



PATH WORKSHOP

PSE Academic Teaching Highway

Workshop: Teaching process modelling in chemical engineering –
perspectives from industry and academia

Contents

Introduction to mathematical modelling – Lumped modelling.....	3
Selected slides.....	3
Home assignment	11
Hands-on session	12
Objectives	12
Lumped modelling – multi-component	19
Home assignment	25
Hands-on session	26
Distributed modelling – introduction	37
Home assignment	43
Hands-on session	44
Flowsheeting – introduction	49
Hands-on session – Flowsheeting – solids blending	54
Brief intro into ProcessBuilder	57
Brief intro into Formulated products.....	59

Introduction to mathematical modelling – Lumped modelling

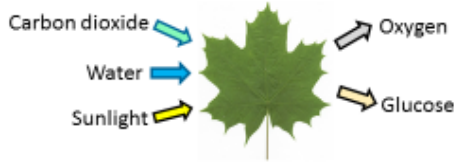
Goal : Introduce the basic concepts of lumped modelling.

Audience : Undergraduate


Selected slides

What is a system?

- A **system** is a set of connected things
 - For example, a tree is a system. A model of this system would try to predict how much water, sunlight and CO₂ it absorbs on a daily basis for example.



- Systems may have sub-systems inside them; *for example, a human being is a large number of interacting subsystems!*

 PSE Academic

4

What are mathematical models used for?

- Models help us develop a better scientific understanding of the behavior of a system
- They allow us to predict future behaviors or results and aid in efficient decision making



Σ PSE Academic

How do models really help us?

Safety: Models to accurately measure the extent of penetration of control rods, pressure, temperature and radioactivity at several points inside a nuclear plant boiler to maintain safe operating conditions

Economy: Financial models for oil price forecasting which homeowners rely on to decide when to purchase heating oil or whether to invest in energy-saving home improvements

Health: Modelling the growth rate of cancerous cells to determine drug dosage and oral absorption mechanisms

Environment: Modelling the capture of CO₂ emissions from power plants in compliance with the Environmental Protection Agency's standards

Society: Mathematical models for describing the spread of the 2014 Ebola epidemic in West Africa, along with modelling the speed of manufacturing the vaccine for Ebola and most efficient delivery system to optimize its eradication

2: https://commons.wikimedia.org/wiki/File:Carbon_sequestration-2009-10-07.jpg

6

Let us analyse a simple example...

- Suppose we have a large lake surrounded by houses, and a ship tied to a pier
- Rain causes the level of the lake to rise. Directly because rains falls in the lake and indirectly because rivers bring water from the area to the lake
- Evaporation and "leakage" to the ground water causes the level of the lake to lower
- To prevent the houses on the lake from flooding, the government has installed pumps that can pump water to the ocean
- The water level should typically be maintained at a level which allows people to get on-board their ship easily
- The pumps do not need to be capable of pumping out the maximum flow all the time (that would be too expensive). We can allow for some variation in the water level.

Real-life system:



Model:



Suppose we learn from the Meteorology department that there will be a massive rainfall. How much water should we pump out over to keep the level of the lake as constant as possible?

Σ PSE Academic

7

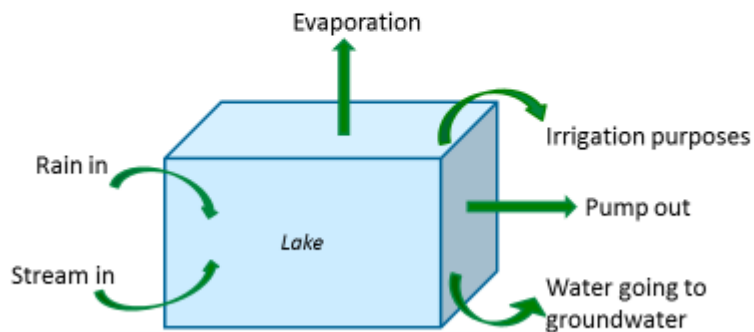
How do we develop a mathematical model for the lake?

Steps in modelling

1. Define the scope of the model
2. Draw the system and its boundaries
3. Get information and acquire knowledge
4. Write down assumptions
5. Write down the equations
6. Implement the equations and run simulation
7. Validate the model
8. Document the model

Step 2 – Draw the system and its boundaries

For this model, can you define your system and identify what crosses its boundaries?



Step 5 – Write down the equations

- The basis of (most) models are the balance equations
- Balance equations are “keeping track” of conserved properties
- We almost always start with the mass balance, which keeps track of the **conservation of mass** in the system
- We track what we call the hold-up, the amount held inside the system
- The general form of a balance is:

Change in hold-up over time = in – out + generation – destruction

Change in hold-up over time is typically called accumulation

Step 5 – Write down the equations

■ For the lake we would write:

– Change in hold-up over time: $\frac{dM}{dt}$

– In : $F_{rain} + F_{stream}$

– Out: $F_{evap} + F_{ground} + F_{farmers} + F_{pump}$

Note: There are no generation or destruction terms (since mass is conserved)

M	Mass of water in the lake (kg)
F_{rain}	Mass flow of rain into the lake (kg/s)
F_{stream}	Mass flow of stream from rivers coming into lake (kg/s)
F_{evap}	Mass flow of evaporating water (kg/s)
F_{ground}	Mass flow of water going to ground water (kg/s)
$F_{farmers}$	Mass flow of water taken out of lake by farmers (kg/s)
F_{pump}	Mass flow of water pumped out of lake to the ocean (kg/s)

Model analysis – Degrees Of Freedom (DOF)

- A Degree Of Freedom analysis helps us find make sure we have the same number of unknown variables and equations.

$$\text{DOF} = \text{No. of unknown variables} - \text{No. of equations} - \text{No. of assigned variables}$$

- In order to solve a system of equations uniquely, we require the degree of freedom to be zero
- This tells us the **minimum amount of information** (i.e., independent variable values) required to describe a unique state of a system

Model analysis – Output Set Assignment (OSA)

- An Output Set Assignment helps to better understand the model and if we cannot find an OSA, the system cannot be solved.
- In an OSA, we assign each variable to an equation from which that variable can be solved, using each equation and variable only once.
 - For example: $(y) : y = 6$
 - The (y) indicates that we are assigning the variable y to this equation (in this case we have no other option)

- Similarly: $(y) : y = x^2 + 2x - 5$, but this is equally valid:

$(x) : y = 5x - 3$

$(x) : y = x^2 + 2x - 5$
 $(y) : y = 5x - 3$

Check unit consistencies

- A unit check of the equations can help identify errors
- The units in each term should be the same (we cannot add apples and oranges)
- For our Lake model, we can clearly see that our units are consistent:

$$\frac{dM}{dt} = F_{rain} + F_{stream} - F_{evap} - F_{ground} - F_{farmers} - F_{pump}$$

[kg/s] [kg/s] [kg/s] [kg/s] [kg/s] [kg/s] [kg/s]

$$M = \rho V$$

[kg] = [kg/m³] * [m³]

$$V = Ah$$

[m³] = [m²] * [m]

Numerical software available for modeling

- Analytical software: Mathcad, Maple, Maxima, ...
- Spreadsheets: Excel, Lotus
- Matrix Algebra tools: Matlab, Scilab, Octave
- Flowsheeting tools: Aspen, HYSYS, PRO/II, ChemCad, Unisim, gPROMS ProcessBuilder
- Equation-Oriented tools: ACM, Modelica, gPROMS ProcessBuilder, gPROMS ModelBuilder, ...
- Fluid dynamic packages: Fluent, OpenFoam, Comsol, ...
- General software languages: Fortran, C++, Java, Python, ...

Commercial video games that simulate 'reality' use software engines which model simplified versions of physical phenomena



Effect of uncertainty

■ Consider the following case:

- If we find out that we received massive rainfall with a varying range over an extended time period of 10 hours (which also implies that there is no evaporation and that farmers did not take water out of the lake to irrigate their farmlands), **how does it impact your lake water level?** (Run a simulation to analyze the behavior of your system)

What measures will you suggest in order to address this situation?



Of course, Dexter could have used some laser beams to dry up the entire river and resolve the possibility of floods for good!

Simple control

- This can be implemented as a simple control algorithm (called proportional control or P-control):

$$F_{pump}(t) = B + K * (h(t) - SP)$$

- The symbols $F_{pump}(t)$ and $h(t)$ imply that F_{pump} and h are changing with time
- Here, the process variable that is controlled (called **Controlled Variable, CV**) is the level of the lake (h) and its desired value is referred to as its **set point (SP)**
- F_{pump} is the **Manipulated Variable (MV)** since it is adjusted in order to keep the water level at or near its set point
- K is called the **controller gain** and can be adjusted to make the controller output changes as sensitive as desired to the deviation of the controlled variable from the set point. We cannot set the gain too high, because this will cause too abrupt changes in the pump.
- The **bias, B**, is the controller output when the error (i.e., $h - SP$) is zero and is the base value

Change the model to include this p-controller and analyze the results

Home assignment

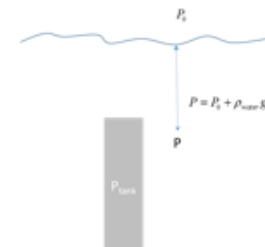
Home assignment: Lumped models – a diving tank model

Scuba divers use air from a tank on their back to breath under water.

Although the volume flow of air is fairly constant during a dive, the mass flow of air used in each breath is dependent on the depth the diver is at. This is because the pressure on the body of the diver increases with depth and to breath in, the pressure of the air coming in must be a little above the pressure of the surroundings. If the pressure is higher the density is higher and for the same volume flow that means the mass flow is higher.

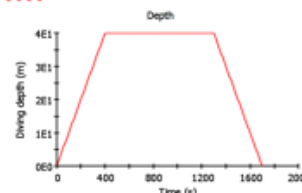


A fanatic modeler and scuba diver wants to find out how much air he will have left in his tank after a nice pre-scheduled dive.



Derive the equations for a model of the scuba diving tank. Assume the following:

- Tank has a volume of 18 liters
- Initial pressure in the tank, P_{tank} , at the start of the dive is 210 bar
- We assume we can use the ideal gas law to determine:
 - the density of the air that the scuba diver is breathing in as a function of the surrounding pressure, P
 - the density of gas inside the tank at the pressure in the tank, P_{tank}
- Assume a constant volume flow of 30 liter per minute
- Assume the following values (R_{gas} is 8.314 J/(mol K), g is 9.81 m/s²):
 - Temperature: 293 K
 - Molecular weight of air: 29 g/mol
 - Pressure of atmosphere, P_0 : 1 bar
 - Density of water: 1000 kg/m³
- Assuming the following dive profile (see example of one way to implement this in a gPROMS model below):
 - Dive at constant rate from surface to 40 m at a rate of 0.1 m/s
 - Stay at 40 meters for 15 minutes
 - Return to surface at a rate of 0.1 m/s



How much will the diver have left in his tank at the end of his dive?

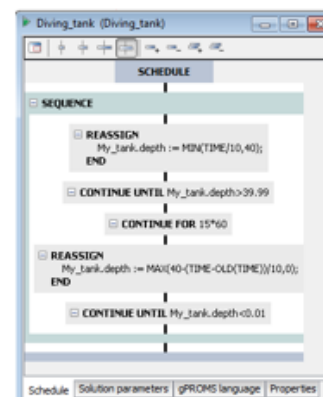
Example of gPROMS code to include diving schedule:

Note molar density is calculated as follows:

$$\rho_{\text{mole}} = \frac{P}{R_{\text{gas}} T}$$

Where ρ_{mole} is the density in mole/m³, P the pressure in Pa, R_{gas} the gas constant in J/(mol K) and T the temperature in K.

Figure of scuba diver taken from: <https://pixabay.com/en/scuba-diving-diver-diving-147683/>



Hands-on session

Background

In this exercise we will develop a model of a gas storage tank. The tank is normally used as a buffer to keep production constant. The tank has a safety system that releases the gas to a flare in case the pressure gets above 10 bar. Over a day the tank heats up due to solar radiation and changes in the surrounding temperature. This affects the pressure in the tank. In this exercise we will try to find what the flow rate into the tank needs to be to meet production with no flaring of the product while maintaining a pressure in the tank of at least 8 bar.

Exercise 1


Creating a lumped model of a gas storage tank

Objectives

By the end of the exercise, you will know how to:

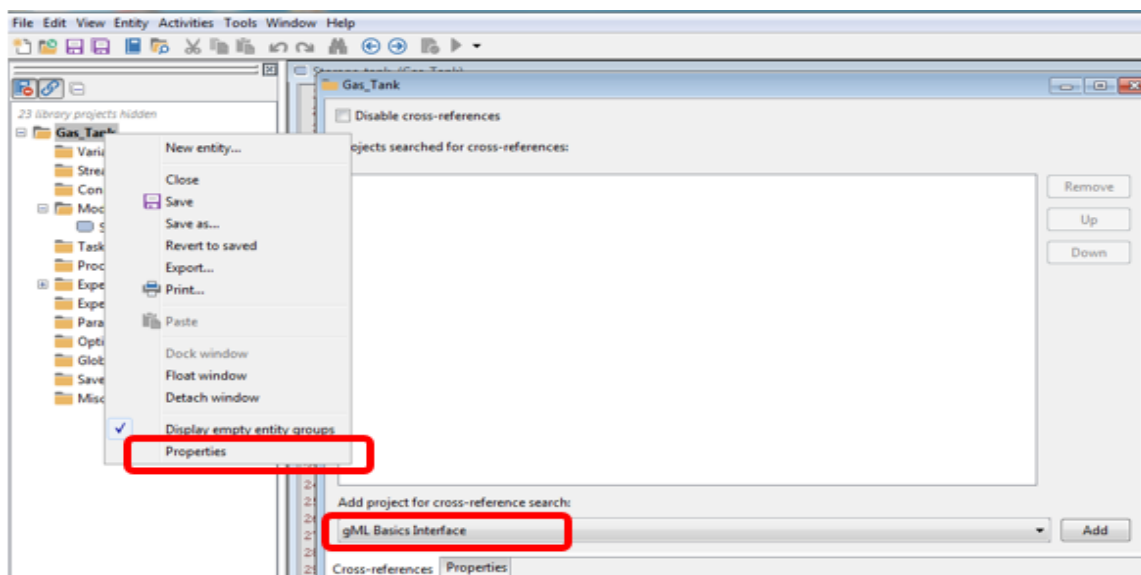
- Take advantage of the gML Basics library
- Build a model from scratch
- Write equations using the gPROMS language
- Run a dynamic simulation

Things to do

1. Open gPROMS ProcessBuilder 1.2.0 from the Windows start menu by clicking on the ProcessBuilder icon. When it opens, load the *gML Basics* library by clicking this icon  on the menu toolbar.
2. Create a new project file by going to the menu *File > New*. Save the newly created project under a different name. Make sure the name is highlighted in the project tree and go to the menu *File > Save as...* Choose an appropriate location and name for the file, e.g. *Gas_Tank.gPJ*.

