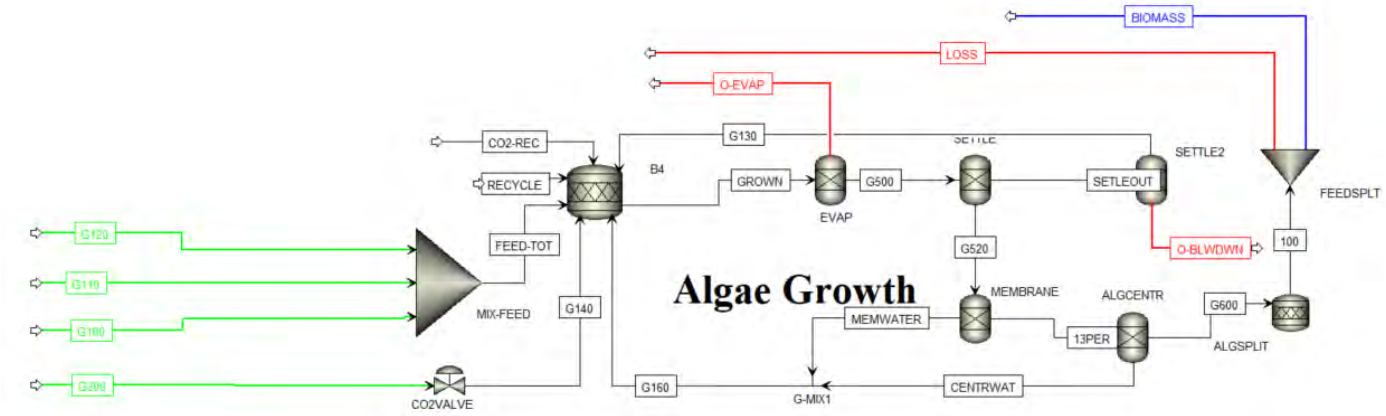

Figure S 1: Process Flow Diagram of the Algae Growth Industrial Node designed using Aspen Plus

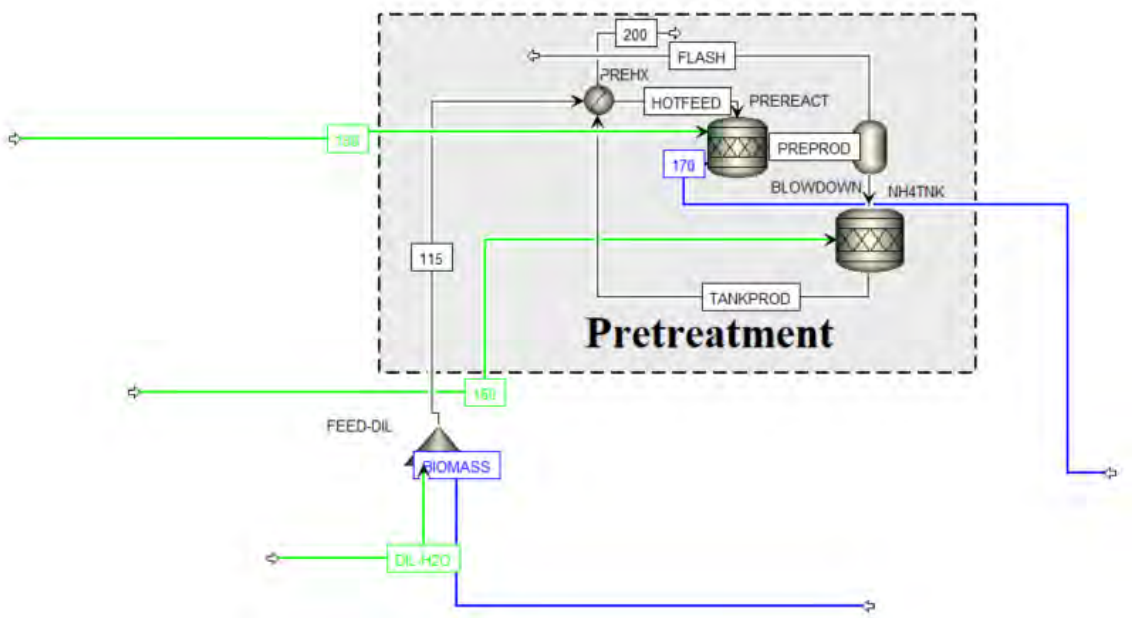

Figure S 2: Process Flow Diagram of the Pretreatment Industrial Node designed using Aspen Plus

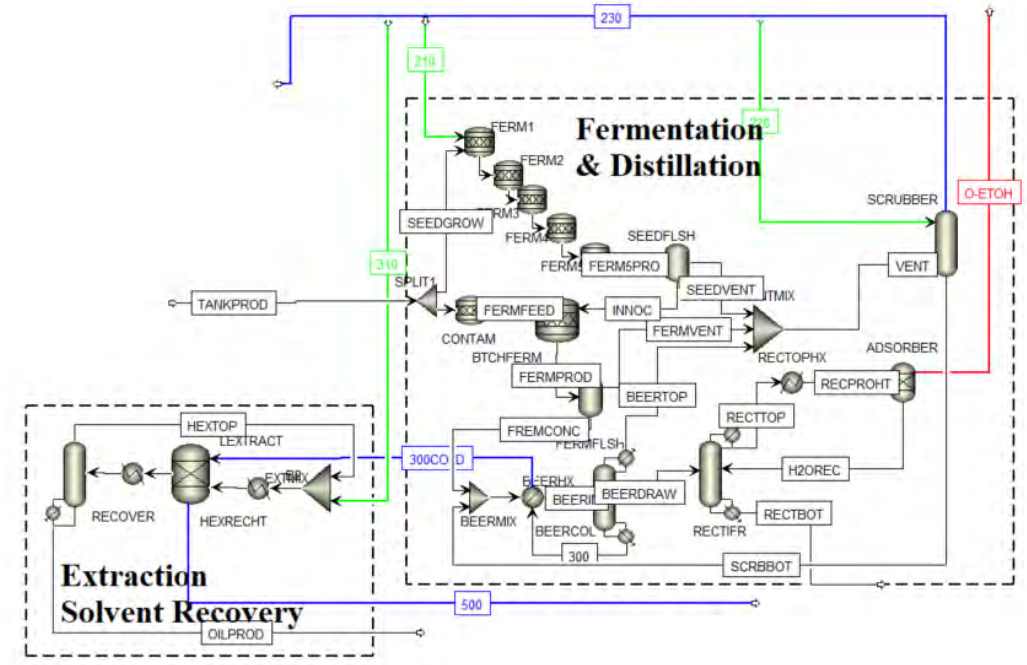

Figure S 3: Process Flow Diagram of the Fermentation & Extraction Industrial Node designed using Aspen Plus

