Tutorial on

Equation-Oriented Dynamic Simulation using EMSO

Argimiro R. Secchi
Rafael de Pelegrini Soares

Mar del Plata
August 12th, 2008
The Dream of a Process Engineer
– Fully-Integrated System and Tools –

- Data Reconciliation
- Gross-Errors Detection
- Steady-State Detection
- Data Validation
- Data Reading
- Sensitivity Analysis
- Inferences
- Simulation
- Model Updating
- Model Validation
- Model Identification
- Model Building
- Inferences
- Simulation
- Model Updating
- Model Validation
- Model Identification
- Model Building
- Assessment
- Diagnosis
- Prevention and Treatment
- Regulatory Control
- Supervisory Control
- RTO and DRTO
- Decision Making
- Configuration and Visualization
- Control and Optimization
- Task Planning and Sequencing
- Report Generator
- Management
- Knowledge Base
- Modeling and Simulation
- Data Processing
- Expert System
- Task Planning and Sequencing
- Report Generator
- Configuration and Visualization
- Management
Unified Modeling Environment
– A Necessary Condition –

Hierarchical Modeling

Process

Decision Making

Data Reconciliation

Inferences

Training, Safety

Advanced Control

Design

Optimization, RTO, DRTO

Integrated Environment

Dual Space
An Example of System Integration

In this context, the Knowledge Base is the link between the two spaces:

- **Model updating for D-RTO**
- **Inferences**
- **Model updating for NMPC**
- **Data pre-processing and dynamic data reconciliation**

**Production Planning**
- Feed specification, product and market

**D-RTO / RTO**
- \( y^*(t) \)
- \( u^*(t) \)

**NMPC**
- \( y(t) \)
- \( u(t) \)

**Process + Regulatory Control**
- \( Y(t) \)
- \( d(t) \)

**Model server**
- (rigorous, empiric, hybrid, reduced)
A movement from Sequential Modular to Equation-Oriented (EO) tools is clear.

Key advantages of EO:
- Models can be inspected
- Models can be refined or reused
- Same model as the source for several tasks: simulation, optimization, parameter estimation, data reconciliation, etc. → integrated environment

Some disadvantages:
- Lack of assistance in model development
- It is very difficult to fix ill-posed models
Outline

Equation-Oriented Dynamic Simulation using EMSO:

1. What is ESMO?
2. Building dynamic models
3. EMSO tutorial
4. Dynamic Degree of Freedom
5. Debugging techniques
1. What is EMSO?

- EMSO stands for “Environment for Modeling Simulation and Optimization”
- Development started in 2001, written in C++ language
- Available in Windows and Linux
- Models are written in an object-oriented modeling language
- Equation-oriented simulator and optimizer
- Computationally efficient for dynamic and steady-state simulations
- Continuous improvements through ALSOC project:

  [http://www.enq.ufrgs.br/alsoc](http://www.enq.ufrgs.br/alsoc)
Welcome to the ALSOC Project homepage

ALSOC is the acronym used to identify the project of a free environment for simulation, optimization, and process control. ALSOC is the acronym used to identify the project of a free environment for simulation, optimization, and process control. ALSOC is the acronym used to identify the project of a free environment for simulation, optimization, and process control.

The ALSOC Project is an effort to bring together university-industry through the standardization and distribution without cost of specifications and software tools among universities and partners companies. The ALSOC Project is an effort to bring together university-industry through the standardization and distribution without cost of specifications and software tools among universities and partners companies. The ALSOC Project is an effort to bring together university-industry through the standardization and distribution without cost of specifications and software tools among universities and partners companies.

Look here the list of institutions that participate and sponsor the project. Look here the list of institutions that participate and sponsor the project. Look here the list of institutions that participate and sponsor the project.

Project Goals

The main goals of the ALSOC Project are:

- to develop, maintain, and distribute specifications of a modeling language and a library of models for the synthesis, simulation, optimization and control of general processes (check the ALSOC OPEN LICENSE);
- to develop and maintain a state-of-the-art software and to distribute it at no cost to the universities and partners companies (check the ALSOC LICENSE);
- to certify third party solution and models as conforming to the developed standards.
EMS0 Key Features

- Open source library of models
- Object-oriented modeling
- Built-in automatic and symbolic differentiation
- Automatic checking and conversion of units of measurement
- Solve high-index problem
- Perform consistency analysis (DoF, DDoF, initial condition)
- Integrated Graphical User Interface (GUI)
- Building blocks to create flowsheets
- Discrete event handling
- Multitask for concurrent and real-time simulations
- Very modular architecture and support to sparse algebra
- Multiplatform: win32 and posix
- Interface with user code written in C/C++ or Fortran
- Automatic documentation of models using hypertexts and LaTeX
What can I do with EMSO?