Concept > gPROMS ProcessBuilder project tree. ProcessBuilder uses a Windows Explorer-like “project tree” structure to organise the different sections of the model description. To open an entity, navigate to it, by expanding the tree if necessary, and then double-click it. It will open in the workspace to the right.

3. Cross-reference the *gML Basics Interface* library by right-clicking on the project name, clicking on the *Properties*, and then in the *Cross-references* tab selecting the library of interest as shown in the following picture:



4. Create a new model entity by right-clicking on the “Models” folder (in the left hand pane) and selecting “New entity...” In the “New entity...” dialog box, provide a name for the new entity (e.g. *Storage_tank*) and make sure that the correct entity is selected (*MODEL*). Leave the “Use template?” box checked and click OK. A new model “*Storage_tank*” will appear. Click on the “*gPROMS language*” tab of this new model to see the template for *gPROMS language*.

A practical tip > Upon opening the model, there are greyed-out commented lines under each section to indicate the syntax convention. The template shows you a list of sections (e.g. **PARAMETER**, **DISTRIBUTION_DOMAIN**, **UNIT** etc.) and the general syntax to be used when writing a *gPROMS* model.

5. Complete the **PARAMETER** section for the *CSTR_lumped* model in the *gPROMS language* tab by adding the parameters given in the following table with grey background:

Symbol	Description	Units	<i>gPROMS</i> identifier	Size	Type
α	Flow coefficient	kg/bar ^{-0.5} /hr	alpha	-	REAL
R	Ideal gas constant	m ³ bar/kmol/K	gas_constant	-	REAL Default value:8.314E-2
A	Higher demand rate of gas	kg/hr	high_consumption	-	REAL
B	Lower demand rate of gas	kg/hr	low_consumption	-	REAL
M_w	Molecular weight of the gas	kg/kmol	molecular_weight	-	REAL
P_{atm}	Surrounding pressure	bar	P_atm	-	REAL Default value:1.01325
P_{open}	Open valve pressure	bar	P_open	-	REAL
P_{close}	Closed valve pressure	bar	P_close	-	REAL
V	Volume of the storage tank	m ³	Volume	-	REAL

PARAMETER

```

alpha                                AS REAL                                # Flow
coefficient

gas_constant                         AS REAL      DEFAULT 8.314E1    # Gas constant
high_consumption                     AS REAL                                # Higher

```

6. Complete the **VARIABLE** section by adding the variables using information given in the following table (note that the variable types required are available through the cross-referenced gML Basics library):

Symbol	Description	Units	gPROMS identifier	Size	Variable type
F_{in}	Inlet flowrate of gas into tank	kg/hr	F_in	-	mass_flowrate_kg_per_hr
$F_{release}$	Flowrate through the release valve	Kg/hr	F_release	-	mass_flowrate_kg_per_hr
F_{out}	Outlet flowrate of gas from the tank	kg/hr	F_out	-	mass_flowrate_kg_per_hr
M_{mass}	Amount of gas contained in the tank	kg	M_mass	-	mass_holdup_gML
M_{molar}	Moles of gas in the tank	mol	M_molar	-	molar_holdup_gML
t	Time	hr	OwnTime	-	no_type_gML
P	Pressure of the tank	bar	P	-	pressure_bar_gML
T	Temperature of outlet stream	K	T	-	temperature_gML

VARIABLE

```

...

M_mass                                AS mass_flowrate_gML    # Amount of gas contained
i

D                                     AS pressure_bar_gML      # Pressure of the tank

```

7. Add the **SELECTOR** section after the VARIABLE section as follows:

SELECTOR

```

ValveState AS (Open, Closed) DEFAULT Closed

```

8. Complete the **EQUATION** section with the mathematical representation of the gas storage tank model using the table below:

Description	Equation form	Size
Mass balance	Derive this balance equation	
Outlet flowrate of gas from the tank	<pre> IF OwnTime < 12 THEN F_out = low_consumption ; ELSE F_out = high_consumption ; END </pre>	-
Flowrate through the release valve	<pre> CASE ValveState OF WHEN Open: F_release = alpha * (P - P_atm) ; SWITCH TO Closed IF P < P_close ; WHEN Closed: F_release = 0 ; SWITCH TO Open IF P > P_open ; END </pre>	-
Ideal gas relation	$P.V = M^{molar}.RT$	-
Temperature of the tank	$T = -0.1t^2 + 2.4t + 20$	-
Molar holdup of gas in the tank	$M^{molar}.M_W = M^{mass}$	-
Time	$\frac{d(OwnTime)}{dt} = 1$	-

EQUATION

```

....
# Temperature of the tank
T = -0.1 * (OwnTime)^2 + 2.4 * OwnTime + 20 ;
...
```

9. Add an INITIAL section after this as below:

INITIAL

```
OwnTime = 0 ;
```

10. You have now created a generic model of the Storage tank. Next, you will set up a simulation using this MODEL. The input information specific to this simulation is provided in a PROCESS entity. To create a PROCESS around the newly created MODEL, right click on the *Storage_tank* model and select "Edit PROCESS" option. A new process *Storage_tank* will appear.

Concept > Relation between process and model entities. A mathematical representation of the system is defined within the model entity. Case specific information of the model is relayed in the process entity.

The model however needs to be referenced for the simulation and is done so in the UNIT section of the process which defines the model (or models) to be used in the simulation.

11. In the *gPROMS language* tab of the *Storage_tank* process, notice the predefined **UNIT** section where the model that will be used in the simulation is automatically defined with the name *Flowsheet*.

UNIT

Flowsheet AS Storage_tank

12. Next, you need to provide a value for parameters in the SET section of the process (add the **SET** keyword after the UNIT section) using the following specifications:

Symbol	gPROMS identifier	Values	Units
α	alpha	250	kg/bar ^{-0.5} /hr
A	high_consumption	250	kg/hr
B	low_consumption	150	kg/hr
M_w	molecular_weight	46	kg/kmol
P_{open}	P_open	10	bar
P_{close}	P_close	9	bar
V	Volume	100	m ³

SET

WITHIN Flowsheet DO

alpha := 250 ;

high_consumption := 250 ;

...

END

13. Assign values to the degrees of freedom in the ASSIGN section of the process (add the **ASSIGN** keyword after the SET section) according to the following information:

Symbol	gPROMS identifiers	Values	Units
F_{in}	F_in	200	kg/hr

ASSIGN

WITHIN Flowsheet DO

F_in := 200 ;

END

14. The SELECTOR specified in the model needs to be given an initial value. Add the gPROMS keyword **INITIALSELECTOR** between the ASSIGN and INTIAL section, and specify the valve as initially *Closed*.

```
INITIALSELECTOR

  WITHIN Flowsheet DO

    ValveState      := Closed ;

  END
```

15. The simulation requires INITIAL values for all differential variables (I.e., at time equals zero). Specify the following initial conditions by adding the **INITIAL** keyword after the ASSIGN section:

Symbol	gPROMS identifiers	Values	Units
<i>P</i>	P	9.5	bar

```
INITIAL

  WITHIN Flowsheet DO


    P              = 9.5 ;

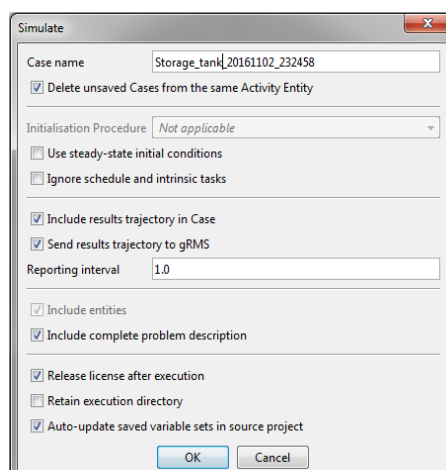
  END
```

```
SCHEDULE

  CONTINUE FOR 24
```

Concept > Defining schedules. Schedules can be generated by entering gPROMS language, by using a graphical interface or by using a combination of the two. Both methods are entirely equivalent and interchangeable: when a schedule is modified using the graphical interface, the equivalent change is automatically made to the language and, similarly, changes made to the gPROMS language are automatically applied to the graphical representation of the Schedule.

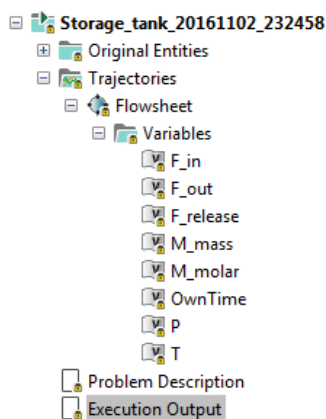
The simulation is now ready to be executed. To execute the simulation, select the process in the project tree and in the toolbar, click the simulate button (). The *Simulate* dialog box will appear. Accept the default settings and run the simulation by clicking OK.



Concept > Analysing simulation results. The execution output appears and gRMS is started. Also, note that a new case is being created – this is the blue folder in the project tree. By default, the name of the case is a concatenation of the process name (i.e. *CSTR_lumped*) plus a date and time stamp.

17. Results may be viewed at a glance within the gPROMS ProcessBuilder environment through expanding the *Trajectories* folder of the case created to reveal the variables within the UNIT Flowsheet. Take note of the values for the following variables:

- Temperature T
- F_{release} and P



Lumped modelling – multi-component

What if we want to track different things?

- In the lake example from the previous module, we looked at the amount of water in the lake, but what if we want to include:

- The number of fish in the lake?
- Birds that live around the lake and are dependent on the fish in the lake?
- A pollutant coming into the lake?
-

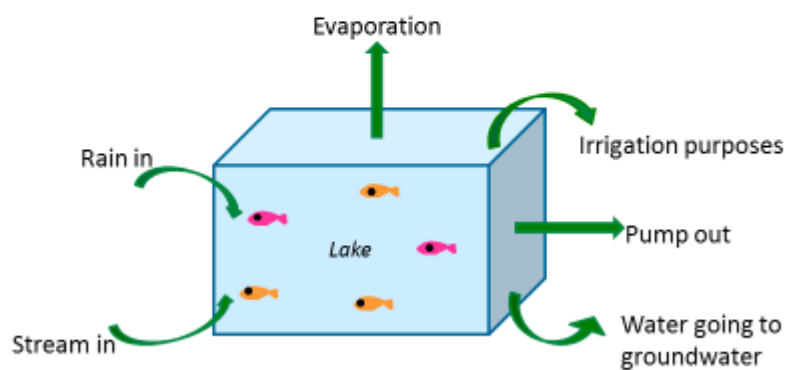


- In these cases, we introduce component balances:

- Typically, "components" are considered different molecules, but more generally, they refer to different "species" in the model

Step 2: Draw the system and its boundaries

Define the system and what crosses its boundaries for this model:



Step 5: Writing the component balance equations

- For each component, we can set-up a balance equation

Change in hold-up over time for a component i
= in-flow of i – out-flow of i + generation of i – destruction of i

- The change in hold-up over time is typically called **accumulation**
- For our lake example, we will thus set-up a balance for
 - The mass of water
 - The number of fish
- When we set-up a balance over a number (in a population), we call this a **population balance**

Step 5: Mass and population balance

- The *water mass balance* was given by:

$$\frac{dM}{dt} = F_{rain} + F_{stream} - F_{evap} - F_{ground} - F_{farmers} - F_{pump}$$

No generation or destruction terms (mass is conserved)

M	Mass of water in the lake (kg)
F_{rain}	Mass flow of rain into the lake (kg/s)
F_{stream}	Mass flow of stream into lake (kg/s)
F_{evap}	Mass flow of evaporating water (kg/s)
F_{ground}	Mass flow of water going to ground water (kg/s)
$F_{farmers}$	Mass flow of water out for irrigation (kg/s)
F_{pump}	Mass flow of water pumped out of lake (kg/s)

Please note that the mass balance ensures the conservation of mass, since mass can neither be created nor be destroyed

- The *fish population balance* is given by:

$$\frac{dN_{fish}}{dt} = B_{fish} - D_{fish}$$

Flow of fish is not considered

N_{fish}	Number of fish in the lake (#)
B_{fish}	Birth rate of fish in the lake (#/s)
D_{fish}	Death rate of fish in the lake (#/s)

Model analysis – Output Set Assignment (OSA)

- The OSA and DOF for our model thus looks like this:

PARAMETERS	
	$\$$
p	1
A	1
α	1
β	1
Total	4

Here we can observe that:

- The DOF is met (12 variables, and 6 equations and 6 assigned variables)
- The OSA is met (each variable is assigned to an equation or assignment only once)

Output Set Assignment (OSA)

VARIABLES		OSA		EQUATIONS	
	$\$$				$\$$
M	1	(1 M)	Mass Balance		1
V	1	(1 V)	Mass/Density		1
h	1	(1 h)	Volume/Area		1
F_{rain}	1	(1 N_{fish})	Population Balance		1
F_{stream}	1	(1 B_{fish})	Birth rate		1
F_{evap}	1	(1 D_{fish})	Death rate		1
F_{ground}	1				
F_{farmers}	1				
F_{pump}	1				
ASSIGNED					
N_{fish}	1	(1 F_{rain})	F_{rain}		1
B_{fish}	1	(1 F_{stream})	F_{stream}		1
D_{fish}	1	(1 F_{evap})	F_{evap}		1
		(1 F_{ground})	F_{ground}		1
		(1 F_{farmers})	F_{farmers}		1
		(1 F_{pump})	F_{pump}		1
Total	12				12
DOF	0				

Modelling of lake with fish and birds:

prey-predator model

What happens when our components are dependent on each other and we have a different time scale?