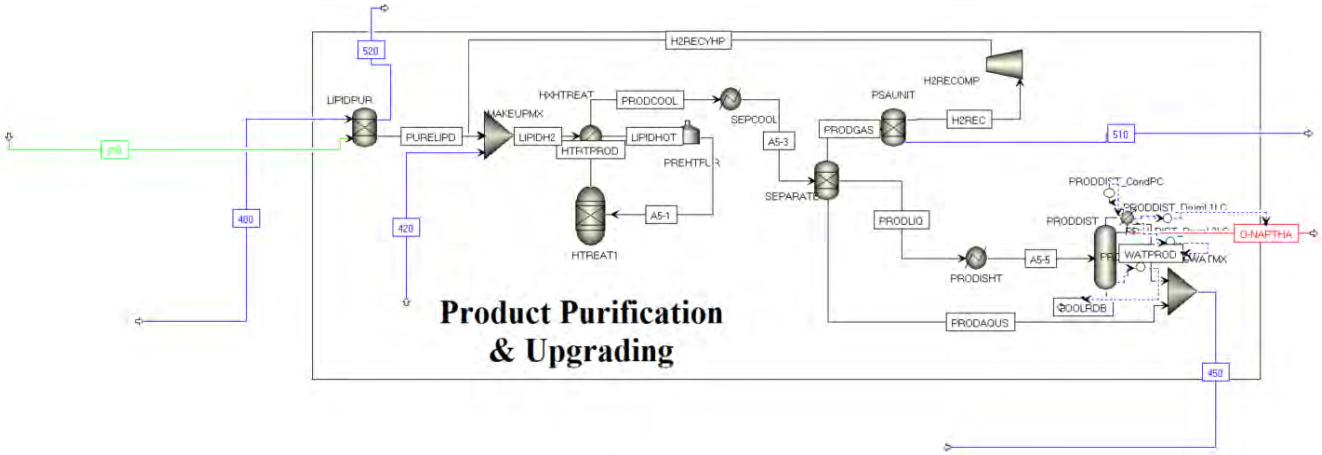

Figure S 4: Process Flow Diagram of the Purification Industrial Node designed using Aspen Plus

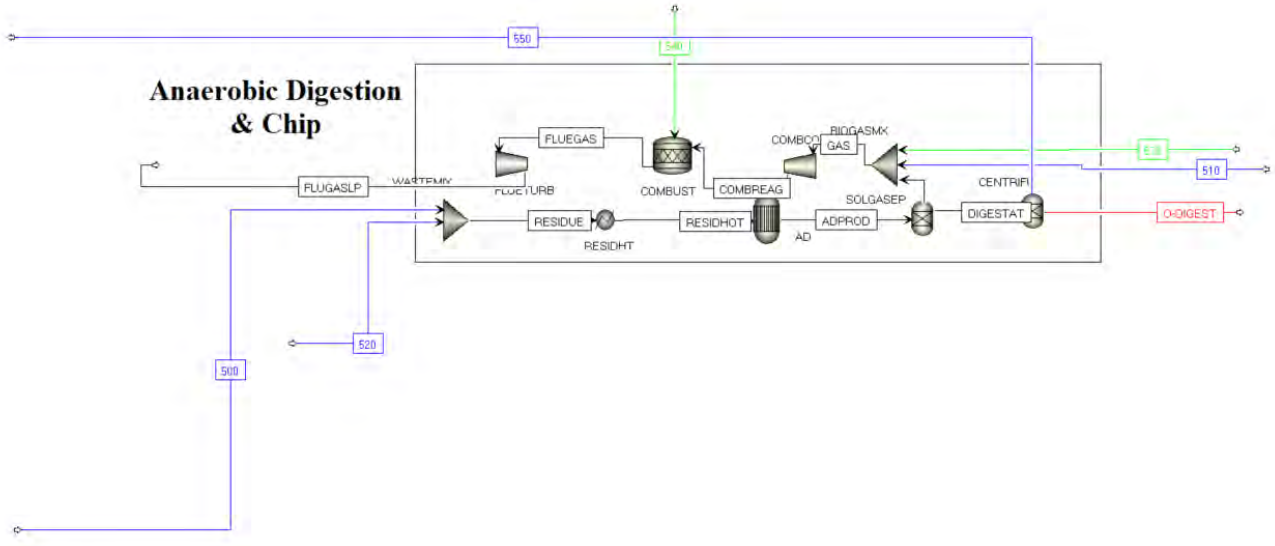

Figure S 5: Process Flow Diagram of the Anaerobic Digestion Industrial Node designed using Aspen Plus

INITIAL VALUES OF STATE VARIABLES				
Algae Growth	Pretreatment	Fermentation & Extraction	Purification	Anaerobic Digestion
x0 = 277819	x5 = 63296.2	x8 = 22575.8	x13 = 4284.71	x17 = 205108
x1 = 56.5043	x6 = 249036	x9 = 237524	x14 = 417.987	x18 = 348170
x2 = 1469970	x7 = 99.7054	x10 = 14823.7	x15 = 588.948	x19 = 1157.78
x4 =24.9671		x11 = 23603.9	x16 = 17425	
		x12 = 7577.74		

FORCING FUNCTION CHARACTERISTICS FOR CONTROL VARIABLES				
	ALGAE STRAIN	CARBON DIOXIDE	WATER	HEXANE & LIPIDIMP
Target Value	5000	53000	26100	2800
Error Tolerance	50	500	200	30
Ramping	Sine RAMP	Sine RAMP	Sine RAMP	RAMP
Dynamics Time	0.1	0.1	0.1	0.5
Loop Time	0.2	0.2	0.2	1

Figure S 6: Initial Values of the variables in the state space of 5 distinct industrial nodes and the characteristics of the forcing function for the chosen control variables in the Algal Biodiesel Production Network

3 Perspective towards the White-box ML algorithm: Sparse Identification of Nonlinear Dynamics (SINDy)

The Sparse Identification of Nonlinear Dynamics (SINDy) is a cutting-edge methodology designed to discover governing equations from data. This approach is particularly useful in a variety of scientific and engineering fields where data are abundant, but models often remain unknown or elusive. The SINDy methodology is based on the assumption that there are only a few important variables in the overall state space that govern the dynamics of a system, making the governing equations sparse in the space of possible functions. This assumption holds for many physical systems when represented in an appropriate basis of control variables regime. The SINDy algorithm operates on the principle of sparse regression. Sparse regression is a statistical technique that involves using a small number of predictor terms to model the response variable via the regularization approach. In the context of SINDy, sparse regression is used to determine the fewest terms in the dynamic governing equations required to accurately represent the data. This results in parsimonious models that balance accuracy with model complexity to avoid overfitting.

The SINDy algorithm can be summarized in the following steps:

1. Data Collection - The SINDy algorithm commences with the Data Collection phase, which is a pivotal step in the entire process. This phase involves the gathering of measurement data from the system that is under investigation. The data collected is represented in the form of a state vector, denoted as x(t), where 'x' signifies the state of the system and 't' represents the time at which the state is observed. This state vector is a mathematical construct that encapsulates the complete information about the system at a specific point in time. It is a snapshot of the system's state variables, which could include quantities like position, velocity, temperature, concentration, and so forth, depending on the nature of the system under study. The state vector x(t) is not just a mere representation of the system's state at a given time, but it is a crucial element that feeds into the SINDy algorithm. The algorithm's ability to derive a sparse and interpretable model that accurately captures the underlying dynamics of the system is heavily reliant on the quality and comprehensiveness of these state vectors. Therefore, the data collection process is not a trivial task but a critical component that can significantly influence the accuracy and reliability of the resulting model. The data collection process should be designed and executed with a focus on covering the entire state space of the system. The state space of a system is the multidimensional space in which all possible states of the system are represented. Each dimension in this space corresponds to one state variable, and each point in this space represents a unique state of the system. By ensuring that the collected data spans the entire state space, we can ensure that the resulting model is capable of accurately representing the dynamics of the system under all possible states. This is particularly important for systems that exhibit non-linear dynamics, where the system's behavior can change drastically from one region of the state space to another. In addition to the coverage of the state space, the quality of the data collected is also of paramount importance. The data should be as free from noise and errors as possible. Noise in the data can arise from various sources, including measurement errors, environmental fluctuations, and inherent stochasticity in the dynamics of the system. These sources of noise can introduce random variations in the state vector x(t), which can obscure the true underlying dynamics of the system and lead to inaccurate or unreliable models. Therefore, efforts should be made to minimize the noise in the data, either by improving the measurement techniques or by applying noise reduction techniques to the collected data.