- Steady-state simulations
- **Dynamic simulations**
- Steady-state optimizations
- Steady-state parameter estimations
- Dynamic parameter estimations
- Steady-state data reconciliations
- Process follow-up and inferences with OPC communication
- Build bifurcation diagrams (interface with AUTO for DAEs)
- Dynamic simulations with SIMULINK (interface with MATLAB)
- Add new solvers (DAE, NLA, NLP)
- Add external routines using the **Plugins** resource
Thermodynamic and Physical Properties – Plugin

Data bank with about 2000 pure compounds

Mixture properties calculation
How can I install EMSO?

1. Download EMSO and VRTherm packages from http://www.enq.ufrgs.br/alsoc
2. Run the setup programs
3. Run EMSO
4. Add the physical properties package using the Config Plugins option in the menu
5. Select and example and run it
To use a plug-in the user needs to register it through the menu

```
Config → Plugins
```

Windows plug-in is a DLL file, and Linux plug-in is a SO file
Integrated GUI
– Running an example –
2. Building Dynamic Models
– Where dynamic simulation is necessary –

• Batch and semi-batch processes
  (Analysis, Control, Dynamic optimization, Optimal design, Parameter estimation, Start-up operations)

• Dynamic real-time optimization (D-RTO)
  (NMPC, Plant-wide optimization, Product transitions, Model updating, Virtual analyzers)

• Advanced process control
  (Control structure design, Model reduction, Controllability and operability, Model-based control, Controller tuning, Nonlinear dynamics)

• Startups, shutdowns and transitions
  (Start-up strategies, Safety studies, Plant shutdown, Process transitions, Troubleshooting)

• Process intensification
  (Complex systems, Oscillatory motion, Reaction/separation processes, Auto-refrigerated reactors)

• Teaching and training
  (Classroom teaching, Operators training)
In Equation-Oriented (EO) simulators a model has:

- A set of model parameters (reaction order, valve constant, etc.)
- A set of variables (temperatures, pressures, flow rates, etc.)
- A set of equations (algebraic and differential) relating the variables

Problems in model building:

- Number of equations and variables does not match
- Equations of the model are inconsistent (linear dependence, etc.)
- The number of initial conditions does not match (dynamic simulation)
Building Dynamic Models
– Difficulties in Dynamic Simulation –

- Reliable models
- Truly standard interfaces and open source models
- High-Index DAE systems
- Large-Scale systems

Model consistency:
  - Degree of Freedom (DoF)
  - Dynamic Degree of Freedom (DDoF)
  - Units of measurement
  - Structural non-singularity
  - Consistent initial condition
Non-isothermal CSTR

\[ F_e, C_{Af}, C_{Bf}, T_f \]

\[ F_{we}, T_{we} \rightarrow V, T \]

\[ F_{ws}, T_w \]

\[ A \rightarrow \underset{k}{B} \]

\[ F_s, C_A, C_B, T \]
In a non-isothermal continuous stirred tank reactor, with diameter of 3.2 m and level control, pure reactant is fed at 300 K and 3.5 m$^3$/h with concentration of 300 kmol/m$^3$. A first order reaction occur in the reactor, with frequency factor of 89 s$^{-1}$ and activation energy of $6 \times 10^4$ kJ/kmol, releasing 7000 kJ/kmol of reaction heat. The reactor has a jacket to control the reactor temperature, with constant overall heat transfer coefficient of 300 kJ/(h.m$^2$.K). Assume constant density of 1000 kg/m$^3$ and constant specific heat of 4 kJ/(kg.K) in the reaction medium. The fully-open output linear valve has a constant of 2.7 m$^{2.5}$/h.
• perfect mixture in the reactor and jacket;
• negligible shaft work;
• \((-r_A) = k C_A\);
• constant density;
• constant overall heat transfer coefficient;
• constant specific heat;
• incompressible fluids;
• negligible heat loss to surroundings;
• \(\Delta\) (internal energy) \(\approx\) \(\Delta\) (enthalpy);
• negligible variation of potential and kinetic energies;
• constant volume in the jacket;
• thin metallic wall with negligible heat capacity.
Mass balance in the reactor