- Let us add birds into our model, which could then be formulated as what is known as a prey-predator model (*Lotka-Volterra equations*)
- The growth rate of birds thus relies on their predation rate of the fish
- A simplified prey-predator model is given by:

Fish \rightarrow Fish
Fish + Birds \rightarrow Birds
Birds \rightarrow Death



Now, let us track the level of the lake and the population balance of both species over a period of 100 years

Can you draw the system and its boundaries for this model?

Numerical solution of the model

- Therefore, the model we now have is:

$$\frac{dM}{dt} = F_{rain} + F_{stream} - F_{evap} - F_{ground} - F_{farmers} - F_{pump}$$

$$M = \rho V$$

$$V = Ah$$

$$\frac{dN_{fish}}{dt} = B_{fish} - D_{fish}$$

$$B_{fish} = \alpha N_{fish}$$

$$D_{fish} = \kappa N_{fish} * N_{bird}$$

$$\frac{dN_{bird}}{dt} = B_{bird} - D_{bird}$$

$$B_{bird} = \delta N_{fish} * N_{bird}$$

$$D_{bird} = \gamma N_{bird}$$

Mass balance of water

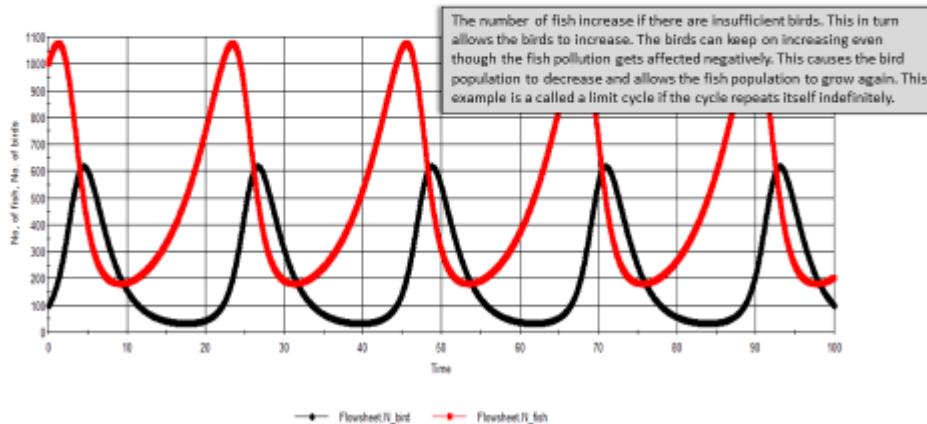
Population balance of fish

Population balance of birds

- When the initial population of fish and birds in our system are 1000 and 100 respectively, **what are the mass hold-up and population balance profiles of the species over a 100-year period** when: $\alpha = 0.2 \text{ year}^{-1}$, $\kappa = 0.001 \text{ year}^{-1}$, $\delta = 0.001 \text{ year}^{-1}$, $\gamma = 0.5 \text{ year}^{-1}$? *[all the 'rates' have to be on a yearly basis]*

Simulation plot

From the population balance plot below, we can see the interesting prey-predator dynamics for our model



How is Reaction Engineering of interest to us?

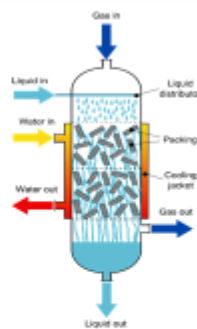
- We, as engineers, are interested in the design of processes for transforming lower value feedstocks to higher value products through chemical reactions
- Reaction engineering deals with chemical reactions and reactors, in order to answer questions like:

Can we manufacture a product economically?

What kind of reactor gives the maximum yield?

What should the reactor volume be to make a product at the desired rate?

How much heat should be removed from a reactor to maintain safe operating conditions?



2: Source: https://en.wikipedia.org/wiki/Trickle-bed_reactor

Σ PSE Academic

25

Reaction rate law

- The **reaction rate law** is an algebraic expression which links the rate of a reaction to the concentration of reactants and temperature
- For a general reaction: $aA + bB \rightarrow cC + dD$
 - The overall reaction rate is given by: $r = k(T)C_A^m C_B^n$
 - $k(T)$ is the **reaction rate constant**, which is predominantly a function of temperature
 - The exponents m and n , called **reaction orders**, are dependent on the reaction mechanism

Hence, the reaction rate per volume for a reaction **mechanism** is given by:

$$r = k \prod c_{i,r}^{|v_i|}$$

$c_{i,r}$ Concentration of component i that is involved in the reaction, i.e. the reactants (mol/m³)
 r Reaction rate (mol/m³/s)

Σ PSE Academic

27

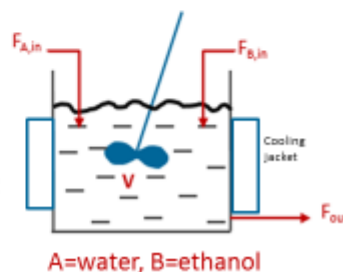
Introducing disturbances into a system

- Understanding system dynamics is vital since disturbances are observed fairly often in chemical process plants. But what really is 'disturbance'?

- Simple example: Suppose you leave for work at 8 AM everyday. However, one morning, your schedule could get delayed because of heavy rains, which would thus be a disturbance
- Therefore, a disturbance is essentially anything that causes a change in the behavior of a system

- Let us consider a simple example of:

- a mixing tank with two inlet streams: pure water and pure ethanol, each with a mass flowrate of 1 kg/s
- a cooling jacket with the coolant temperature being the same as that of the inlet streams (which are both at 300 K)



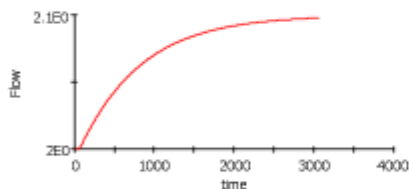
Σ PSE Academic

33

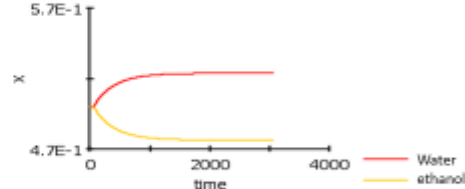
Inlet mass flowrate disturbance (step change)

- Let us introduce a 10% step change in the inlet mass flowrate of water (that is, from 1 to 1.1 kg/s)
- The profiles of some state variables change as follows:

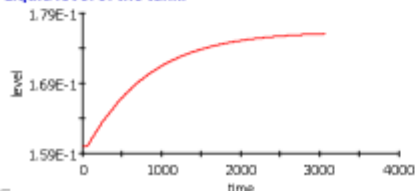
Outlet stream flowrate:



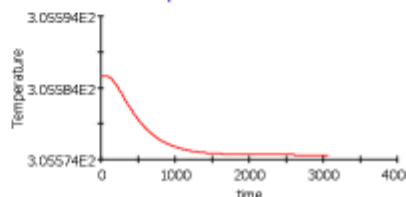
Outlet mass fractions:



Liquid level of the tank:



Outlet stream temperature:



Σ PSE Academic

35

Home assignment

Home assignment: Multiple component models – Prey-predator model

Problem statement: Toxicity is a very concerning factor for fish kills. Let us consider a system of a lake (with fish and birds), where we will track three entities for a period of 100 years:

- Water level
- Fish population
- Bird population,



The holdup of water in the lake is based on:

- Rainfall at a rate of 5346 kg/s
- Mountain stream coming in at a rate of 180 kg/s
- Evaporation rate of 1.543 kg/s
- Water used for irrigation purposes at a rate of 20 kg/s
- Seepage into ground water at a rate of 10 kg/s
- Water being pumped out at a rate to avoid flooding of the area

This flooding can be prevented only if the water level in the lake is kept within 10-10.25 meters

The fish-bird interrelationship obeys a Lotka-Volterra model, the coefficients of which have been specified below.

However, a factory nearby starts using this lake as a chemical dumping site for compound A, which reacts with the water in a pseudo first-order reaction to yield pollutant C. This pollutant affects the death of the fish at a rate of λ times the amount of pollutant.

This practice continues for two years after which it is terminated owing to environmental regulations. The population balance of the species in the lake is studied for another 100 years for research purposes.

Objective: How does the species population balance change over this span of 202 years? How would you ensure that the neighborhood does not get flooded?

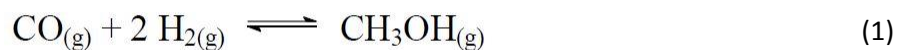
- What observations do you make from the population trends over this entire time span?
- How do you explain this behavior?

- | | |
|-----------------------------|---|
| <i>A</i> | Area of the lake [m^2] = 10^5 m^2 |
| <i>α</i> | Fish birth rate coefficient [1/year] = 0.2 |
| <i>B</i> | Predation rate [1/year] = 0.001 |
| <i>C</i> | Predator effectiveness coefficient [1/year] = 0.001 |
| <i>D</i> | Birds death rate coefficient [1/year] = 0.1 |
| <i>λ</i> | Fish death rate coefficient due to pollutant [$\#/\text{kg}\cdot\text{year}$] = $2\text{E-}2$ |
| <i>k</i> | Reaction rate constant [1/year] = $5\text{E}5$ |
| <i>G</i> | Gain for p-controller = $8\text{E}11$ |

Hands-on session

Background

The catalytic hydrogenation of carbon monoxide is nowadays the most important route to methanol production. A perfectly mixed gas-phase continuous stirred tank reactor is used to carry out the following methanol synthesis reaction:



The reaction is carried out in the presence of a solid catalyst held within a cage integrated with the stirring mechanism of the reactor system with a cooling jacket, as shown schematically below.

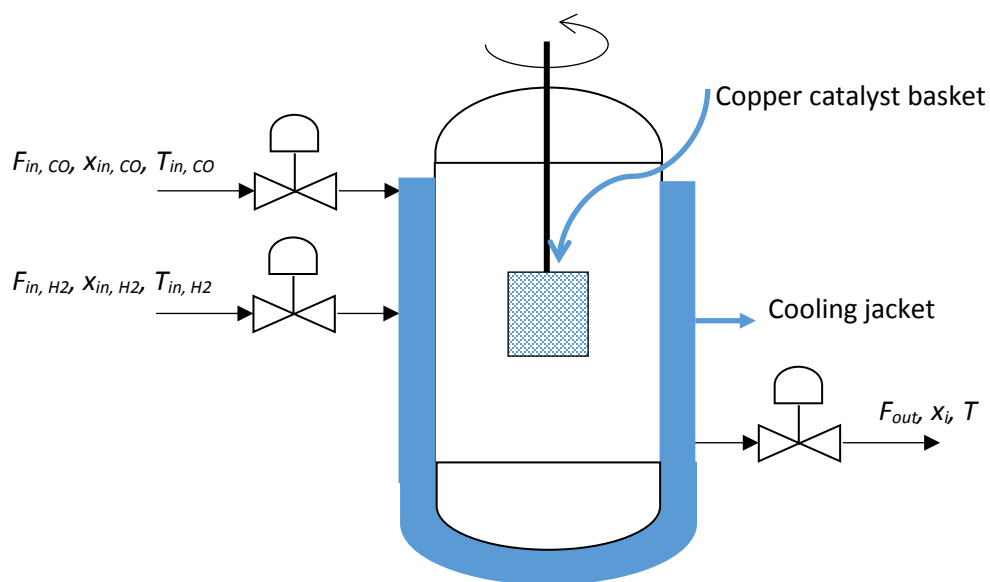


Figure: The heterogeneous gas/solid catalytic reactor system

For methanol synthesis over this Cu/ZnO catalyst, the surface reaction is rate-limiting, giving rise to a Langmuir-Hinshelwood-type rate expression [r is in kmol/kg-cat·hr and partial pressures in kPa]:

$$r = \frac{k_r \left(p_{\text{CO}} \cdot p_{\text{H}_2}^2 - \frac{p_{\text{CH}_3\text{OH}}}{K_{eq}} \right)}{\left(1 + K_{\text{CO}} \cdot p_{\text{CO}} + K_{\text{H}_2} \cdot p_{\text{H}_2} + K_{\text{CH}_3\text{OH}} \cdot p_{\text{CH}_3\text{OH}} \right)^3}, \quad (2)$$

where the thermodynamic equilibrium constant K_{eq} is found by:

$$K_{eq} = \exp^{-k_{eq}^1 + k_{eq}^2 / T} \quad (3)$$

with the other kinetic parameters (k_r , K_{CO} , K_{H_2} and K_{CH_3OH}) found through the following Arrhenius expressions:

$$k_r = 9.93 \times 10^{10} * \exp^{\frac{-135000}{RT}} \quad (4)$$

$$K_{CO} = 5.6 \times 10^6 * \exp^{\frac{-74000}{RT}} \quad (5)$$

$$K_{H_2} = 1.35 \times 10^4 * \exp^{\frac{-54955.5}{RT}} \quad (6)$$

$$K_{CH_3OH} = 5.97 \times 10^6 * \exp^{\frac{-74000}{RT}} \quad (7)$$

Objective

The purpose of this hands-on session is to develop a full mathematical model for a gas-phase continuous stirred tank reactor for the synthesis of methanol from hydrogen and carbon monoxide through the following stage-wise approach:

- Create a lumped model for dynamic calculations
- Introduce input disturbances to study the profiles of the system state variables

Exercise 1


Creating a lumped model of a gas-phase CSTR

Objectives

By the end of the exercise, you will know how to:

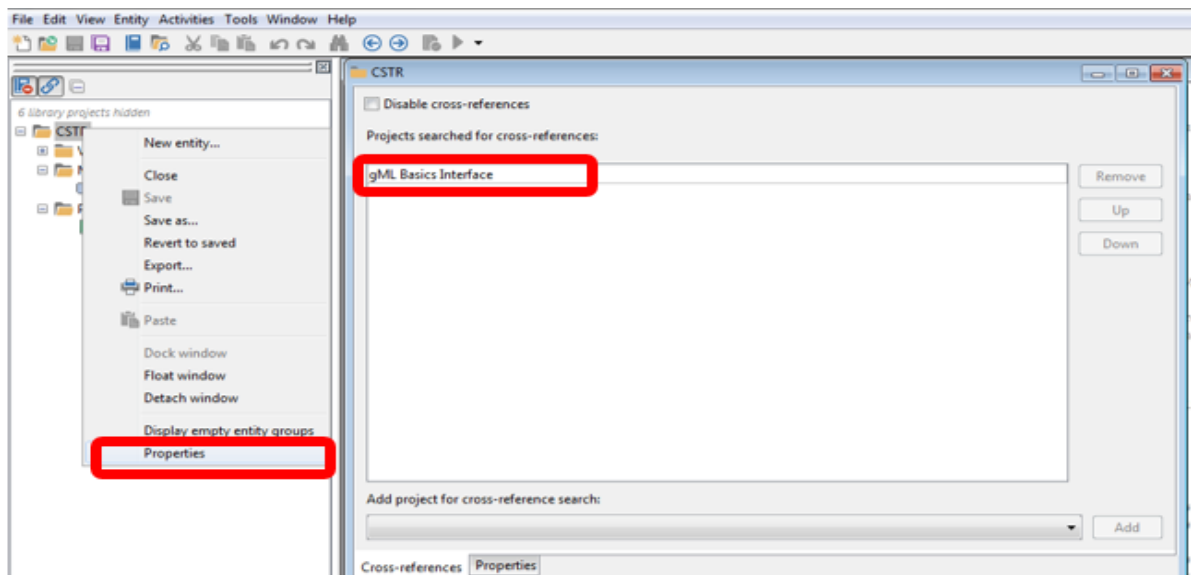
- Take advantage of the gML Basics library
- Write equations using the gPROMS language
- Run a dynamic simulation

Things to do

18. Open gPROMS ProcessBuilder 1.2.0 from the Windows start menu by clicking on the ProcessBuilder icon. When it opens, load the *gML Basics* library by clicking this icon  on the menu toolbar.
19. Create a new project file by going to the menu *File > New*. Save the newly created project under a different name. Make sure the name is highlighted in the project tree and go to the menu *File > Save as...* Choose an appropriate location and name for the file, e.g. *CSTR.gPJ*.