2. Construction of Candidtae Library Functions - The second phase in the SINDy algorithm is the Library Construction. This step involves the creation of a library of candidate functions, denoted as $\Theta(X)$, which serves as a pool of potential terms that could appear in the governing equations of the system. This library is represented as a matrix, where each column corresponds to a candidate function. The candidate functions are functions of the state variables and their derivatives, and they represent the possible terms that could appear on the right-hand side of the governing equations in the oridnary differential equation (ODE) form. The construction of the library is a critical step in the SINDy algorithm, as it defines the search space for the sparse regression problem that is solved in the next step of the algorithm. The quality and appropriateness of the library can significantly influence the accuracy and interpretability of the resulting model. A well-constructed library can enable the SINDy algorithm to discover accurate and interpretable models, while a poorly constructed library can lead to inaccuracy and non-interpretability in the models. The choice of functions to include in the library is a flexible aspect of the SINDy algorithm. The library can include a wide variety of functions, such as polynomials, fourier functions, exponential functions, and other basis functions. The choice of functions should be guided by the underlying physics of the system and prior knowledge about the system dynamics. For instance, if the system is known to exhibit sinusoidal oscillations, then it would be appropriate to include fourier sine-based functions in the library. Similarly, if the system is known to exhibit exponential growth or decay, then it would be appropriate to include exponential functions in the library. In addition to the choice of functions, the order of the functions is also an important consideration in the construction of the library. The order of a function refers to the highest power of the state variables or their derivatives that appear in the function. For example, a quadratic function has an order of 2, while a cubic function has an order of 3. The order of the functions

in the library should be chosen to match the complexity of the system dynamics. A system with simple linear dynamics may only require functions of order 1, while a system with complex nonlinear dynamics may require functions of higher order. The construction of the library is not a generalizable or universal-across-all-systems process, but rather a flexible, iterative, and adaptable process that can be tailored to the specific characteristics of the system under study. The goal of this stage is to construct a library that is rich enough to capture the complexity of the system dynamics, but not so complex that it leads to overfitting or uninterpretable models. This requires a delicate balance between complexity and simplicity, and it often requires a good understanding of the dynamics of the system and a careful consideration of the trade-offs involved.

3. Application of Regularization Techniques - The third phase in the SINDy algorithm is the Sparse Regression or Regularization. This step involves solving a sparse regression problem to determine the sparse vectors of coefficients, denoted as Ξ, that dictate which terms are active in the dynamics of the system. The sparse regression problem is formulated as an optimization problem, where the objective is to find the sparsest Ξ that satisfies the equation 3. In this equation, the left hand side is the time derivative of the state x(t), $\Theta(X)$ is the library of candidate functions, and Ξ is the matrix of coefficients. The sparse regression problem lies at the heart of the SINDy algorithm. It is the step where the algorithm transitions from a purely data-driven approach to a model-driven approach via optimization techniques. The sparse regression problem leverages the data collected in the first step and the library of candidate functions constructed in the second step to discover a sparse and interpretable model that accurately captures the underlying dynamics of the system. The sparse regression problem is typically solved using a sparsity-promoting optimization algorithm such as STLSQ (Sequentially-Thresholded Least Squares), LASSO (Least Absolute Shrinkage and Selection Operator), or Elastic Net, among many others. These algorithms are designed to promote sparsity in the solution by adding a penalty term to the objective function that penalizes the number of non-zero coefficients in Ξ. The penalty term is controlled by a hyperparameter, often denoted as λ, which determines the level of sparsity in the solution. A larger value of λ will result in a sparser solution with fewer non-zero coefficients, while a smaller value of λ will result in a denser solution with more non-zero coefficients, often known as a complex solution. The choice of the sparsity-promoting optimization algorithm and the value of the sparsity parameter λ are critical aspects of the sparse regression problem. Different algorithms and values of λ can result in different solutions, which can lead to different models of the system dynamics. Therefore, the choice of the algorithm and the value of λ should be made with appropriate knowledge, taking into account the characteristics of the system, the quality of the data, and the desired level of sparsity in the model. The solution to the sparse regression problem, denoted as Ξ^*, is a matrix of coefficients that determines which terms in the library of candidate functions are active in the dynamics of the system. Each column of Ξ^* corresponds to a state variable, and each row corresponds to a function in the library. A non-zero coefficient in Ξ^* indicates that the corresponding function is active in the dynamics of the corresponding state variable.

4. Validation of the Surrogate Models and Performance Calculation - The fourth and final phase in the SINDy algorithm is Model Selection via Validation and Performance Calculation. This step involves the selection of the model that best balances accuracy with model complexity. The model selection process is a critical component of the SINDy algorithm, as it ensures that the resulting model is not only accurate in representing the observed data but also parsimonious in its complexity, thereby preventing overfitting and ensuring that the model generalizes well to unseen data under a particular dynamic regime. The model selection process is essentially a process of comparison and selection. It involves comparing the performance of different models, each represented by a different solution to the sparse regression problem, on a validation dataset. The validation dataset is a subset of the collected data that is set aside and not used in the training process of the sparse regression problem. It serves as new, unseen data for the models and provides an unbiased estimate of their predictive performance. The performance of the models is evaluated using a suitable accuracy metric. Two of the commonly used accuracy metrics are the R-squared (R^2) and the Root Mean Square Error (RMSE). The R-squared (R^2) is a statistical measure that represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s). It provides a measure of how well the regression predictions approximate the real data points. An R^2 of 100% indicates that all changes in the dependent variable are completely explained by changes in the independent variable(s). In the context of the SINDy algorithm, a higher R^2 indicates a model that explains a larger proportion of the variance in the dynamics of the system, and hence, is more accurate. On the other hand, the Root Mean Square Error (RMSE) is a frequently used measure of the averaged Euclidean distance differences between values predicted by a model and the values observed. It squares the differences before they are averaged, giving relatively high weight to large errors. This means the RMSE is most useful when large errors are particularly undesirable. In the context of the SINDy algorithm, a lower RMSE indicates a model that has smaller prediction errors, and hence, is more accurate. The model that achieves the best performance according to the chosen accuracy metric is selected as the final model. This model represents the sparsest combination of functions from the library that can accurately represent the system dynamics. It is the model that best balances accuracy with model complexity, thereby preventing overfitting and ensuring good generalization to unseen data. In addition, the model selection process is also an iterative process dependent on the accuracy of the system, that can be tailored to the specific characteristics of the

system and the data. The choice of the accuracy metric, the validation dataset, and the trade-off between accuracy and model complexity are all the necessary factors that need to be carefully considered in the stage.

The SINDy algorithm can be adapted to handle a variety of systems. For example, it can be generalized to handle parameterized systems, systems that are time-varying, or systems with external forcing. It can also be used in conjunction with dimensionality reduction techniques for high-dimensional systems that evolve on a low-dimensional manifold. The success of the algorithm often depends on the ability to choose the right coordinates and function basis that yield sparse dynamics.