Overall:

\[
\frac{d(\rho V)}{dt} = \rho_f F_e - \rho F_s = \rho \frac{dV}{dt}
\]

\[
\frac{dV}{dt} = F_e - F_s \quad \text{(1)}
\]

Component:

\[
\frac{d\left(VC_A\right)}{dt} = V \frac{dC_A}{dt} + C_A \frac{dV}{dt} = F_e C_A f - F_s C_A - V (-r_A)
\]

\[
V \frac{dC_A}{dt} = F_e (C_{A f} - C_A) - (-r_A) V \quad \text{(2)}
\]

\[
\tau = \frac{V}{F_e} \quad \text{(3)}
\]
Energy balance in the reactor:

\[
\frac{d}{dt} \left[ \rho V (\hat{U} + \hat{K} + \hat{\phi}) \right] = F_e \rho \left( \hat{U}_f + P_f \hat{V}_f + \frac{v_f^2}{2} + g z_f \right) - F_s \rho \left( \hat{U} + P \hat{V} + \frac{v_s^2}{2} + g z_s \right) + q_r - q - w_s
\]

where \( \hat{H} = \hat{U} + P \hat{V} \)

\[
\frac{d}{dt} \left( \rho V \hat{H} \right) = \rho V \frac{d\hat{H}}{dt} + \rho \hat{H} \frac{dV}{dt} = F_e \rho \hat{H}_f - F_s \rho \hat{H} + q_r - q
\]

\[
\rho V \frac{d\hat{H}}{dt} = F_e \rho (\hat{H}_f - \hat{H}) + q_r - q
\]

\[
\rho V C_p \frac{dT}{dt} = F_e \rho C_p (T_f - T) + q_r - q
\] (4)
where

\[ q = U A_t (T - T_w) \]  \hspace{2cm} (5)

\[ q_r = (-\Delta H_r) V (-r_A) \]  \hspace{2cm} (6)

\[ (-r_A) = k C_A \]  \hspace{2cm} (7)

\[ k = k_0 \exp(-E/RT) \]  \hspace{2cm} (8)

\[ A = \pi D^2/4 \]  \hspace{2cm} (9)

\[ V = A h \]  \hspace{2cm} (10)

\[ A_t = A + \pi D h \]  \hspace{2cm} (11)

\[ F_s = x C\sqrt{\nu h} \]  \hspace{2cm} (12)

\[ x = f(h) \quad \text{Level control} \]  \hspace{2cm} (13)

\[ T_w = f(T) \quad \text{Temperature control} \]  \hspace{2cm} (14)
<table>
<thead>
<tr>
<th>variable</th>
<th>units of measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_e, F_s$</td>
<td>m³ s⁻¹</td>
</tr>
<tr>
<td>$V$</td>
<td>m³</td>
</tr>
<tr>
<td>$t, \tau$</td>
<td>s</td>
</tr>
<tr>
<td>$C_A, C_{Af}$</td>
<td>kmol m⁻³</td>
</tr>
<tr>
<td>$r_A$</td>
<td>kmol m⁻³ s⁻¹</td>
</tr>
<tr>
<td>$\rho$</td>
<td>kg m⁻³</td>
</tr>
<tr>
<td>$C_p$</td>
<td>kJ kg⁻¹ K⁻¹</td>
</tr>
<tr>
<td>$T, T_f, T_w$</td>
<td>K</td>
</tr>
<tr>
<td>$q_r, q$</td>
<td>kJ s⁻¹</td>
</tr>
<tr>
<td>$U$</td>
<td>kJ m⁻² K⁻¹ s⁻¹</td>
</tr>
<tr>
<td>$A_p, A$</td>
<td>m²</td>
</tr>
<tr>
<td>$h, D$</td>
<td>m</td>
</tr>
<tr>
<td>$C_v$</td>
<td>m².⁵ h⁻¹</td>
</tr>
<tr>
<td>$x$</td>
<td>–</td>
</tr>
<tr>
<td>$\Delta H_r, E$</td>
<td>kJ kmol⁻¹</td>
</tr>
<tr>
<td>$R$</td>
<td>kJ kmol⁻¹ K⁻¹</td>
</tr>
<tr>
<td>$k, k_0$</td>
<td>s⁻¹</td>
</tr>
</tbody>
</table>
variables: $F_e$, $F_s$, $V$, $t$, $C_A$, $C_{Af}$, $r_A$, $\rho$, $C_p$, $T$, $T_f$, $T_w$, $q$, $q_r$, $q$, $U$, $A$, $h$, $D$, $C_v$, $x$, $\Delta H_r$, $E$, $R$, $k$, $k_0$, $\tau \Rightarrow 27$