Concept > gPROMS ProcessBuilder project tree. ProcessBuilder uses a Windows Explorer-like “project tree” structure to organise the different sections of the model description. To open an entity, navigate to it, by expanding the tree if necessary, and then double-click it. It will open in the workspace to the right.

20. Cross-reference the *gML Basics Interface* library by right-clicking on the project name, clicking on the *Properties*, and then in the *Cross-references* tab selecting the library of interest as shown in the following picture:



21. Create a new model entity by right-clicking on the “Models” folder (in the left hand pane) and selecting “New entity...” In the “New entity...” dialog box, provide a name for the new entity (e.g. *CSTR_lumped*) and make sure that the correct entity is selected (MODEL). Leave the “Use template?” box checked and click OK. A new model “CSTR_lumped” will appear. Click on the “gPROMS language” tab of this new model to see the template for gPROMS language.

A practical tip > Upon opening the model, there are greyed-out commented lines under each section to indicate the syntax convention. The template shows you a list of sections (e.g. **PARAMETER**, **DISTRIBUTION_DOMAIN**, **UNIT** etc.) and the general syntax to be used when writing a gPROMS model.

22. Complete the **PARAMETER** section for the *CSTR_lumped* model in the *gPROMS language* tab by adding the parameters given in the following table with grey background:

Symbol	Description	Units	gPROMS identifier	Size	Type
Phys props	Physical property call	-	phys_prop	-	FOREIGN_OBJECT
NC	Components	-	components	-	ORDERED_SET
NF	Number of inlet streams	-	number_of_inlets	-	INTEGER
MW_i	Molecular weight of component i	kg/kmol	molecular_weight	components	REAL
v_i	Stoichiometry of component i	-	reaction_stoichiometry	components	INTEGER
$k_{r,0}$	Pre-exponential factor in k_r reaction kinetic expression	kmol/hr/kg _{cat}	pre_constant_r	-	REAL
Ea_r	Activation energy for k_r reaction	kJ/kmol	activation_energy_r	-	REAL
$k_{i,0}$	Pre-exponential factor in reaction kinetic expression for each component i	kPa ⁻¹	pre_constant	components	REAL
Ea_i	Activation energy for reaction of each component i	kJ/kmol	activation_energy	components	REAL
$k_e q^1, k_e q^2$	Equilibrium constants for K_{eq} kinetic expression	-	equilibrium_constants	2	REAL
P_{atm}	Atmospheric pressure	kPa	Patm	-	REAL Default value: 101.325
R	Ideal gas constant	kJ/kmol/K	ideal_gas_constant	-	REAL Default value: 8.314
m_{cat}	Weight of catalyst	kg	catalyst_weight	-	REAL
V	Volume of reactor	m ³	volume	-	REAL
UA	Overall heat transfer coefficient	kW/K	heat_transfer_coefficient	-	REAL
ΔH_{rxn}	Enthalpy of reaction	kJ/kmol	reaction_enthalpy	-	REAL

PARAMETER

```

reaction_stoichiometry AS ARRAY(components) OF REAL
# Stoi

catalyst_weight AS REAL
# Weig

...
```

23. Complete the **VARIABLE** section by adding the variables using information given in the following table (note that the variable types required are available through the cross-referenced gML Basics library):

Symbol	Description	Units	gPROMS identifier	Size	Variable type
M_i^{mass}	Amount of mass contained in the tank of component i	kg	mass_holdup	components	mass_holdup_gML
$F_{in,j}$	Inlet mass flowrate for stream j	kg/s	inlet_mass_flowrate	number_of_inlets	mass_flowrate_gML
$w_{in,i,j}$	Inlet mass fraction of component i for stream j	-	inlet_mass_fraction	Components, number_of_inlets	mass_fraction_gML
F_{out}	Overall outlet mass flowrate	kg/s	outlet_mass_flowrate	-	mass_flowrate_gML
w_i	Mass fraction of component i	-	mass_fraction	components	mass_fraction_gML
r	Molar reaction rate	kmol/s/ kg _{cat}	molar_reaction_rate	-	reaction_rate_kmol_per_s_kg_gML
M_T^{mass}	Total amount of mass in system	kg	total_mass_holdup	-	mass_holdup_gML
K_{eq}	Equilibrium constant	-	k_eq	-	no_type_gML
K_i	Kinetic parameters for component i	kPa ⁻¹	k	components	no_type_gML
k_r	Kinetic parameter for main reaction	kmol/hr /kg _{cat}	kr	-	no_type_gML
p_T	Total pressure	kPa	total_pressure	-	pressure_kPa_gML
v_p	Stem position of outlet valve	-	valve_position	-	no_type_gML
C_v	Valve proportionality constant	-	valve_constant	-	no_type_gML
p_i	Pressure of component i	kPa	partial_pressure	components	pressure_partial_kPa_gML
x_i	Molar fraction of component i	-	molar_fraction	components	molar_fraction_gML
M_T^{molar}	Total amount of moles in system	kmol	total_molar_holdup	-	moles_kmol_gML
E_{tot}	Energy hold-up in the system	kJ	energy_holdup	-	energy_holdup_kJ_gML
$H_{in,j}$	Mass enthalpy of inlet streams	kJ/kg	inlet_mass_enthalpy	number_of_inlets	mass_specific_enthalpy_kJ_gML

H	Mass enthalpy of outlet stream	kJ/kg	mass_enthalpy	-	mass_specific_enthalpy_kj_gML
Q	Heat duty	kW	heat_duty	-	heat_duty_gML
T_{in}	Temperatures of inlet streams	K	inlet_temperature	number_of_inlets	temperature_gML
T	Temperature of outlet stream	K	temperature	-	temperature_gML
T_C	Coolant temperature	K	coolant_temperature	-	temperature_gML

VARIABLE

mass_holdup AS ARRAY(components) OF mass_holdup_gML
 inlet_mass_flowrate AS ARRAY(number_of_inlets) OF mass_flowrate_gML
 inlet_mass_fraction AS ARRAY(components, number_of_inlets) OF mass_fraction_gML

24. Complete the **EQUATION** section with the mathematical representation of the CSTR model using the table below:

Description	Equation form	Size
Mass balance	$\frac{dM_i^{mass}}{dt} = \sum_{j=1}^{NF} (F_{in,j} \cdot w_{in,i,j}) - F_{out} \cdot w_i + m_{cat} \cdot v_i \cdot r \cdot MW_i$	$i = \text{CO}, \text{H}_2, \text{CH}_3\text{OH}$
Langmuir-Hinshelwood reaction rate expression	$r = \frac{k_r \left(p_{\text{CO}} \cdot p_{\text{H}_2}^2 - \frac{1}{K_{eq}} \cdot p_{\text{CH}_3\text{OH}} \right)}{\left(1 + K_{\text{CO}} \cdot p_{\text{CO}} + K_{\text{H}_2} \cdot p_{\text{H}_2} + K_{\text{CH}_3\text{OH}} \cdot p_{\text{CH}_3\text{OH}} \right)^3}$	-
Equilibrium constant	$K_{eq} = \exp \left(\frac{-k_{eq}^1 + k_{eq}^2}{T} \right)$	-
Reaction constant	$k_r = k_{r,0} \cdot \exp \left(\frac{-E_{a_r}}{RT} \right)$	-
Reaction constants	$K_i = k_{i,0} \cdot \exp \left(\frac{-E_{a_i}}{RT} \right)$	$i = \text{CO}, \text{H}_2, \text{CH}_3\text{OH}$
Ideal Gas Relation	$p_T \cdot V = M_T^{molar} \cdot R \cdot T$	-
Outlet Flowrate	$F_{out} = \frac{V_p \cdot C_v}{\sqrt{T}} \cdot (p_T - p_{atm})$	-
Total mass holdup	$M_T^{mass} = \sum_{i=1}^{NC} (M_i^{mass})$	-
Total molar holdup	$M_T^{molar} = \sum_{i=1}^{NC} \left(\frac{M_i^{mass}}{MW_i} \right)$	-
Holdup to total Holdup relation	$M_i^{mass} = w_i \cdot M_T^{mass}$	$i = \text{CO}, \text{H}_2, \text{CH}_3\text{OH}$

Partial pressure calculation	$p_i = x_i \cdot p_T$	$i = \text{CO}, \text{H}_2, \text{CH}_3\text{OH}$
Mass fraction to molar fraction relation	$x_i \cdot \sum_{i=1}^{NC} \left(\frac{w_i}{MW_i} \right) = \frac{w_i}{MW_i}$	$i = \text{CO}, \text{H}_2, \text{CH}_3\text{OH}$
Energy balance	$\frac{dE_{tot}}{dt} = \sum_{j=1}^{NF} (F_{in,j} \cdot H_{in,j}) - F_{out} \cdot H + m_{cat} \cdot r \cdot (-\Delta H_{rxn}) - Q$	-
Holdup of energy	$E_{tot} = M_T^{mass} \cdot H$	-
Inlet mass enthalpy	$H_{in,j} = f(T_{in,j}, p_T, w_{in,i,j})$	$j = 2$
Outlet mass enthalpy	$H = f(T, p_T, w_i)$	-
Heat duty	$Q = UA \cdot (T - T_C)$	-

EQUATION

Mass balance

FOR i IN components DO

```
$mass_holdup(i) = SIGMA(inlet_mass_flowrate( ) *
inlet_mass_fraction(i, )) - outlet_mass_flowrate *
mass_fraction(i) + catalyst_weight * reaction_stoichiometry(i)
* molar_reaction_rate * molecular_weight(i) ;
```

END

....

25. You have now created a generic model of the CSTR. Next, you will set up a simulation using this MODEL. The input information specific to this simulation is provided in a PROCESS entity. To create a PROCESS around the newly created MODEL, right click on the *CSTR_lumped* model and select “Edit PROCESS” option. A new process *CSTR_lumped* will appear.

Concept > Relation between process and model entities. A mathematical representation of the system is defined within the model entity. Case specific information of the model is relayed in the process entity.

The model however needs to be referenced for the simulation and is done so in the UNIT section of the process which defines the model (or models) to be used in the simulation.

26. In the *gPROMS language* tab of the *CSTR_lumped* process, notice the predefined **UNIT** section where the model that will be used in the simulation is automatically defined with the name *Flowsheet*.

UNIT

Flowsheet AS CSTR_lumped

27. Next, you need to provide a value for parameters in the SET section of the process (add the **SET** keyword after the UNIT section) using the following specifications:

Symbol	gPROMS identifier	Values	Units
Phys prop	phys_prop	"Multiflash::mass:MIXTURE_PengRobinson.mf l"	-
NC	components	phys_prop.components	-
NF	number_of_inlets	2	-
MW_i	molecular_weight	phys_prop.molecularweight	kg/kmol
ν_i	reaction_stoichiometry	[-1, -2, +1]	-
$k_{r,0}$	pre_constant_r	9.93×10^{10}	kmol/hr/kg _{cat} t
Ea_r	activation_energy_r	135000	kJ/kmol
$k_{i,0}$	pre_constant	$[5.60 \times 10^6, 1.35 \times 10^4, 5.97 \times 10^6]$	kPa ⁻¹
Ea_i	activation_energy	[74000, 54955.5, 74000]	kJ/kmol
$k_e q^1, k_e q^2$	equilibrium_constants	[5, 500]	-
m_{cat}	catalyst_weight	10	kg
V	volume	5	m ³
UA	heat_transfer_coefficien t	1.2	kW/K
ΔH_{rxn}	reaction_enthalpy	-91000	kJ/kmol

SET

WITHIN Flowsheet DO

```

    phys_prop      :=
    "Multiflash::mass:MIXTURE_PengRobinson.mf1" ;

    components     := phys_prop.components ;
    number_of_inlets := 2 ;
    MW             := phys_prop.molecularweight ;
    ...