The goal of SINDy is to discover a model for dynamical systems in the form given in Equation 1.

$$\frac{d(x(t))}{dt} = f(x(t)) \tag{1}$$

where $x(t) \in R^n$ is time-series data of the state variables and the dynamics encoded by the function f.

For modeling of non-linear dynamical systems with known external forcing function given as $u(t) \in R^n$, the $f\left(x(t), u(t)\right)$ is a linear combination of non-linear functions of $x(t)$ and $u(t)$. This leads to Equation 2:

$$\dot{x_t} = \Sigma_{i=1}^{k} \xi_i \theta_i(x(t), u(t)) \tag{2}$$

where θs are non-linear functions called the candidate terms of $f(x, u)$ that can comprise of user-provided function which can be of the form constant, polynomial, Fourier, or any custom-defined function as well, while ξs are the coefficients of the terms identified by SINDy algorithm. It is crucial to choose the library of candidate functions carefully and can include any function that might describe the data. Additionally, numerical differentiation methods such as finite differences or smoothed finite differences method are used to calculate the time derivative of state variable dynamical data that are used in SINDy.

The system in Equation 2 can be written in terms of these data matrices:

$$\dot{X}_t = \Theta\left(X\left(t\right), u(t)\right)\Xi \tag{3}$$

4 Surrogate Models obtained for each Industrial Nodes

In this section, we provide the surrogate dynamical models as ODEs for all the 20 state variables that describe the dynamics of full algal biodiesel network. These models can be used independently to represent the dynamics of the nodes based on specific control variables mentioned in each individual node models. In our work, we have coupled these models to study the overall algal biodiesel network dynamics in response to key control variables shown in Figure S7.

$$(x0)' = -40.860\ x1 + -40.041\ u0 + 14.894\ x0^2\ x1 + 31.045\ x0\ x1^2 + 16.129\ x1^3 + -49.031\ x1\ x2\ x3 + 51.280\ x1\ x3\ u1 + -50.611\ x2\ x3\ u0 + 52.873\ x3\ u0\ u1$$

$$(x1)' = 43.519\ x1 + 43.507\ u0 + -47.548\ x0^3 + -73.043\ x0^2\ x1 + 40.590\ x0^2\ u0 + -83.561\ x0\ x1^2 + -32.295\ x1^3 + 62.400\ x1\ x2\ x3 + -65.163\ x1\ x3\ u1 + 65.308\ x2\ x3\ u0 + -68.109\ x3\ u0\ u1 + -14.830\ u0^3$$

$$(x2)' = -27.168\ x2 + 26.829\ u1 + 82.052\ x0^2\ x3 + -90.347\ x0\ x1\ x2 + 78.252\ x0\ x1\ x3 + 61.264\ x0\ x1\ u1 + -76.784\ x0\ x3\ u0 + -26.863\ x0\ u0\ u1 + -69.684\ x1^2\ x2 + 79.140\ x1^2\ u1 + -75.466\ x1\ x3\ u0 + 47.520\ x1\ u0\ u1 + 35.143\ u0^2\ u1$$

$$(x3)' = -55.844\ x1^2\ x2 + 23.995\ x1^2\ u1 + -70.347\ x1\ x2\ u0 + -40.419\ u0^2\ u1$$

$x0 = Stream\ Flowrate\ of\ Algal\ Biomass\ (BIOMASS)$

$x1 = Density\ of\ LOSS\ Stream\ majorly\ comprised\ of\ Fatty\ Acids,\ Glucan,\ and\ Water\ (LOSS)$

$x2 = Stream\ Flowrate\ of\ CarbonDioxide\ and\ Water\ Mixture\ coming\ out\ of\ Evaporator\ (O-EVAP)$

$x3 = Temperature\ of\ the\ Algae\ Culture\ Reactor\ (B4)$

$u0 = Stream\ Flowrate\ of\ Algal\ Complex\ Strain\ (G120)$

$u1 = Stream\ Flowrate\ of\ Carbon\ Dioxide\ (G300)$

Figure S 7: Surrogate model obtained for the Algae Growth Industrial Node in a degree 3 polynomial function form

$$(x4)' = 854.292\,x4 - 1.351\,x5 + 4.789\,x0 - 855.728\,u2 - 335563.528\,x4^3 + 180.767\,x4^2\,x5 + 11.377\,x4^2\,x6 + 1010259.534\,x4^2\,u2 + 27.967\,x4\,x5^2 + 247.804\,x4\,x5\,x0 - 16.942\,x4\,x6^2 + 66.142\,x4\,x6\,x0 - 175.442\,x4\,x0^2 - 1013624.938\,x4\,u2^2 + 17.361\,x5^3 - 22.960\,x5^2\,x0 + 7.634\,x5\,x0^2 - 269.220\,x5\,x0\,u2 - 172.828\,x5\,u2^2 + 17.235\,x6^2\,u2 - 65.596\,x6\,x0\,u2 - 11.734\,x6\,u2^2 + 176.765\,x0^2\,u2 + 338926.006\,u2^3$$

$$(x5)' = -695.686\,x4 + 6.849\,x5 - 7.979\,x0 + 701.529\,u2 + 7464.239\,x4^3 - 11159.272\,x4^2\,u2 - 161.117\,x4\,x5^2 + 55.215\,x4\,x5\,x0 - 1.048\,x4\,x6^2 - 51.080\,x4\,x6\,x0 + 53.960\,x4\,x0^2 + 161.822\,x5^2\,u2 - 55.673\,x5\,x0\,u2 + 0.328\,x6^2\,u2 + 51.151\,x6\,x0\,u2 - 53.266\,x0^2\,u2 + 3694.917\,u2^3$$

$$(x6)' = -260.405\,x4 + 261.016\,u2 + 17.112\,x4^2\,x6 - 17.009\,x6\,u2^2$$

$x4 = Stream\ Flowrate\ of\ water\ coming\ out\ of\ flash\ unit\ operation$

$x5 = Stream\ Flowrate\ of\ pretreated\ slurry$

$x6 = Reactor\ temperature\ for\ ammonium\ sulfate\ production$

$x0 = Control\ Variable = Stream\ Flowrate\ of\ biomass\ slurry\ coming\ out\ of\ the\ Algae\ Growth\ node$

$u2 = Control\ Variable = Stream\ Flowrate\ of\ Water$

Figure S 8: Surrogate model obtained for the Pretreatment Industrial Node in a degree 3 polynomial function form

$$(x7)' = -0.282 * x7 - 1.881 * x8 + 0.841 * x9 + 0.614 * x10 - 0.340 * x11 + 0.715 * x5$$

$$(x8)' = -0.643 * x8 + 1.380 * x9 + 0.741 * x10 - 1.463 * x11 + 0.730 * x5$$

$$(x9)' = 0.230 * x7 + 2.036 * x8 - 0.960 * x10 - 3.439 * x11 + 1.222 * x5$$

$$(x10)' = -1.801 * x8 + 0.532 * x9 + 0.572 * x10 + 0.657 * x5$$

$$(x11)' = 3.017 * x9 + 0.663 * x10 - 2.106 * x11 - 0.529 * x5$$

$x7 = Stream\ Flowrate\ of\ algal\ oil$

$x8 = Stream\ Flowrate\ of\ Extraction\ Mixture$

$x9 = Stream\ Flowrate\ of\ Carbon\ Dioxide\ coming\ out\ of\ the\ scrubber$

$x10 = Stream\ Flowrate\ of\ Water\ from\ the\ distillation\ column$

$x11 = Stream\ Flowrate\ of\ Ethanol$

$x5 = Control\ Variable = Stream\ Flowrate\ of\ pretreated\ slurry$

Figure S 9: Surrogate model obtained for the Fermentation & Extraction Industrial Node in a degree 1 polynomial function form