constants: $\rho$, $C_p$, $U$, $D$, $C_v$, $\Delta H_r$, $E$, $R$, $k_0 \Rightarrow 9$

specifications: $t \Rightarrow 1$

driving forces: $F_e$, $T_f$, $C_{Af} \Rightarrow 3$

unknown variables: $F_s$, $V$, $C_A$, $r_A$, $T$, $T_w$, $q$, $q_r$, $q$, $A$, $A_r$, $h$, $x$, $k$, $\tau \Rightarrow 14$

equations: 14


Initial condition: $h(0)$, $C_A(0)$, $T(0) \Rightarrow 3$

Dynamic Degree of Freedom (index < 2) = differential equations – initial conditions = $3 – 3 = 0$
Running EMSO

Open MSO file
Consistency Analysis

EQUATIONS

"Overall Mass Balance"
\[ \text{diff}(V) = \text{Inlet}.F - \text{Outlet}.F; \]

"Component Mass Balance"
\[ V \times \text{diff}(C_a) = \text{Inlet}.F \times (C_a - C_a) - (-r_A) \times V; \]

"Average Residence Time"
\[ \tau_a \times \text{Inlet}.F = V; \]

"Energy Balance"
\[ \text{ro} \times V \times \text{Cp} \times \text{diff}(T) = \text{Inlet}.F \times \text{ro} \times \text{Cp} \times (T - T) + qr - q; \]

"Heat Transfer Rate"
\[ q = U \times \text{At} \times (T - T_w); \]

"Reaction Heat Rate"
\[ qr = (-H_r) \times (-r_A) \times V; \]

"Reaction Rate"
\[ -r_A = k \times C_a; \]

"Arrhenius Equation"
\[ k = k_0 \times \exp(-E_a/(kT)); \]

"Geometry"
\[ A \times h = V; \]
\[ At = A + \pi D \times h; \]

"Valve Equation"
\[ \text{Outlet}.F = x \times CV \times \text{sqrt}(h); \]

"Perfect Mixture"
\[ \text{Outlet}.C_a = C_a; \]
\[ \text{Outlet}.T = T; \]
# Process with uncontrolled CSTR and multiple steady-states

FlowSheet CSTR_no_control

DEVICES
FEED as stream_cstr;
CSTR1 as CSTR;

CONNECTIONS
FEED to CSTR1.Inlet;

SET

# CSTR Parameters
CSTR1.R = 8.3144 * 'kJ/kmol/k';
CSTR1.U = 300 * 'kJ/h/mA2/k';
CSTR1.ro = 1000 * 'kg/mA3';
CSTR1.Cp = 4 * 'kJ/kg/K';
CSTR1.Hr = -7000 * 'kJ/kmol';
CSTR1.Ea = 664 * 'kJ/kmol';
CSTR1.ko = 89 * '1/s';
CSTR1.D = 1.2 * 'm';
CSTR1.Cv = 2.7 * 'mA2.5/h';

EQUATIONS
"Manipulated Variables"
CSTR1.x = 1;
CSTR1.Tw = 300 * 'K';

"Feed Stream"
FEED.Ca = 300 * 'kmol/mA3';
FEED.F = 3.5 * 'mA3/h';

# disturbance
if time < 50 * 'h' then
  "Feed Temperature" FEED.T = 300 * 'K';
else
  "Feed Temperature" FEED.T = 350 * 'K';
end

INITIAL
"Concentration" CSTR1.Ca = 50 * 'kmol/mA3';
"Level" CSTR1.h = 1.7 * 'm';
"Temperature" CSTR1.T = 570 * 'K'; # increase to 580 K to change steady-state

OPTIONS
TimeStep = 1;
TimeEnd = 100;
TimeUnit = 'h';
DAESolver(File = "dass1");

end
Building Dynamic Models
– Checking Units of Measurement –

```plaintext
123  CSTR1.ko = 89 * '1/s';
124  CSTR1.D = 3.2 * 'm';
125  CSTR1.CV = 2.7 * 'm^2.5/h';
126
127  EQUATIONS
128  "Manipulated Variables"
129  CSTR1.x = 1;
130  CSTR1.Tw = 300 * 'm';
131  "Feed Stream"
132  FEED.Ca = 300 * 'kmol/m^3';
133  FEED.F = 3.5 * 'm^2/h';
```