```

28. Assign values to the degrees of freedom in the ASSIGN section of the process (add the **ASSIGN** keyword after the **SET** section) according to the following information:

Symbol	gPROMS identifiers	Values	Units
$F_{in,1}$	inlet_mass_flowrate(1)	7.78E-04	kg/s
$F_{in,2}$	inlet_mass_flowrate(2)	1.40E-05	kg/s
$w_{in,i,1}$	inlet_mass_fraction(,1)	[1, 0, 0]	kg/kg
$w_{in,i,2}$	inlet_mass_fraction(,2)	[0, 1, 0]	kg/kg
V_p	valve_position	0.5	-
C_v	valve_constant	2.88E-05	-
$T_{in,1}$	inlet_temperature(1)	475.15	K
$T_{in,2}$	inlet_temperature(2)	475.15	K
T_C	coolant_temperature	475.15	K

ASSIGN

WITHIN Flowsheet **DO**

```

inlet_mass_flowrate(1)      := 7.78E-4 ;
inlet_mass_flowrate(2)      := 1.40E-5 ;
inlet_mass_fraction( ,1)    := [1, 0, 0] ;
...
```

29. The simulation requires INITIAL values for all differential variables (i.e., at time equals zero). Specify the following initial conditions by adding the **INITIAL** keyword after the ASSIGN section:

Symbol	gPROMS identifiers	Values	Units
M_{CO}^{mass}	mass_holdup('CARBON MONOXIDE')	39.89	kg
$M_{H_2}^{mass}$	mass_holdup('HYDROGEN')	0.084	kg
$M_{CH_3OH}^{mass}$	mass_holdup('METHANOL')	5.756	kg
T	temperature	475.15	K

```
INITIAL
WITHIN Flowsheet DO
    mass_holdup('CARBON MONOXIDE') = 39.89 ;
    ...
END
```

30. At the bottom of the process, add the gPROMS keyword **SCHEDULE** and simulate the process for 3600 seconds.

```
SCHEDULE
CONTINUE FOR 3600
```

The simulation is now ready to be executed. To execute the simulation, select the process in the project tree and in the toolbar, click the simulate button (▶). The *Simulate* dialog box will appear. Accept the default settings and run the simulation by clicking OK.

Concept > Analysing simulation results. The execution output appears and gRMS is started. Also, note that a new case is being created – this is the blue folder in the project tree. By default, the name of the case is a concatenation of the process name (i.e. *CSTR_lumped*) plus a date and time stamp.

31. Results may be viewed at a glance within the gPROMS ProcessBuilder environment through expanding the *Trajectories* folder of the case created to reveal the variables within the UNIT Flowsheet. Take note of the values for the following variables:

- Outlet temperature T
- Mass holdup M^{mass} of each component.

32. Now, add 10% sinusoidal disturbances to the inlet mass flowrate of carbon monoxide and the heat duty by adding the following operating procedure to the process:

Time [s]:	0 - 3600	3,600 – 7200	7200-14400
$F_{in,l}[kg / s]$	7.78×10^{-4}	$7.78 * 10^{-4} + 7.78 * 10^{-5} * SIN(0.01*TIME)$	7.78×10^{-4}
$T_c[K]$	475.15	$475.15 + 47.515 * SIN(0.011*TIME)$	475.15

To do so, use the tasks CONTINUE and REASSIGN in the **SCHEDULE** section to describe the given operating procedure.

Concept > Defining schedules. Schedules can be generated by entering gPROMS language, by using a graphical interface or by using a combination of the two. Both methods are entirely equivalent and interchangeable: when a schedule is modified using the graphical interface, the equivalent change is

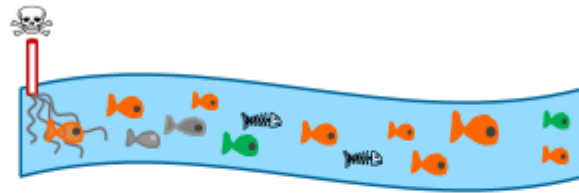
automatically made to the language and, similarly, changes made to the gPROMS language are automatically applied to the graphical representation of the Schedule.

```
SCHEDULE
SEQUENCE
  CONTINUE FOR 3600
  REASSIGN
    Flowsheet.inlet_mass_flowrate(1) := 7.78E-04 + 7.78E-5 * SIN(0.01 * TIME) ;
    Flowsheet.coolant_temperature := 475.15 + 47.515 * SIN(0.011 * TIME) ;
  END
  CONTINUE FOR 3600
  REASSIGN
    Flowsheet.inlet_mass_flowrate(1) := 7.78E-04 ;
    Flowsheet.coolant_temperature := 475.15 ;
  END
  CONTINUE FOR 7200
END
```

33. Execute the simulation.
34. Read the result trajectories in the case file and take note of the values for the mass holdup M^{mass} of each component and the outlet temperature in different times.

Distributed modelling – introduction

Example



- Suppose there is a river with fish
- A plant is discharging pollutant into the river
- The pollutant will cause the fish to get sick and die
- Government measured the concentration of the pollutant downstream of the plant, but they measure a safe concentrations of pollutants (not even causing to get significantly sick)
- We know however that the concentration of pollutant is different along the river, thus, a **distributed model** is needed to model the pollutant concentration profile

Mathematical background

■ Partial derivative

- The partial derivative of $f(t, x, y)$ with respect to t is:

$$\frac{\partial f}{\partial t} = \frac{\partial f(t, x, y)}{\partial t} \bigg|_{x, y} \quad (\text{Where } x \text{ and } y \text{ are kept constant})$$

■ Total derivative

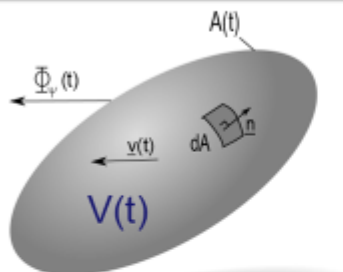
- The total derivative of $f(t, x, y)$ with respect to t is:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} \bigg|_{x, y} \frac{dt}{dt} + \frac{\partial f}{\partial x} \bigg|_{t, y} \frac{dx}{dt} + \frac{\partial f}{\partial y} \bigg|_{t, x} \frac{dy}{dt}$$

- In this lecture, we will use the following expression:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} \frac{dx}{dt} + \frac{\partial f}{\partial y} \frac{dy}{dt}$$

Derivation of balance equations – The formal way



Scalars:

V volume

A surface

dA surface element

Φ_ψ transport of quantity ψ across boundary

ψ Quantity of interest

Vectors:

\underline{n} normal vector (wrt surface)

\underline{v} velocity vector

$$\frac{d\Psi}{dt} = \Phi_\psi + \Pi_\psi$$

$$\Psi = \int_{V(t)} \psi dV$$

$$\Phi_\psi = - \int_{A(t)} \langle \phi_\psi \cdot d\mathbf{A} \rangle$$

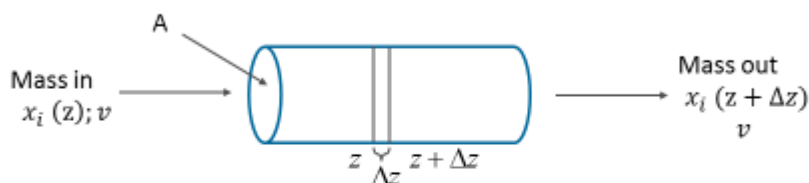
$$\Pi_\psi = \int_{V(t)} \pi_\psi dV = \int_{V(t)} \pi_\psi^{int} + \pi_\psi^{ext} dV$$

$$\frac{d}{dt} \left(\int_{V(t)} \psi dV \right) = - \int_{A(t)} \langle \phi_\psi \cdot d\mathbf{A} \rangle + \int_{V(t)} \pi_\psi^{int} + \pi_\psi^{ext} dV$$

Σ PSE Academic

7

Derivation of balance equations – An engineering way



General balance equations for this element

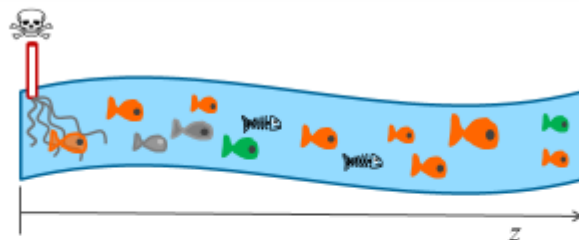
Accumulation = in – out + generation – consumption

- Generation and consumption can be reaction
- A : cross sectional area of the element (constant)
- v : velocity of flow
- x_i : mass concentration of component i
- Z : position along direction of flow

Σ PSE Academic

10

Example



Mass balance of pollutant $\frac{\partial C_{\text{pollutant}}}{\partial t} = -\frac{v \partial C_{\text{pollutant}}}{\partial z} - r_{\text{pollutant}} - 10^{-5} r_{\text{accu_sick}} \quad \forall z \in (0, L], B.C.: C_{\text{pollutant}}|_{z=0} = 10$

$r_{\text{pollutant}} = \alpha C_{\text{pollutant}}$ **Pollutant adsorption rate**

Mass balance of fish $\frac{\partial C_{\text{fish}}}{\partial t} = -\frac{v \partial C_{\text{fish}}}{\partial z} - r_{\text{accu_sick}} \quad \forall z \in (0, L], B.C.: C_{\text{fish}}|_{z=0} = 100$

Mass balance of sick fish $\frac{\partial r_{\text{accu_sick}}}{\partial t} = -\frac{v \partial r_{\text{accu_sick}}}{\partial z} + r_{\text{sick_accel}} \quad \forall z \in (0, L], B.C.: r_{\text{fish_sick}}|_{z=0} = 0$

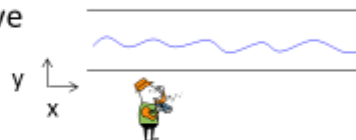
$r_{\text{sick_accel}} = \beta C_{\text{pollutant}} C_{\text{fish}}$ **Fish sickness rate**

Basis of velocity

Frame of reference – fixed volume (most common)

- Velocity is relative -> need a frame of reference!
- Typical, observer at a fixed location in the frame of reference, we use the partial derivative

- The velocity is relative to the observer



$$\frac{\partial \psi}{\partial t} = -\frac{\partial (\psi v + \phi_{\psi})}{\partial z} + \pi_{\psi}^{\text{int}} + \pi_{\psi}^{\text{ext}}$$

- Mass balance we have: $\psi = \rho, \quad \phi_{\rho} = 0, \quad \pi_{\rho} = 0$

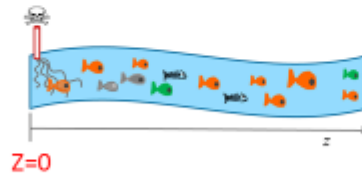
(by definition; see diffusion and (no mass generation)

in 1-D: $\frac{d\rho}{dt} = \frac{\partial \rho}{\partial z} \frac{dz}{dt} + \frac{\partial \rho}{\partial t} = -\frac{\partial(\rho v)}{\partial z} \xrightarrow{\text{Fixed frame } \frac{dz}{dt}=0} \frac{\partial \rho}{\partial t} = -\frac{\partial(\rho v)}{\partial z}$

Boundary conditions

Location of boundary condition: Non-linear wave theory

- The location of the boundary condition should be where the non-linear wave enters
 - In most practical cases this means where the fluid enters the system
 - More correctly it is where the information enters the system



Giving boundary condition at **Z=0**

Boundary conditions

Number of boundary conditions

- The number of boundary conditions is determined by the number of waves
 - In practical terms this means one for each partial derivative in a domain in each equation, e.g.
- $$0 = -\frac{\partial w}{\partial x} - \frac{\partial w}{\partial y} - \frac{\partial v}{\partial x} - \frac{\partial v}{\partial y}$$
- requires 2 (one for x domain and one for y domain)
- There exists no equivalent theory for non-hyperbolic systems (e.g. parabolic systems found in equations that have diffusion). These systems are treated as pseudo-hyperbolic .
 - The number of boundary conditions can (almost always) be found by counting the highest derivative in each domain, e.g.

$$0 = -\frac{\partial w}{\partial x} - \frac{\partial w}{\partial y} - \frac{\partial v}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial^2 u}{\partial x^2}$$

requires 3: 2 in x-domain and 1 in y domain

Boundary conditions

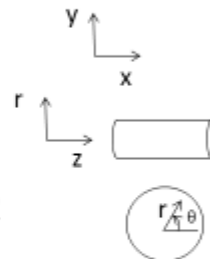
Type of boundary conditions

- The type only identifies the possibilities as mentioned earlier:
 - Type I : specify a variable value (also known as Dirichlet)
 - Type II: specify a derivative value (also known as Neumann)
 - Type III: specify a combination of the two (also known as Robin, e.g. Danckwertz)
- A first safe choice is to choose the flux (i.e. what is in the partial statement), because this is what is actually “flowing” into the system
 - Example:
$$\frac{\partial X}{\partial t} = -\frac{\partial(vX)}{\partial z}$$
 - Specify vX as boundary conditions

Coordinate system

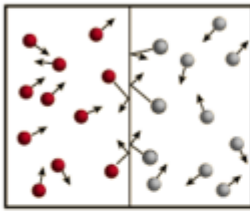
- Definition of the coordinate system required (2-D mass balance)

- Cartesian
$$\frac{\partial \rho}{\partial t} = -\frac{\partial(\rho v_x)}{\partial x} - \frac{\partial(\rho v_y)}{\partial y}$$
- Cylindrical
$$\frac{\partial \rho}{\partial t} = -\frac{\partial(\rho v_z)}{\partial z} - \frac{1}{r} \frac{\partial(r \rho v_r)}{\partial r}$$
- Spherical
$$\frac{\partial \rho}{\partial t} = -\frac{1}{r^2} \frac{\partial(r^2 \rho v_r)}{\partial r} - \frac{1}{r \sin \theta} \frac{\partial(\rho v_\theta \sin \theta)}{\partial \theta}$$



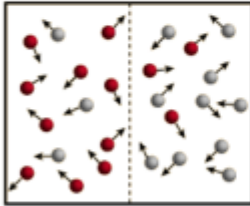
Remember that coordinate system can influence (differential) index (this is very rare; pendulum example)

Diffusion



Due to random walk (Brownian motion)
molecules all “walk” in different direction

Statistics show that the highest probability
is for the system to reach a well mixed
phase (equilibrium)



On a larger scale we call this process:

Diffusion

Diffusion of:
Different molecules (mixing)
Heat (heat mixing)
.....