$$(x12)' = -11.151 * sin(x12) - 0.001 * sin(x13) + 0.013 * sin(x14) - 1.514 * sin(x15) + 36.957 * sin(x16) - 25.210 * sin(x7)$$

$$(x13)' = -6.790 * sin(x12) - 0.002 * sin(x13) + 0.001 * sin(x14) - 0.890 * sin(x15) + 21.623 * sin(x16) - 14.482 * sin(x7)$$

$$(x14)' = 0.012 * sin(x13) - 0.066 * sin(x14) - 0.409 * sin(x15) + 1.214 * sin(x16) - 1.283 * sin(x7)$$

$$(x15)' = 0.036 * sin(x13) + 0.423 * sin(x14) - 0.541 * sin(x15) + 4.256 * sin(x16) - 2.857 * sin(x7)$$

$$(x16)' = -10.350 * sin(x12) - 0.001 * sin(x13) + 0.013 * sin(x14) - 1.508 * sin(x15) + 37.031 * sin(x16) - 26.098 * sin(x7)$$

$$x12 = Stream\ Flowrate\ of\ Gaseous\ mixture\ of\ CO2,\ CO,\ H2,\ and\ Propane$$

$$x13 = Stream\ Flowrate\ of\ dilute\ Phosphoric\ acid$$

$$x14 = Stream\ Flowrate\ of\ Naptha\ like\ long\ C\ chain\ compounds$$

$$x15 = Stream\ Flowrate\ of\ Renewable\ Diesel\ Blendstock\ (RDB)$$

$$x7 = Control\ Variable = Stream\ Flowrate\ of\ algal\ oil$$

Figure S 10: Surrogate model obtained for the Purification Industrial Node in a Fourier function form

$$(x17)' = -3.854\ x17 - 7.806\ x19 + 7.769\ x12 + 4.161\ u3$$

$$(x18)' = -4.723\ x18 + 4.732\ x19$$

$$(x19)' = -4.732\ x18 + 4.725\ x19$$

$$x17 = Stream\ Flowrate\ of\ Centrifugation\ Mixture$$

$$x18 = Stream\ Flowrate\ of\ Flue\ Gas$$

$$x19 = Reactor\ Temperature\ of\ the\ Combustion\ Reactor$$

$$x12 = Control\ Variable = Stream\ Flowrate\ of\ Gaseous\ mixture\ of\ CO2,\ CO,\ H2,\ and\ Propane$$

$$u3 = Control\ Variable = Stream\ Flowrate\ of\ Hexanee\ and\ Lipid\ Impurities\ Mixture$$

Figure S 11: Surrogate model obtained for the Anaerobic Digestion Industrial Node in a degree 1 polynomial function form

5 Numerical Integration of Surrogate Models for Dynamic Reconstruction used for Visual Validation of Models

In this section we describe the integration method used for each surrogate models to test the dynamical reconstruction of state variable data.

(a) **Algae Growth node:** Based on the ODE model obtained for the Algae growth node that consists of degree 3 polynomial terms, as observed from the methodology section, the stiffness check experiment was conducted for this system, as detailed previously, and based on the conclusion that the ODEs are stiff the implicit numerical integrator LSODA was used to integrate the differential equation obtained for the 4 state variables viz. flowrate of biomass stream (STREAMS("BIOMASS").Fm), density of loss stream (STREAMS("LOSS").Rho), flowrate of the byproduct stream coming from the evaporator (STREAMS("O-EVAP").Fm), and temperature of the stoichiometric reactor B4 (BLOCKS("B4").T). Furthermore, since the dynamical simulation for the algae growth node has been carried out for 500 hours, the integration has been performed iteratively in 16 steps with some steps having an integration run of 20 hours, and some having that of 40 hours. For each iteration of integration, the time step of integration was kept extremely small to the value of 0.00005 hours (this comparison can be visualized based on the sampling time of the dynamical simulation which was kept as 0.05 hours). As detailed before, due to the limitations of the number of data point generation to 10000 by Aspen Dynamics tool, it was imperative to interpolate the control variable data between the data points of two different sampling time stamps. The need for the interpolation of control variable data is attributed to its use in the *model.simulate()* command, for instance, for integrating the system of ODEs generated for this node from 0 to 20 hours with a time step of 0.00005 hours, the number of control variable data points necessary is 400000. In totality, the number of data points (counting the number of rows in the control variable array) necessary for the integrative simulation of the algae growth system of differential equations is 10000000, since the dynamical simulation has been run for 500 hours. Importantly, the initial 450 hours of data point has been saved for the training of model while the data points worth the final 50 hours of simulation has been saved for the in-sample testing to have an accuracy validation. The interpolation for the control variable, thus, has been performed using the cubic interpolation technique, while appropriately allocating the number of data points in each of the iterations of the numerical integration. Further down this stage, the array of values for all the state variables generated in each iteration is then concatenated together to form a single array which spans across 0 to 500 hours. This aids in the generation of dynamically reconstructed plots, as shown in figure 12 against the original data, which provides a clarity on the behavior of the generated mathematical model of the system, while additionally providing a visual interpretation on the accuracy of the said model.

(b) **Pretreatment node:** As observed in the previous section detailing about the mathematical model obtained via the use of PySINDy, the pretreatment node had the polynomial behavior as well ,with the degree 3 terms in the differential equation. Continuing from this stage, the integration has been performed using the same PySINDy command as used for the Algae Growth node, with the numerical integration method as LSODA and the time step here has again been kept as low as 0.00001 hours for the integrative simulation of all the state variables of the system viz. flowrate of the stream coming out of flash operation (STREAMS("FLASH").Fm), stream flowrate of the pretreated tank slurry (STREAMS("TANKPROD").Fm), and the temperature of the ammonia mixing stoichiometric reactor (BLOCKS("NH4TNK").T). Since the dynamical simulation for the node has been run for 400 hours where the first 360 hours worth of data is used for the training of model while the data obtained from the final 40 hours is used for testing, the interpolation of the control variable has been done accordingly using the cubic interpolation technique, resulting in the generation of the total of 40000000 data points. The number of iterations used in the numerical integration approach is 10 with each iteration having a varying simulation time (1 hour, 30 hours, 50 hours, 100 hours, etc.), while the number of data points from the control variable has been appropriately allocated in each iterations. Again, once each iteration of the integration is performed, the array of values obtained is concatenated to arrange it into a single array, which can further be plotted against the original data of the pretreatment node obtained upon the dynamical simulation thus generating the dynamically reconstructed plots, as can be seen in figure 13.