Compatible units

---

```plaintext
135  CSTR1.Tw [K] and 300*m [m] have incompatible units
136  FEED.F [m^3/s] and 3.5*(m^2/h) [m^2/h] have incompatible units
```

Incompatible units
EMSO has 3 main entities in the modeling structure

**FlowSheet** – process model, is composed by a set of **DEVICES**

**DEVICES** – components of a FlowSheet, an unit operation or an equipment

**Model** – mathematical description of a **DEVICE**
EMSO Tutorial
– Modeling Structure –

Model: equation-based

FlowSheet: component-based
The modeling and simulation of complex systems is facilitated by the use of the Object-Oriented concept.

The system can be decomposed in several components, each one described separately using its constitutive equations.

The components of the system exchange information through the connecting ports.
Parameters and variables are declared within their valid domains and units using types created based on the built-in types: **Real, Integer, Switcher, Plugin**

```plaintext
efficiency as Real (Brief = "Efficiency", Default=0.5, final Lower=0, final Upper=1);

# Pressure
pressure as Real (Brief = "Pressure", Default=1, Lower=1e-30, Upper=5e7, final Unit = 'atm');
press_delta as pressure (Brief = "Pressure Difference", Default=0.01, Lower=-5e6);
head_mass as Real (Brief = "Head", Default=50, Lower=1e6, Upper=1e8, final Unit = 'kJ/kg');
head as Real (Brief = "Head", Default=50, Lower=1e6, Upper=1e8, final Unit = 'kJ/kmol');

# Temperature
temperature as Real (Brief = "Temperature", Default=300, Lower=27, Upper=5273, final Unit = 'K');
temp_delta as temperature (Brief = "Temperature Difference", Default=30, Lower=-1000, Upper=1000);

# Time
time_h as positive (Brief = "Time in hours", Default=1, Upper=1e4, final Unit = 'h');
time_min as time_h (Brief = "Time in minutes", DisplayUnit = 'min');
time_sec as time_h (Brief = "Time in seconds", DisplayUnit = 's');
frequency as positive (Brief = "Frequency", Default=1, Upper=100, final Unit = '1/s');

# Size related
angle as Real (Brief = "Angle", Default=0, Lower=-7, Upper=7, final Unit = 'rad');
area as positive (Brief = "Area", Default=1, Upper=1E6, final Unit = m^2);length as positive (Brief = "Length", Default=1, Upper=5e8, final Unit = m);
length_delta as length (Brief = "Difference of Length", Lower=0-1000);volume as positive (Brief = "Volume", Default=10, Upper=1000, final Unit = m^3);
volume_mol as positive (Brief = "Molar Volume", Default=10, Upper=1E6, final Unit = m^3/mol);
volume_mass as positive (Brief = "Specific Volume", Default=10, Upper=1E30, final Unit = m^3/kg);
```
Including sub-models and types

Automatic model documentation

Basic sections to create a math. model

Input and output connections

Symbol of variable in LaTeX command for documentation

Port location to draw a flowsheet connection
EMSO Tutorial
– FlowSheet Components –

Degree of Freedom

Dynamic Degree of Freedom

Simulation options

Parameters of DEVICES

Explorer

simpleTank.mso

FlowSheet Simple_Tank

DEVICES

Feed as source (Brief = "Feed stream");
Tanque as tank_circular (Brief = "Tank");
Process as sink (Brief = "Sink stream");

CONNECTIONS

Feed.Outlet to Tanque.Inlet;
Tanque.Outlet to Process.Inlet;

SET
Tanque.k = 8 * m^2.5/h;
Tanque.Dh = 2 * m;

SPECIFY
"Feed flow"
Feed.Outlet.Pvol = 10 * m^3/h;

"Feed temperature de entrada"
Feed.Outlet.T = 300 * K;

"Feed pressure"
Feed.Outlet.P = 1 * atm;

INITIAL
"Initial level of tank"
Tanque.h = 1 * m;

OPTIONS
TimeStart = 0;
TimeStep = 0.1;
TimeEnd = 2;
TimeUnit = 'h';
end
Horizontal axis is always the independent variable (usually time)
Choose the file format