Dynamic diffusion

Dynamic component balance:

No reaction, A constant, no convective flow

$$\frac{\partial(A\delta z c_i)}{\partial t} = J_i|_z - J_i|_{z+\delta z} = -D_{ij}A \frac{\partial c_i}{\partial z} \Big|_z - \left(-D_{ij}A \frac{\partial c_i}{\partial z} \Big|_{z+\delta z} \right) \rightarrow \frac{\partial c_i}{\partial t} = \frac{\partial}{\partial z} \left(D_{ij} \frac{\partial c_i}{\partial z} \right)$$

If D_{ij} is constant $\rightarrow \frac{\partial c_i}{\partial t} = D_{ij} \frac{\partial^2 c_i}{\partial z^2}$

Including convective flow:

(1-D, no reaction)

$$\frac{\partial c_i}{\partial t} = - \frac{\partial(v c_i)}{\partial z} + \frac{\partial}{\partial z} \left(D_{ij} \frac{\partial c_i}{\partial z} \right)$$

↑
↑
convection
diffusion

Home assignment

Home assignment: Distributed models – infectious disease spread

On a popular very long hike trail that has a steady stream of hikers a couple of contagious people start a hike. Due to their interaction with healthy or susceptible people, they become contagious and the contagious people become ill. After a couple of days the ill people get better and they can continue their hike and are susceptible again. The susceptible people have a healthy walking plus climbing speed, the contagious people slow down a bit and the ill people move very slowly.

A researcher has found that on this 100 km hike trail a fairly aggressive and fast moving disease appeared and he would like to understand where along this hike he could best set-up a small pharmacy shop. A friend and entrepreneur simply states that half way is best. We want to find out if that is true for this particular infection.



We will track 4 types of groups:

- Susceptible people that walk plus climb at a speed of 10 km/day
- Contagious people that walk plus climb at a speed of 5 km/day
- Ill people that walk plus climb at a speed of 1 km/day
- Recovered people that walk plus climb at a speed of 10 km/day

You can assume the following:

- Susceptible people
 - catch the disease from contagious people at a rate proportional to a constant a (2/day) times the number of susceptible times the number of contagious people divided by the total number of people.
 - Are increased by the rate of recovering people.
 - 10 susceptible people start at the trail basically continuously.
- Contagious people
 - Come from susceptible people as stated above.
 - Fall ill at a rate proportional to a constant b (0.5/day) times the number of contagious people.
 - 2 susceptible people start at the trail basically continuously.
- Ill people
 - Come from the contagious people as stated above.
 - Recover at a rate proportional to a constant d times the number of ill people.
 - No ill people start at the trail.
- Recovered people
 - Come in at the rate as described in ill people.
 - No recovered people start the trail (they are simply susceptible), because we also want to know how many people recovered over the hike.

Derive the 4 people balances. You can assume at the start of the experiment there are 10 susceptible people everywhere along the trail and no people from any other group. Check what happens over a period of 60 days. Can you explain the profiles in time and space of each group and the total number? Is that what you expected?

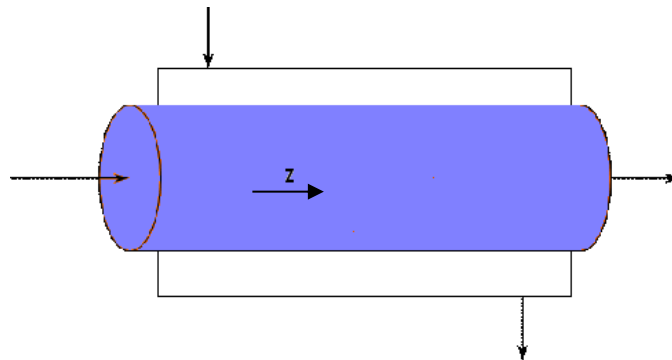
*Image taken from: <https://www.flickr.com/photos/archivesnz/15580284219/in/photostream/>

Hands-on session

Problem description

We want to simulate for 5 seconds the operation of a catalytic tubular reactor system in which a gas phase reaction takes place.

Process schematic



Process description

The oxidation of o-xylene to phthalic anhydride is carried out in an externally-cooled tubular reactor system. The reaction takes place in the gas phase of the reactor which is packed with catalyst particles. A eutectic mixture of molten potassium nitrate and potassium nitrite is used as a coolant.

Key modelling assumptions

- The tubular reactor behaviour is adequately described by a pseudo-homogeneous, one-dimensional mathematical model.
- The heat of reaction goes into the bulk fluid phase rather than into heating up the catalyst particles.
- The temperature in the cooling jacket is approximately uniform.
- The resistance to heat transfer occurs primarily between the reactor contents and the wall of the tube, the latter being effectively at the cooling medium temperature.
- Physical properties are constant over the range of conditions in the system.
- The ideal gas law holds in the gas phase.
- The reaction takes place in a dilute gas mixture with an inert carrier; the model considers the two reacting components only, denoted as A and B.

Mathematical model

Parameters

Symbol	Definition	gPROMS code	Unit
L	Reactor length	reactor_length	m
R	Reactor radius	reactor_radius	m
ρ_b	Bed density	bulk_density	kg m ⁻³
ρ_f	Fluid density	fluid_density	kg m ⁻³
C_{pf}	Fluid specific heat capacity	specific_heat_capacity	J kg ⁻¹ K ⁻¹
D_z	Axial diffusivity	axial_diffus	m ² s ⁻¹
D_r	Radial diffusivity	radial_diffus	
k_z	Axial thermal conductivity	axial_therm_conduct	
k_r	Radial thermal conductivity	radial_therm_conduct	W m ⁻¹ K ⁻¹
ε	Bed voidage fraction	Void	-
ΔH	Reaction enthalpy	reaction_enthalpy	J mol ⁻¹
A	Pre-exponential Arrhenius constant	arrhenius_constant	mol kg ⁻¹ s ⁻¹ Pa ⁻²
E	Activation energy	activation_energy	J mol ⁻¹
U	Heat transfer coefficient	heat_transfer_coeff	W m ⁻² K ⁻¹
R_g	Ideal gas constant	ideal_gas_constant	J mol ⁻¹ K ⁻¹
ρ_c	Coolant density	coolant_density	kg m ⁻³
C_{pc}	Coolant specific heat capacity	coolant_heat_capacity	J kg ⁻¹ K ⁻¹
V_c	Cooling jacket volume	jacket_volume	m ³
U	Overall heat transfer coefficient	heat_transfer_coeff	W m ⁻² K ⁻¹

Variables

Symbol	Definition	gPROMS code	Unit
$C_i(z)$	Concentration of component i	molar_conc(z)	mol m ⁻³
$T(z)$	Reactor temperature	temperature(z)	K
$R_{rxn}(z)$	Reaction rate	molar_reaction_rate(z)	mol kg ⁻¹ s ⁻¹
$Q(z)$	Heat flux absorbed by coolant	coolant_input_energy_flux(z)	W m ⁻²
p_i^{feed}	Feed partial pressure of component i	in_partial_pressure	Pa
T^{feed}	Feed temperature	in_temperature	K
u	Superficial gas velocity	Velocity	m s ⁻¹
T_w	Wall temperature	wall_temperature	K
F_c	Coolant mass flowrate	coolant_mass_flowrate	kg s ⁻¹
T_c^{in}	Coolant inlet temperature	coolant_in_temperature	K
T_c	Coolant temperature in the jacket	coolant_temperature	K

Equations

- Please derive the mass balance for the system. (The answer is at the end of this material)
Accumulation = in – out +/- generation

- Energy balance

$$\rho_f C_{pf} \frac{\partial T}{\partial t} = -\rho_f C_{pf} u \frac{\partial T}{\partial z} + k_z \frac{\partial^2 T}{\partial z^2} + \rho_b R_{rxn} (-\Delta H) - \frac{2}{R} Q(z) \quad z \in (0, L)$$

- Reaction rate

$$R_{rxn} = A e^{\frac{E}{R_g T}} (R_g T)^2 C_A C_B, \quad z \in [0, L]$$

- Coolant energy balance

$$\rho_c V_c C_{pc} \frac{dT_c}{dt} = F_c C_{pc} (T_c^{in} - T_c) + 2\pi R \int_0^L (Q) dz$$

- Coolant energy integration

$$Q(z) = U(T(z) - T_c)$$

Boundary Conditions

- Reactor entrance ($z = 0$)

$$-\varepsilon D_z \frac{\partial C_i}{\partial z} = u \left(\frac{P_i^{feed}}{R_g T^{feed}} - C_i \right) \quad i = A, B$$

$$-k_z \frac{\partial T}{\partial z} = \rho_f C_{pf} u (T^{feed} - T)$$

- Reactor exit ($z = L$)

$$\frac{\partial C_i}{\partial z} = 0, \quad i = A, B$$

$$\frac{\partial T}{\partial z} = 0$$

Model configuration

Parameter specifications

Parameter name	Value
L	3 m
R	0.0127 m
ρ_b	1300 kg m ⁻³
ρ_f	1.293 kg m ⁻³
C_{pf}	992 J kg ⁻¹ K ⁻¹
D_z	0.01 m ² s ⁻¹
D_r	0.01 m ² s ⁻¹
k_z	0.5 W m ⁻¹ K ⁻¹
k_r	0.5 W m ⁻¹ K ⁻¹
ε	0.35
ΔH	- 1.2 10 ⁶ J mol ⁻¹
A	11.45·10 ⁻³ mol kg ⁻¹ s ⁻¹ Pa ⁻²
E	113370 J mol ⁻¹
R_g	8.314 J mol ⁻¹ K ⁻¹
U	100 W m ⁻² K ⁻¹
ρ_c	2000 kg m ⁻³
C_{pc}	1239 J kg ⁻¹ K ⁻¹
V_c	0.1·10 ⁻³ m ³

Variable specifications

Variable name	Value
p_A^{feed}	1100 Pa
p_B^{feed}	21100 Pa
u	0.877 m s ⁻¹
T^{feed}	625 K
F_c	0.1 kg s ⁻¹
T_c^{in}	625 K

Initial conditions

$$C_A = 0, \quad C_B = 4, \quad z \in (0, L), \quad r \in (0, R)$$

$$T = 625, \quad z \in (0, L), \quad r \in (0, R)$$

$$T_c = 625$$

Things to do

- Open the template project file DS1D1_tubular_template.gPJ in ProcessBuilder and save it under a new name.
- In the equation section, add the mass & energy balances to the tubular reactor model
- Observe the PROCESS entity incorporating the data shown above. Use [BFDM,1,100] to discretise the axial domain.
- Fill in the initial section in PROCESS.
- Run the process.
- Use gRMS to plot the component concentrations and temperature as functions of time, reactor length.
- Copy and paste the existing PROCESS and rename it to Simulate_Tubular_NUG (for non-uniform grid). Change the specification for the axial domain discretisation to use a logarithmic transformation [BFDM, 1, 30, TRANSFORM(LOG, 4)]. Simulate the new process.
- Use gRMS to compare the temperature profile at time=5 obtained with the uniform grid (point 3) and the non-uniform grid (point 4).

ANSWERS:

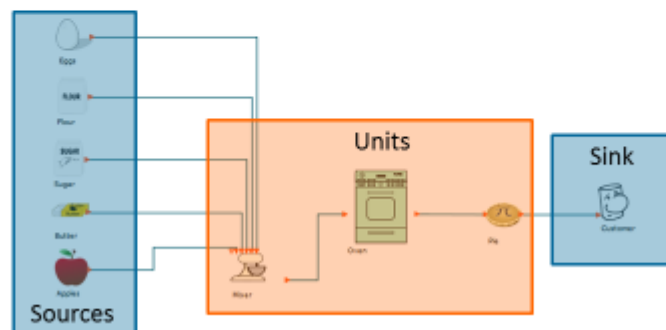
Mass balance for components:

$$\frac{\partial C_i}{\partial t} = -u \frac{\partial C_i}{\partial z} + \varepsilon D_z \frac{\partial^2 C_i}{\partial z^2} - \rho_b R_{rxn} \quad z \in (0, L), \quad i = A, B$$

Flowsheeting – introduction

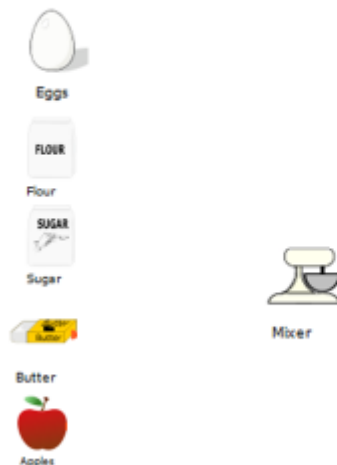
Elements on a flowsheet

- **Sources:** they bring endless material, energy or information into the flowsheet
- **Sinks:** they are an endless sink for material, energy or information
- **Units:** units typically change something, e.g. change the temperature or composition due to reaction or separation

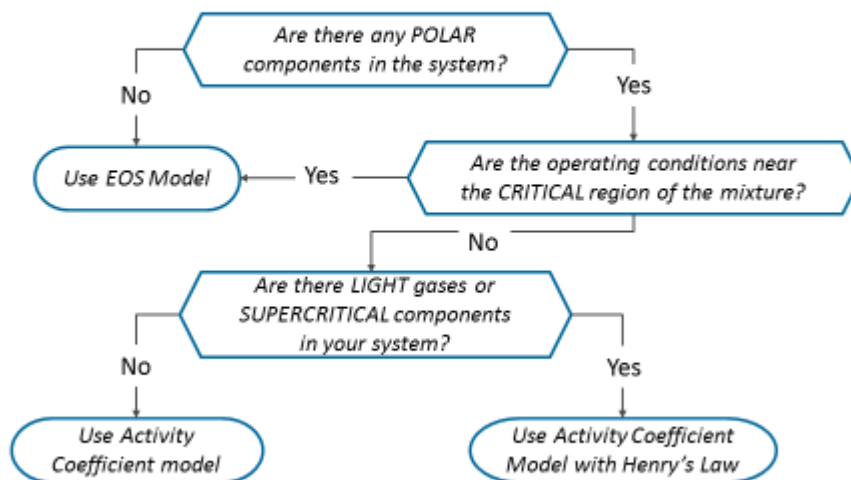


Building a flowsheet-unit by unit

- Add elements (sources, sinks, units) and change the settings



Selecting a thermodynamic model

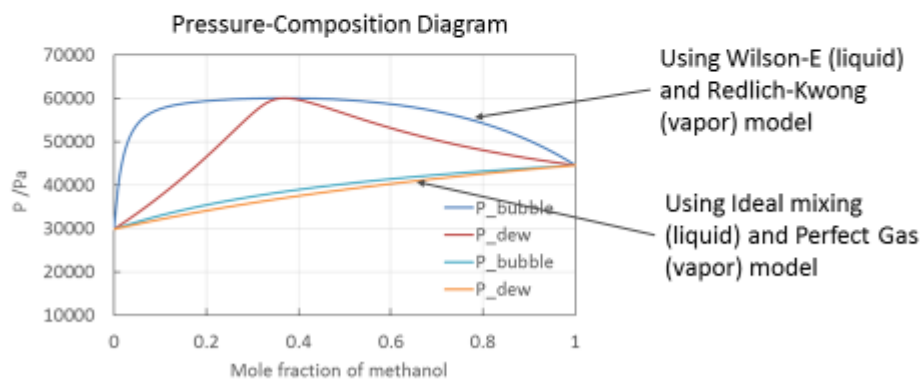


Σ PSE Academic Reference of the plot: The lab materials of course ChE 450 from Purdue University