(c) **Fermentation & Extraction node:** Adopting the similar procedure as was done for the previous two nodes, the integrative simulation for this node has been performed in an iterative manner where the model obtained consists of degree 1 terms albeit putting the function library as degree 2 polynomial during the training using PySINDy, as discussed in the previous section. Here again, LSODA numerical integration method has been applied for integrating the system of ODEs obtained for 5 different state variables of the system viz. flowrate of the stream consisting of the fatty acid and methyl esters based oil (STREAMS("OILPROD").Fm), flowrate of the stream consisting of protein (STREAMS("500").Fm), flowarate of the CO2 stream coming out of the scrubber (STREAMS("230").Fm), flowrate of water stream (STREAMS("RECTBOT").Fm), and flowrate of the ethanol stream (STREAMS("O-ETOH").Fm). The time step of integration for each iteration has been

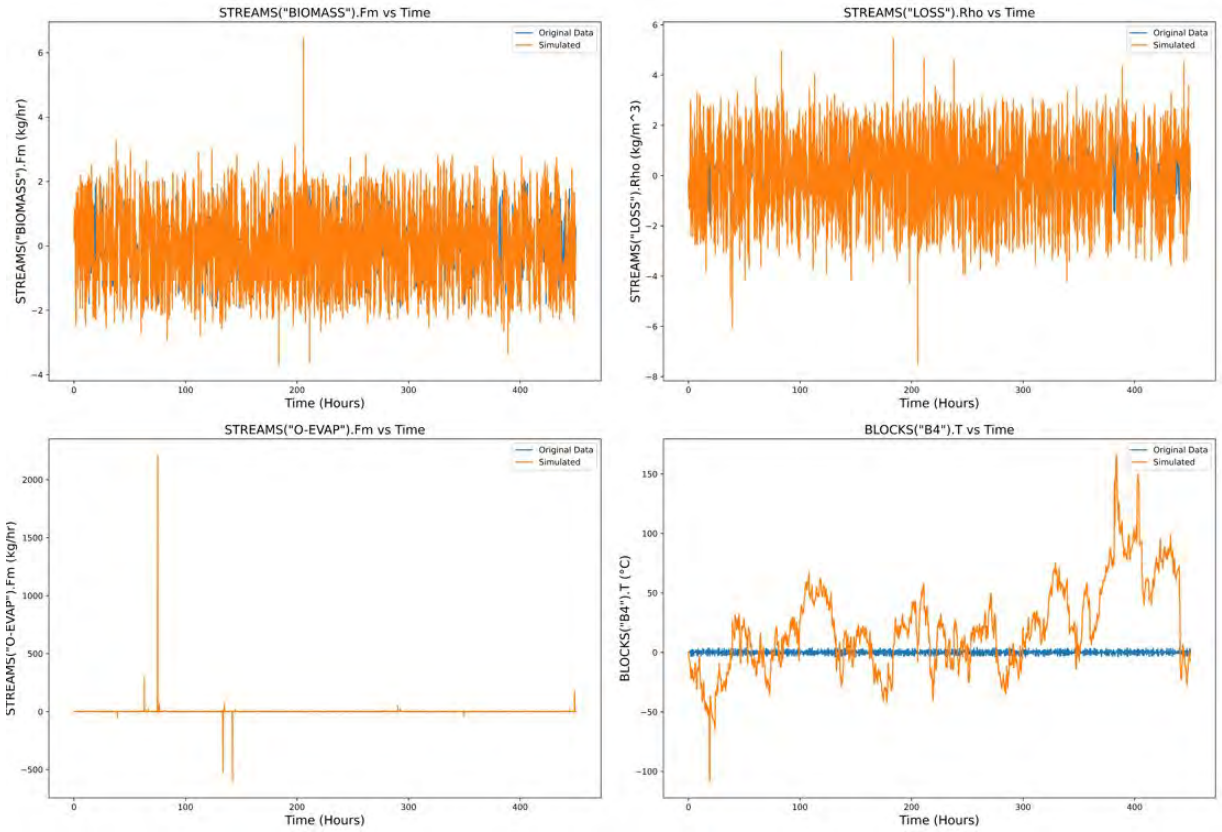

Figure S 12: Dynamical Reconstruction Plot of the Algae Growth Node simulated upto 500 hours

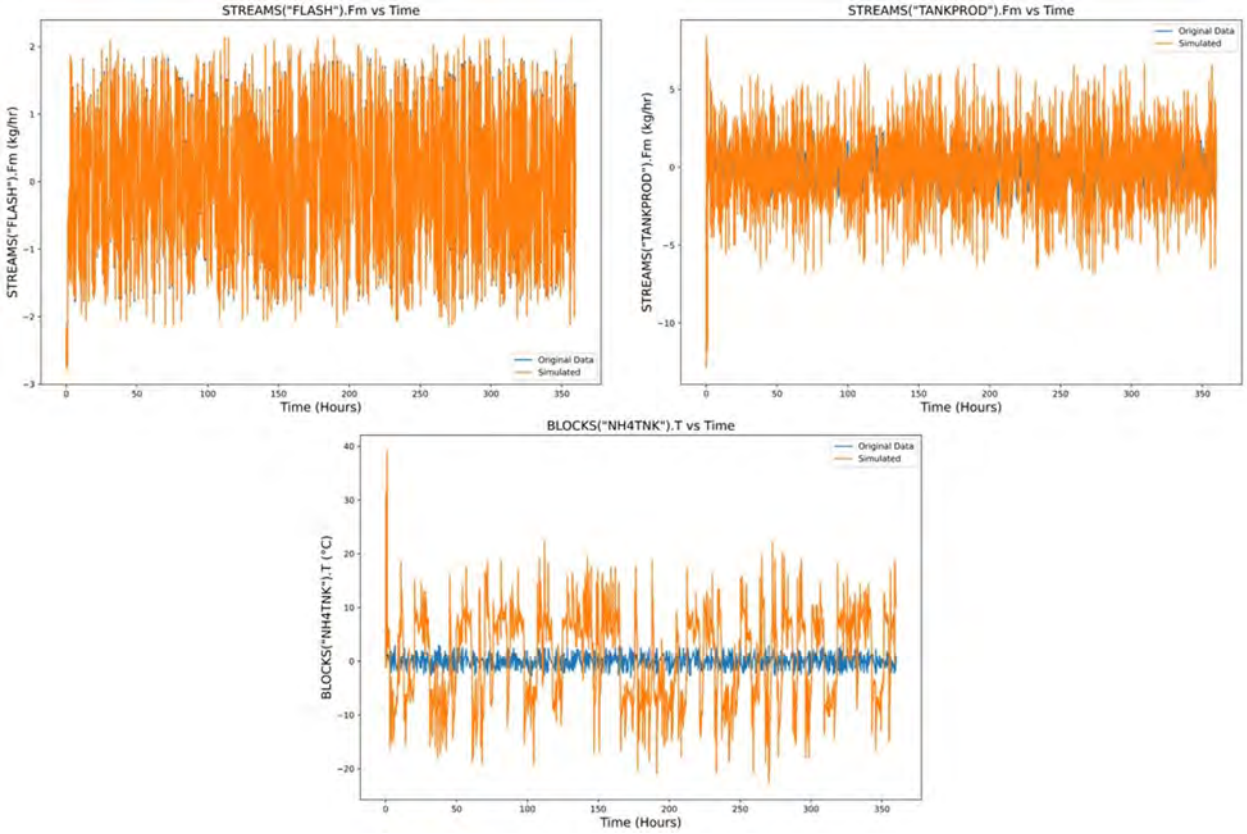

Figure S 13: Dynamical Reconstruction Plot of the Pretreatment Node simulated upto 400 hours

kept as low as 0.00005 hours, compared to the sampling time of 0.1 hours for the aspen dynamical simulations. Since the number of data points of the control variable here is 10000 obtained upon simulating the system for a

total of 1000 hours where, cubic interpolation method has been used to obtain a total of 20000000 data points used in 49 iterations of integration, with each iteration having a varying total time of integration viz. 10 hours, 15 hours, and 25 hours. It is important to make a note here again that the first 900 hours worth of data has been utilized in training the model to obtain the ODEs while the final 100 hours worth of data is used for the accuracy testing. Ultimately, once each of the 49 iterations have bee run for the integrative simulation, the arrays are concatenated to form a single array of state variable values, and the dynamical reconstruction, as seen in figure 14, is performed to visualize the behavior of the model obtained via PySINDy.