Right-click the mouse button and select “Export Image”
Choose the file format

RLT: MATLAB/SCILAB
XML: EXCEL/OpenOffice
Using EXCEL to analyze the results

Results separated by devices
Using MATLAB to analyze the results
EMSO Tutorial
– Building Block Diagrams: create file –

Selected components from physical properties package

Devices found in the model library
EMSO Tutorial
– Building Block Diagrams: select devices –

When making a connection, only compatible ports become available to connect.

drag & drop ports to create a connection

click to create a device
EMSO Tutorial
– Building Block Diagrams: set case study –

Variable status: unknown (Evaluate)
known (Specify)
initial condition (Initial)
estimate (Guess)
Available models

<table>
<thead>
<tr>
<th>EOS Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ideal</td>
<td>Ideal Gas, valid only for vapour models</td>
</tr>
<tr>
<td>IdealLiquid</td>
<td>Ideal Liquid, valid only for liquid models</td>
</tr>
<tr>
<td>RK</td>
<td>Redlich-Kwong</td>
</tr>
<tr>
<td>SRK</td>
<td>Soave-Redlich-Kwong</td>
</tr>
<tr>
<td>PR</td>
<td>Peng-Robinson</td>
</tr>
<tr>
<td>APR</td>
<td>Assymetric-PR</td>
</tr>
<tr>
<td>ASRK</td>
<td>Assymetric-RK</td>
</tr>
<tr>
<td>UNIFAC</td>
<td>UNIFAC (Dortmund)</td>
</tr>
</tbody>
</table>

In development: PC-SAFT
EMSO Tutorial
– Building Block Diagrams: simulating –

Sample Diagram: Check Consistency

Output Levels: Normal Output
Number of variables: 95
Number of equations: 86
Number of specifications: 9
Degrees of freedom: 0
Structural differential index: 1
Extra equations: 91
Extra Variables: 0
Dynamic degrees of freedom: 4
Number of initial Conditions: 4

Simulation of Sample_diagram finished successfully in 0.234 seconds.
Note: LaTeX must be installed.
4. Dynamic Degree of Freedom – Consistency Analysis –

Degree of Freedom (DoF)

= 0 (for simulation) > 0 (for optimization)

Dynamic Degree of Freedom (DDoF)

= number of given initial conditions

Check → Units of measurement
      → Structural non-singularity
      → Consistent initial conditions
Given a system of DAE: \( F(t, y, y') = 0 \)

The **Dynamic Degree of Freedom (DDoF)** is the number of variables in \( y(t_0) \) that can be assigned arbitrarily to compute a set of consistent initial conditions \( \{y(t_0), y'(t_0)\} \) of the DAE system. Is the true number of states of the system (or the system order of the DAE). Is the **number of initial conditions** that must be given.

For **low-index** DAE system (index 0 and 1) the DDoF is equal to the number of differential equations.

For **high-index** DAE system (index > 1) the DDoF is equal to the number of differential variables minus the number of hidden constraints.
Example: classical pendulum problem

\[ x' = w \]  \hspace{1cm} (1)  \hspace{1cm} x(0) = 1 \hspace{1cm} x'(0) = 0

\[ y' = z \]  \hspace{1cm} (2)  \hspace{1cm} y(0) = 0 \hspace{1cm} y'(0) = 0

\[ w' = T \cdot x \]  \hspace{1cm} (3)  \hspace{1cm} w(0) = 0 \hspace{1cm} w'(0) = 1

\[ z' = T \cdot y - g \]  \hspace{1cm} (4)  \hspace{1cm} z(0) = 0 \hspace{1cm} z'(0) = -g

\[ x^2 + y^2 = L^2 \]  \hspace{1cm} (5)

Inconsistent initial condition:

\[ F(t, y, y') = 0 \hspace{1cm} F(0, y(0), y'(0)) = 0 \]

Hidden constraints:

Differentiating (5) and using (1) and (2):
\[ x \cdot w + y \cdot z = 0 \]  \hspace{1cm} (6)  \hspace{1cm} \rightarrow \hspace{1cm} x(0) \cdot w(0) + y(0) \cdot z(0) = 0

Differentiating (6) and using (1)–(5):
\[ w^2 + z^2 + T \cdot L^2 = g \cdot y \]  \hspace{1cm} (7)  \hspace{1cm} \rightarrow \hspace{1cm} w(0)^2 + z(0)^2 + T(0) \cdot L^2 \neq g \cdot y(0)