16

Validation of thermodynamic model

- Thermo models need to be validated before using
- Different thermo models could lead to very different results
- Example: a solution with methanol and benzene

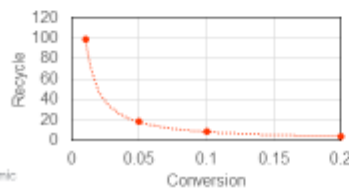
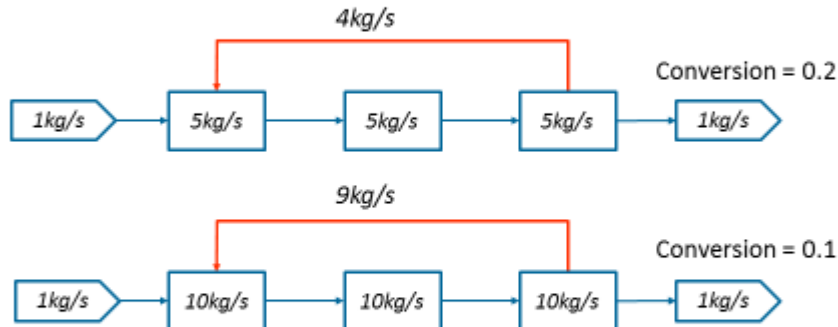


Σ PSE Academic

17

Recycles

■ Recycles could affect flowsheets significantly



The plot shows the size of recycle against conversion

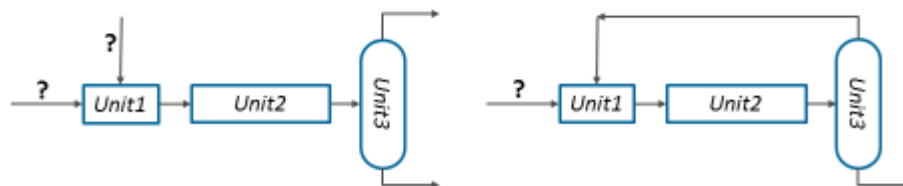
Degree of Freedom (DOF)

■ DOF analysis

- While building a flowsheet step by step, we need to make sure the DOF is satisfied
- This can be achieved by entering data in the dialogue boxes

■ In some cases the degree of freedom assignment may change

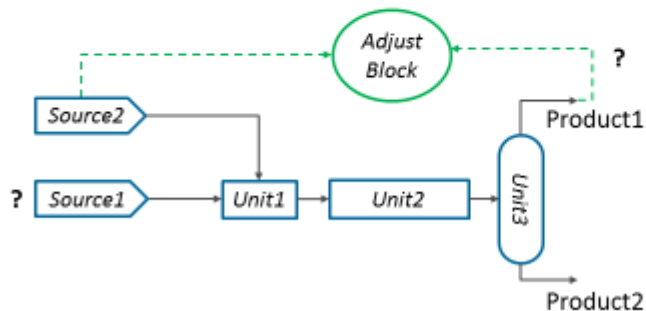
- Forming recycles



Degree of Freedom (DOF)

■ Changing the degrees of freedom assignment

- Forming recycles
- Adjust block in all tools
 - Usually, we specify source 1 and 2, simulate the rest of the process
 - If we know the production, or we have a target of the production (e.g. product 1), we can use an adjust block to back-calculate source (source2)



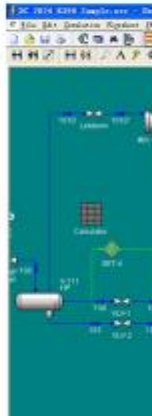
- Adjust block changes the DOF by specifying product 1 instead of source 2

Debugging Your Flowsheet

- If the problem is in a large flowsheet, cut the flowsheet down and introduce local sources and sinks to isolate where problem occurs
- Once we have fixed it, we go back to full flowsheet and check if this has resolved the problem

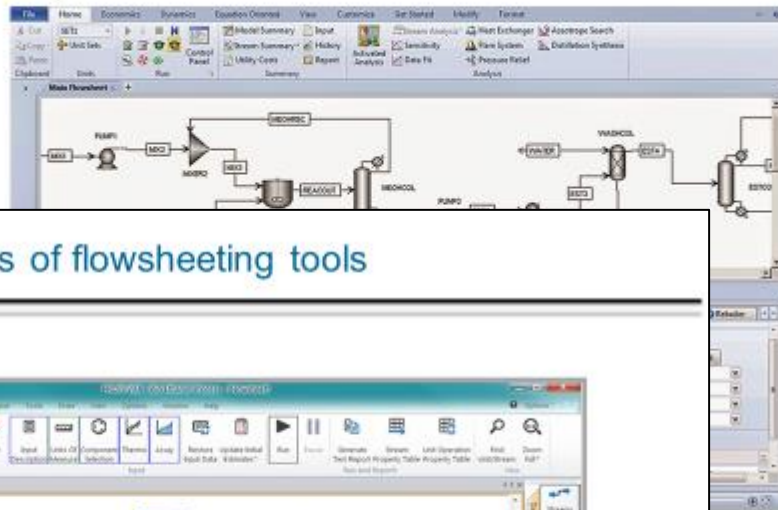
Some examples of flowsheeting tools

■ Unisim



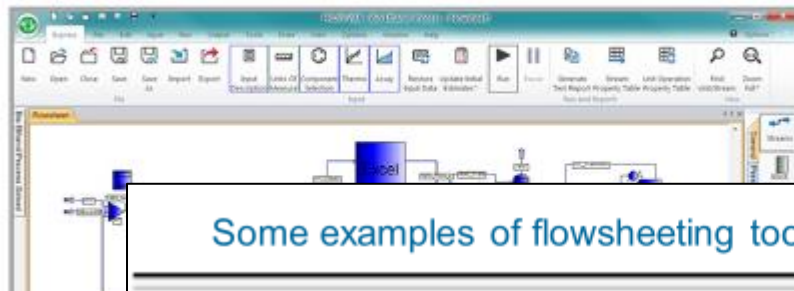
Some examples of flowsheeting tools

■ Aspen Plus



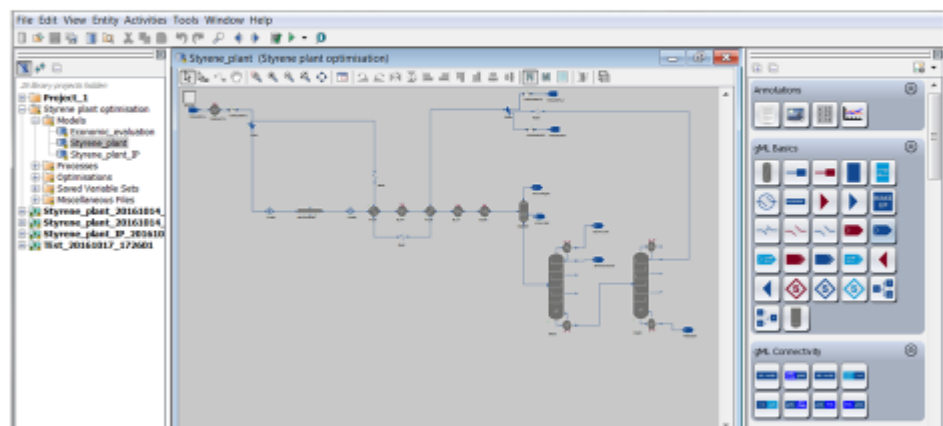
Some examples of flowsheeting tools

■ SimSci PRO/II



Some examples of flowsheeting tools

■ ProcessBuilder



Hands-on session – Flowsheeting – solids blending

Introduction

This exercise introduces the suite of continuous blender unit operations currently found in the 'gSOLIDS Distributed Mixing' library. The models can be used separately or in an integrated manner to dynamically assess the composition of the final product produced by the system. This information is key to understanding how disturbances in the ingredient feed rate affect the final API content in the tablets, as well as identifying when an 'off-spec' drug will be produced.

In this exercise, you will explore the features of the blender models and examine how disturbances in the feed to the blenders affect the continuous production of tablets in the feed frame.

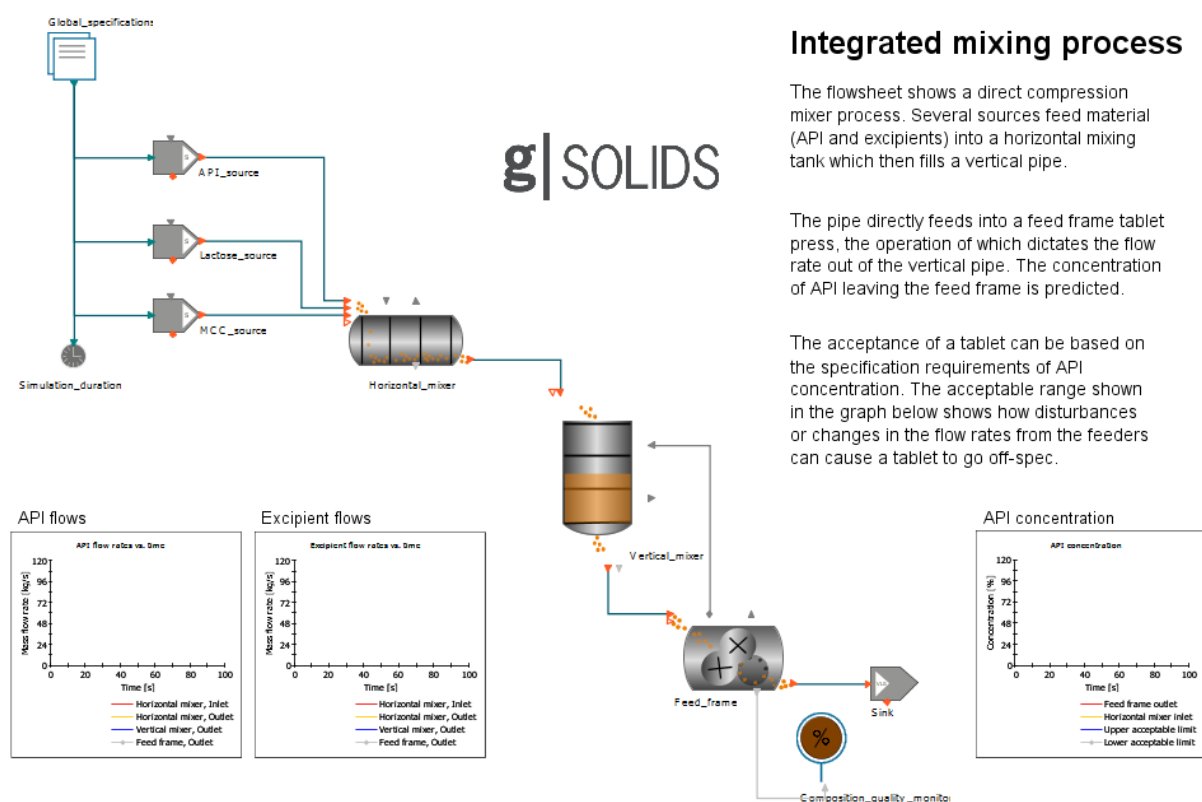


Figure 3.1. An example of a continuous blending flowsheet from gPROMS FormulatedProducts

Features of the models

There are three models to consider; a horizontal blender, a vertical hopper and a feed frame. Each model is governed by convection-dispersion equations with model specific features.



The horizontal blender allows consideration of material adhering to and detaching from the blender walls. The residence time distribution (RTD) of the powder in the blender (or indeed, the whole system) can be understood and the dispersive effects quantified such as how the amplitude of a disturbance in the feed rate will be dampened as it passes through the process.

The vertical hopper considers how the inlet and outlet flowrates affect the level in the vessel, which in turn affects the RTD of the powder in the system and the time it takes for a step change or disturbance to propagate through to the final tablet production.

The feed frame outlet is governed by the configuration of the tablet press and considers its rotational speed, the die fill height and the number of stations.

The feed frame operation, along with the initial flow rate through the horizontal blender, also determines the level of the vertical surge hopper and may be controlled to keep the hopper level constant.

Getting started

Open the project 'gSOLIDS Continuous blending' by clicking on the examples button . Double-click the flowsheet model 'Integrated_blending_process_pulse' and simulate the flowsheet by pressing .

Things to think about / try

1. **Observe the 'API flows' and 'API concentration' graphs on the simulated flowsheet. At what time is there a step change in the API flow? How long until there is a response at the outlet of the feed frame?**
2. **Double-click the sensor connected to the feed frame to open the report. What is the upper and lower bound of acceptable API concentration? How long does the final output concentration stay out of bounds? What happens to the icon when the concentration is out of range?**
3. **Try changing the amplitude and/or the length of time of the step change (by double-clicking on the 'disturbance' model) – how small (or short) does the disturbance have to be so the concentration never goes out of range (approximately)?**

Now open the flowsheet 'Integrated_blending_process_random' and simulate. Here the disturbance is random which is indicative of a problem or unstable feeder. When disturbances occur it is important to understand whether they propagate with enough ferocity to push the API concentration out of spec.

4. Again, observe the 'API flows' and 'API concentration'; notice how the amplitude of the disturbance is dampened by the dispersive effects within the blenders.