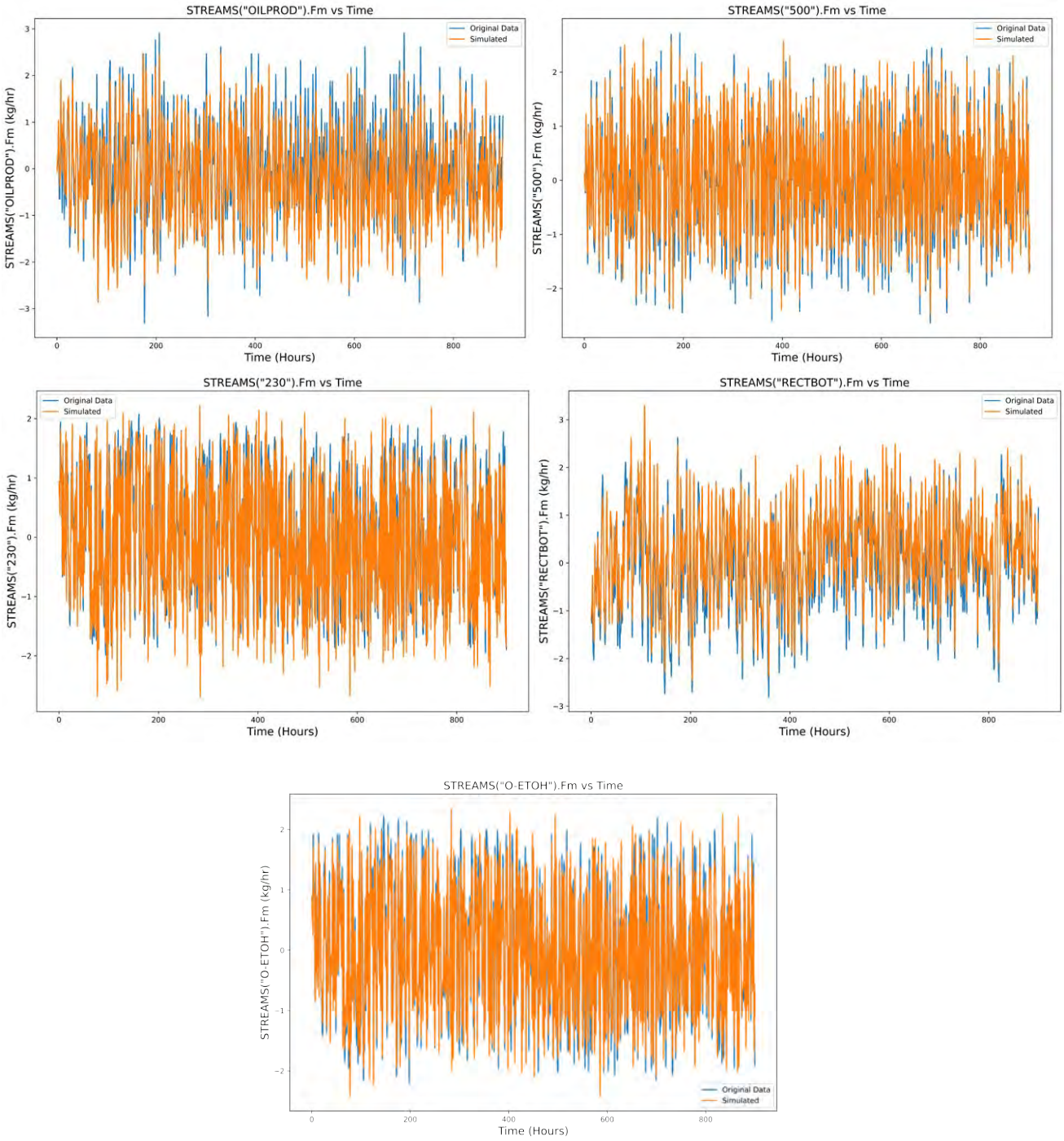

Figure S 14: Dynamical Reconstruction Plot of the Fermentation and Extraction Node simulated upto 1000 hours

(d) **Purification node:** Similar to the integration approach applied to the previous three nodes, the purification node undergoes the integrative simulation through different iterations via LSODA numerical intergrator. As it has been demonstrated before, the model comprise of the fourier sine terms in the ODE

format for all the five state variables viz. flowarte of the stream comprising of propane, CO2, CO, and H2 coming out of a separator unit (STREAMS("510").Fm), stream flowarte of phosphoric acid and water mixture (STREAMS("450").Fm), stream flowarte of the naptha (STREAMS("O-NAPTHA").Fm), the flowrate of the renewable diesel blendstock (STREAMS("COOLRDB").Fm), and the temperature of the HTREAT1 yield reactor (BLOCKS("HTREAT1").T), and this strategy provides a stable result upon performing the iterative integrative simulation by keeping a time step of 0.00005 hours. For this system as well ,there has been 10000 data points for the control variable which was needed to be interpolated, since the total time run of the dynamical simulation is 1000 hours with the sampling being done every 0.1 hours. Furthermore, the first 900 hours worth of data has been utilized for training the model using PySINDy while the final 100 hours worth of data has been kept for the accuracy testing in the in-sample mode. The interpolation of the control variable has been done using the same cubic interpolation method that was utilized in the previous three nodes, generating a total of 20000000 data points. This huge dataset has been utilized in 32 iterations of integrative simulation with varying time of simulation used in each iteration. Finally, the arrays of values for all the 32 iterations are concatenated together, generating a single big array of all the five state variables which is used for the visual generation of dynamically reconstructed plots as can be seen in the figure 15 below.

(e) **Anaerobic Digestion node:** The integrative simulation for the final node of the algal biodiesel production network has been performed in a continuous manner and not in different iterations, as was done for the previous four nodes, the principal reason being that the system has lesser number of state variables, the mathematical model obtained having simpler terms in the differential equation, and the occurrence of the collinear state variables in the system which has been discussed in the prior section. As shown previously, the system produces a mathematical model comprising of degree 1 terms even though the training has been performed using the function library of degree 2 polynomial, while it consists of 3 state variables viz. flowrate of the recycle water and nutrient stream that goes back to the algae growth node (STREAMS("550").Fm), stream flowrate of the flue gas stream comprising of CO2, N2, O2, and water, which is also recycled back to the algae growth node (STREAMS("FLUGASLP").Fm), and the temperature of the combusting stoichiometric reactor that produces flue gas stream (BLOCKS("COMBUST").Fm). the obtained model has been integrated using the LSODA integration technique with the time step of 0.00001 hours for the total time of 1000 hours, which is also the total time of the dynamical simulation run. Similar to the previous node ,the anaerboic digestion node has also been trained using the first 900 hours worth of data set while the accuracy testing for the node has been done using the final 100 hours worth of dynamical simulation data. Additionally, the interpolation of the data for the control variable array was necessary in this case as well, so that the integrative simulation can be carried out for such extremely small time step for the total time of 1000 hours. The interpolation, again, has been performed using the cubic interpolation technique that leads to the generation of overall 100000000 data points in the control variable array. The integrative simulation results in the generation of the array of values for all the three state variable, for which the dynamic reconstruction, as shown in Figure 16, has been done against the original data obtained post the aspen dynamical simulation, which aids in the visual observation of the system behavior.