Differentiating (7) and using (2), (3), (4), (6):
\[ T' = -3 \frac{g \cdot z}{L^2} \]  \hspace{1cm} (8)  \hspace{1cm} \rightarrow \hspace{1cm} T'(0) = -3 \frac{g \cdot z(0)}{L^2}
Dynamic Degree of Freedom
– High-Index DAE System –

Example: classical pendulum problem

\[
\begin{align*}
    x' &= w \\
    y' &= z \\
    w' &= T \cdot x \\
    z' &= T \cdot y - g \\
    x^2 + y^2 &= L^2 \\
    x \cdot w + y \cdot z &= 0 \\
    w^2 + z^2 + T \cdot L^2 &= g \cdot y \\
    T' &= -3 g \cdot z / L^2
\end{align*}
\]

10 variables \((y, y')\)

8 equations

2 DDoF

\text{Satisfies the inconsistent I.C.}

\text{But not any pair!}
Three general approaches:

1) Manually modify the model to obtain a lower index equivalent model

2) Integration by specifically designed high-index solvers (e.g., PSIDE, MEBDFI, DASSLC)

3) Apply automatic index reduction algorithms
using "types.mso";

FlowSheet pend
PARAMETERS
  g as acceleration (Brief="Gravity acceleration");
  L as length (Brief="Pendulum cable length");

VARIABLES
  x as length_delta (Brief="Position x");
  y as length_delta (Brief="Position y");
  w as velocity (Brief="Velocity for x");
  z as velocity (Brief="Velocity for y");
  T as Real (Brief="Tension on cable";
      Default=10, Unit="1/s^2");

EQUATIONS
  "Velocity on x"
  diff(x)=w;

  "Velocity on y"
  diff(y)=z;

  "Tension on x"
  diff(w)=T*x;

  "Tension on y"
  diff(z)=T*y-g;

  "Position Constraint"
  x^2+y^2=L^2;

SET
  g = 9.8 * 'm/s^2';
  L = 0.9 * 'm';

INITIAL
  "Initial Position x"
  x = 0.9 * 'm';

  "Initial x Velocity"
  w = 0 * 'm/s';

OPTIONS
  TimeStep = 0.1;
  TimeEnd = 36;
  Integration = "index0";

NLASolver(
    RelativeAccuracy = 1e-8,
    AbsoluteAccuracy = 1e-9
);

DAESolver(
    File = "dassl",
    #File = "mebdf",
    RelativeAccuracy = 1e-6,
    AbsoluteAccuracy = 1e-8
);

SparseAlgebra = true;
end
Dynamic Degree of Freedom
– High-Index DAE: consistency analysis –

EQUATIONS
31
"Velocity on x"
32
diff(x)=w;
33
"Velocity on y"
34
diff(y)=z;
35
"Tension on x"
36
diff(w)=T*x;
37
"Tension on y"
38
diff(z)=T*y-g;
39
"Position Constraint"
40
x^2+y^2=L^2;
41

Checking the consistency for "pend" in file D:\User\Argel\PROJETOSC\AIso\EMSO\msos\sample\sample_pend.mso:
Number of variables: 5
Number of equations: 5
Degrees of freedom: 0
Structural differential index: 3
Extra Equations: 9
Extra Variables: 6
Dynamic degrees of freedom: 2
Number of initial Conditions: 2
System is consistent.
Error propagation

index-0 solver vs index-3 solver

Drift-off effect

$L = 0.9 \text{ m}, \ g = 9.8 \text{ m/s}^2 \quad \therefore \ I.C.: \ x(0) = 0.9 \text{ m} \quad \text{and} \quad w(0) = 0$
5. Debugging Techniques

- Questions to be answered to assist the user of a CAPE tool - debugging:
  - For an under-constrained model which variables can be fixed or specified?
  - For an over-constrained model which equations should be removed?
  - For dynamic simulations, which variables can be supplied as initial conditions?
  - How to report the inconsistencies making it easy to fix?