5. For how long, approximately, does the concentration fall out of acceptable bounds?

There are three further flowsheets to explore to consider the individual features of each of the blender models. Try running each of the flowsheets and explore the dialogs and reports of each by double-clicking on the model (before the simulation for dialogs and after the simulation for reports).

Hint: To access the model help documentation, click the "Help" button in the equipment dialogs, or the "Help documentation" links in the palette and model reports. This documentation describes the configuration of each model in more detail along with the governing equations implemented in the model.

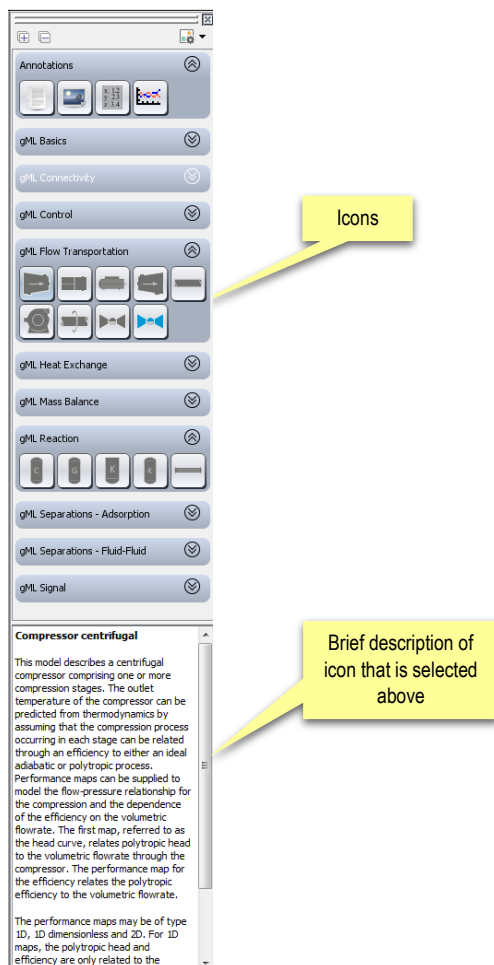
6. Continuous horizontal blender: how does the holdup of API on the surface change with the step change in the flow? Why does the change happen?
7. Vertical hopper: how does the icon change over the course of the simulation? Edit the 'Disturbance' dialog and change the set point to a higher and then lower value (remember to make sure the amplitude is not greater than the set point!) – What happens to the fill level?
8. Feed frame: the feed frame RPM is controlled to prevent accumulation –when the flow rate increases how does the RPM respond?


Brief intro into ProcessBuilder

1. Find the folder “gPROMS ProcessBuilder” on the USB provided. It is recommended that you transfer the folder to your local drive. The software can be used directly from the USB stick, though this will reduce the speed of execution.
2. Open the folder and double click on the file “gPROMS ProcessBuilder.bat” to launch the program.
3. Open the project file for the exercise of interest.

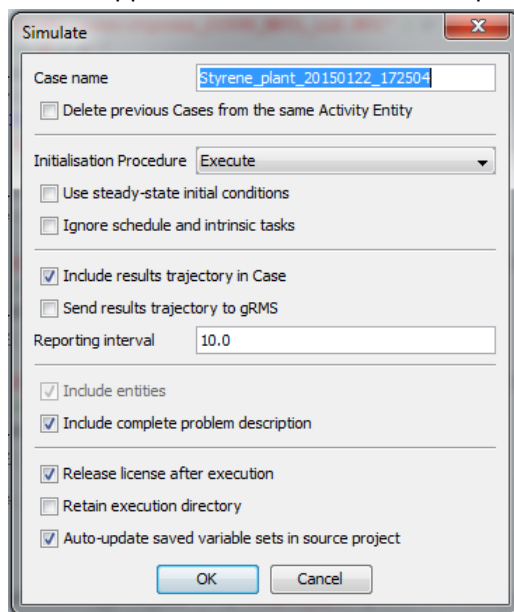
Concept > gPROMS ProcessBuilder project tree. ProcessBuilder uses a Windows Explorer-like “project tree” structure to organise the different sections of the model description. To open an entity, navigate to it, by expanding the tree if necessary, and then double-click it. It will open in the workspace to the right.

4. To create a new model, right-click on the *Models* folder in the project tree and select **New entity....** In the *New Entity* dialog box, provide the name for the new entity and make sure the right type of entity is selected (**MODEL**). Leave the *Use template?* box checked and click OK to create a new model. A new model will appear.
5. To access the libraries on the palette (if they are not already visible on the right pane), go to the menu **View > Palette** and tick the Palette. In the Palette you can expand a library folder by clicking on the arrow next to the library name. This will reveal a set of models as icons, as shown below.



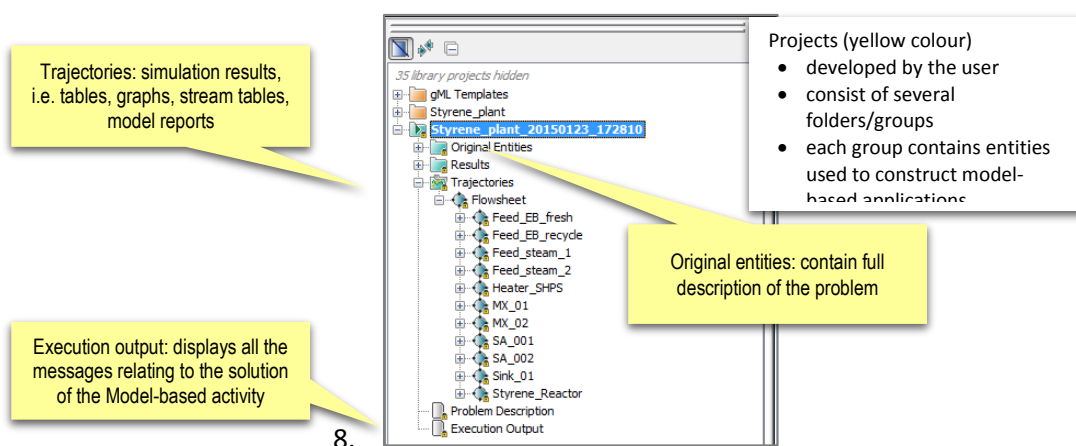
- To simulate a process, select the process of interest and click the icon  in the toolbar, or right-click it and select “Simulate” from the drop-down list, or just press F5 to simulate the PROCESS.

The following pop-up window will appear and have the time-stamped case name.



- Click OK to proceed with the simulation with default options.

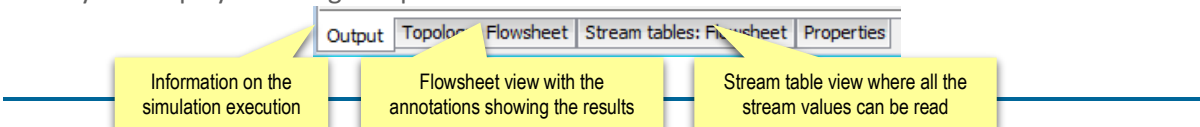
A time-stamped case file (turquoise colour) will be created in the project tree, as seen below:



8. This can be saved into your working directory (right-click and *Save as...*).

- Once the simulation completes, results can be examined by expanding the *Trajectories* folder of the case file to selected variables.

Concept > Execution Output window. All the messages relating to the solution of the Model-based activity are displayed during and post the execution. The tabs of the window are elaborated here:



Brief intro into Formulated products

- **Start gPROMS FormulatedProducts from the files provided:**
 - Find the folder “gPROMS FormulatedProducts” on the USB provided. It is recommended that you transfer the folder to your local drive. The software can be used directly from the USB stick, though this will reduce the speed of execution.
 - Open the folder and double click on the file “gPROMS FormulatedProducts.bat” to launch the program.
- The gPROMS FormulatedProducts libraries will load by default, as seen in the palette. These libraries contain core models that are shared between gCRYSTAL, gSOLIDS, and gCOAS.
 - Click on the Libraries icon to load the remaining libraries, as shown below. All four libraries will be used in this demonstration.
 - Check the boxes next to the libraries and click OK. Observe as they appear on the palette.

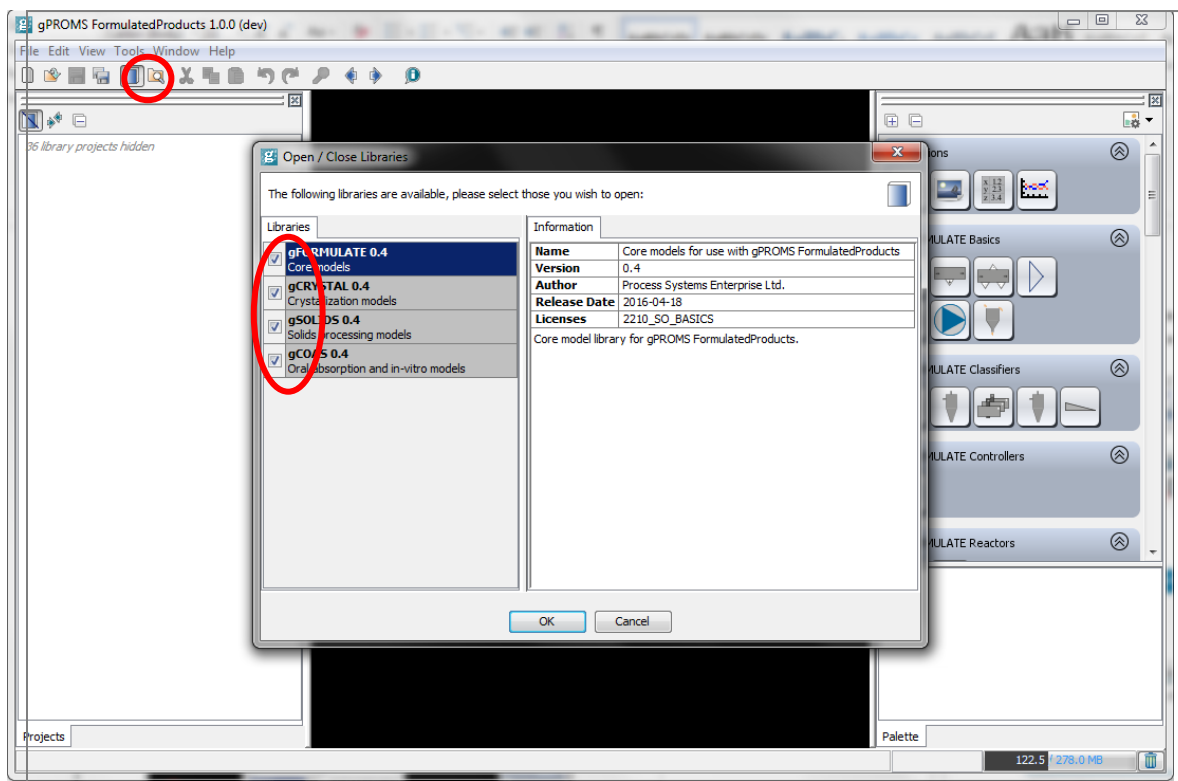



Figure 0.1. Opening gPROMS FormulatedProducts libraries

- To begin exploring, click on the Examples icon , and select an example from the dialog.
 - Expand the folders in the project tree to find flowsheets in the “Models” folder, and double click on an entity to open the flowsheet.

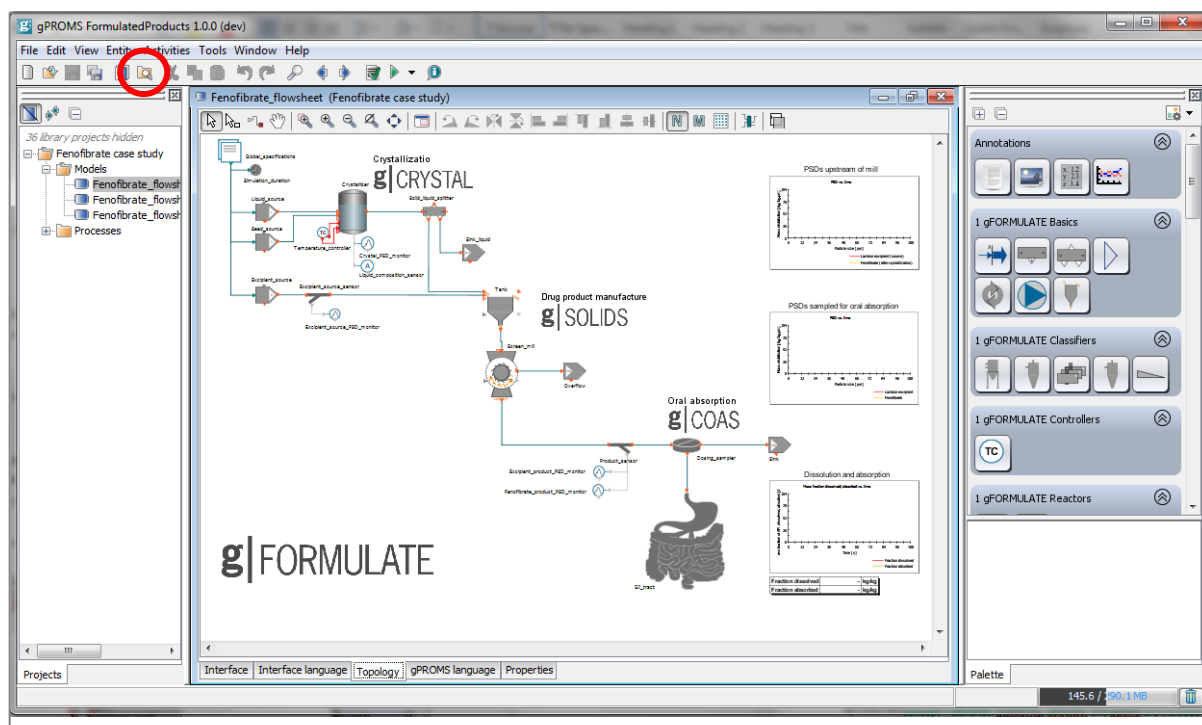


Figure 0.2. Opening gPROMS FormulatedProducts example flowsheets

g|FORMULATE Concept > Libraries. gCRYSTAL, gSOLIDS, and gCOAS are included in gPROMS FormulatedProducts as separate libraries that can be loaded individually or simultaneously.