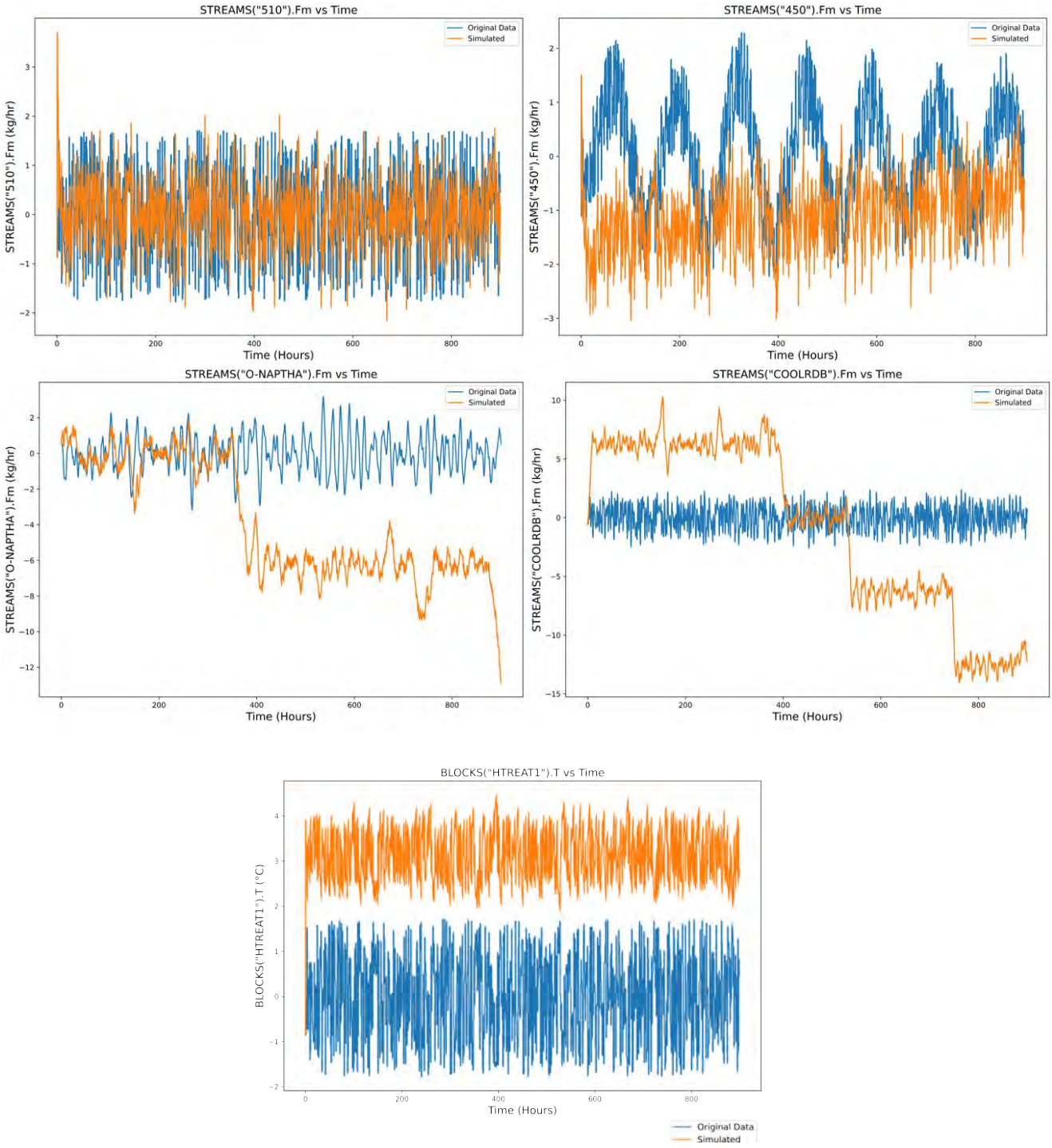

Figure S 15: Dynamical Reconstruction Plot of the Purification Node simulated upto 1000 hours

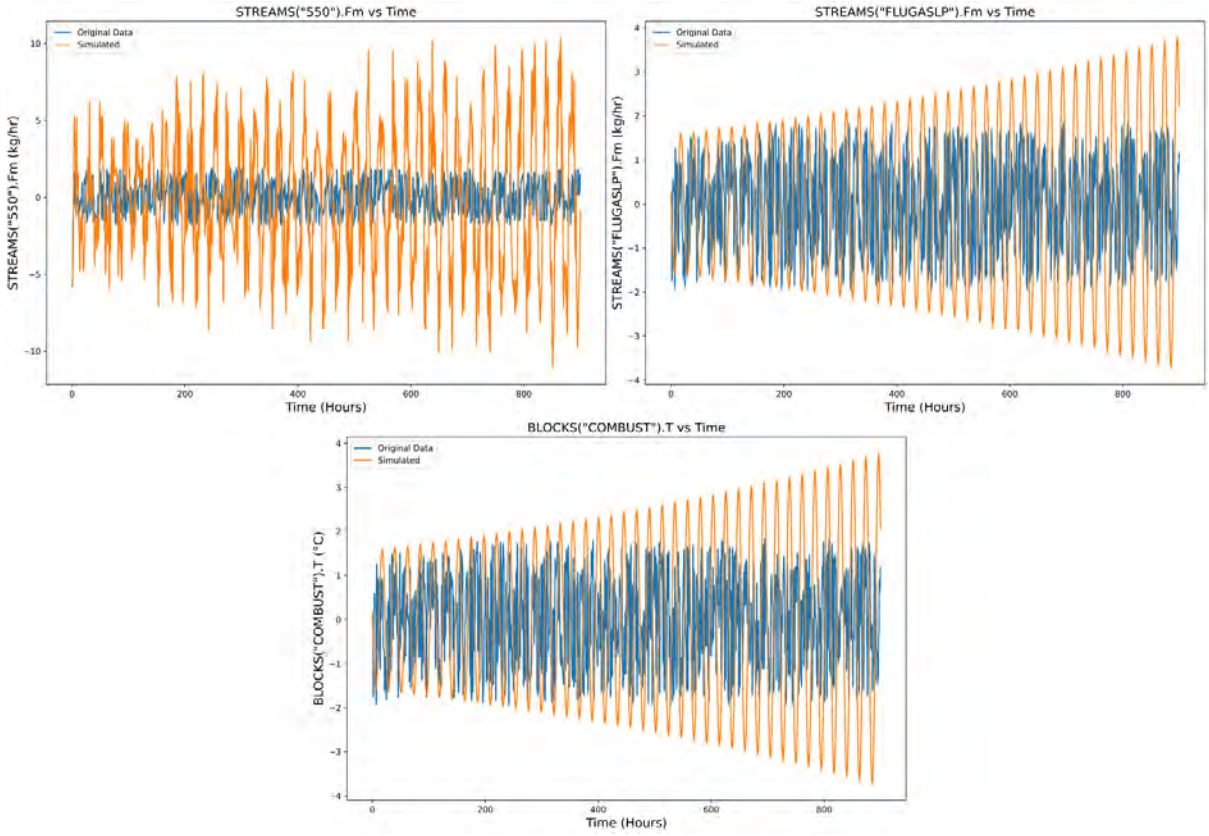

Figure S 16: Dynamical Reconstruction Plot of the Anaerobic Digestion Node simulated upto 1000 hours