- In other words, debugging methods need to go beyond degrees of freedom and the currently available index analysis methods.
Static models - Nonlinear Algebraic (NLA) systems:

- Several structural analysis methods available on the literature
- Most EO tools implement a degrees of freedom (DoF) and structural solvability analysis but user assistance is very limited when ill-posed models are found

Dynamic models - Differential Algebraic Equation (DAE) systems:

- Currently available methods are limited to index and dynamic degrees of freedom (DDoF) analysis
- The well-known EO commercial tools have a high-index check which can fail even for some simple low-index problems
Bipartite graphs can be used to solve combinatorial problems:

- Tasks to machines
- Classes to rooms
- Equations to variables

Bipartite graph $G(V = V_e \cup V_v, E)$ have two independent sets of vertices

- Vertices in the same partition must not be adjacent
- We can have alternating and augmenting paths

Matching $\{\{1,5\}, \{3,6\}, \{4,7\}\}$ w/ augmenting path
Graph for variable-equation relationship

\[
\begin{align*}
  f_1(x_1) &= 0 \\
  f_2(x_1, x_2) &= 0 \\
  f_3(x_1, x_2) &= 0 \\
  f_4(x_2, x_3, x_4) &= 0 \\
  f_5(x_4, x_5) &= 0 \\
  f_6(x_3, x_4, x_5) &= 0 \\
  f_7(x_5, x_6, x_7) &= 0
\end{align*}
\]

Maximum Matching
Multiple Solutions

variables values
or equations forms
are irrelevant
Debugging Nonlinear Problems

- Discover if there are over or under-constrained partitions
- Start from unconnected vertices and walk in alternating paths

Dulmage and Mendelsohn (DM) decomposition
A Simple Example

\[ x'_1 - x'_2 = a(t) \]
\[ x_2 = b(t) \]

Solution:

\[ x_1(t) = x_1(0) + \int_0^t a(\tau) d\tau + b(t) \]
\[ x_2(t) = b(t) \]

- Only two differential variables
- Index-1 system
- Requires only one initial condition
- Initial condition must be \( x_1 \)
- \( x_1 \) is the only state of the model
Classic Algorithm

\[ x_1' - x_2' = a(t) \]
\[ x_2 = b(t) \]

- Who are the states?
- Which variables should be specified as initial conditions?
If only one initial condition is given (which is correct):

\[ x_1' - x_2' = a(t) \]

\[ x_2 = b(t) \]
If two initial condition are given (which is wrong):

\[ x_1' - x_2' = a(t) \]
\[ x_2 = b(t) \]

---

Performing initialization calculation at time: 0

Variables

- Known: 0
- Unknown: 2
  - Differential: 2
  - Algebraic: 0
- Model equations: 2
- Initial conditions: 2

Checking index of differential-algebraic equations (DAEs)...

ERROR: Your problem is a DAE system of index greater than 1. Your differential variables ("states") are not independent
\[ x'_1 - x'_2 = a(t) \]
\[ x_2 = b(t) \]

Determining specification state...
specification state determined.
Preparing simulation for solution
Starting new snapshot file.
Simulation ready for solution
Simulation has 4 variables, 2 equations and 3 non-zeros
Number of equations = 2, number of states = 2
DAE Systems \((G = (V_e, V_v, E), M)\)

1: \(M \leftarrow \emptyset\)
2: for \(v_e \in V_e\) do
3: if not AugmentMatching2\((G = (V_e, V_v, E), M, v_e, \text{false})\) then
4:  mark all colored \(v_k \in V_e\)
5:  uncolour \(V_e\)
6: if not AugmentMatching2\((G = (V_e, V_v, E), M, v_e, \text{true})\) then
7:  return false
8: end if
9: diff all marked \(v_k \in V_e\)
10: else
11: uncolour \(V_e\)
12: end if
13: end for
14: return true
AugmentMatching2\((G = (V_e \cup V_v, E), M, v_e, \text{alg})\)

1: colour \(v_e\)
2: if \(\{v_e, v_v\} \in E\) and \(\{v_e, v_v\} \notin M\) and \(v_v\) is eligible then
3: \(M \leftarrow M \cup \{v_e, v_v\}\)
4: return true
5: end if
6: for all \(\{v_e, v_v\} \in E\) do
7: if \(\{v_{e2}, v_v\} \in M\) and \(v_{e2}\) not colored and \(v_v\) is eligible then
8: if AugmentMatching2\((G = (V_e \cup V_v, E), M, v_{e2}, \text{alg})\) then
9: \(M \leftarrow M \cup \{v_e, v_v\}\)
10: return true
11: end if
12: end if
13: end for
14: return false
Debugging Techniques
– Applying the New Algorithm –

\[ x'_1 - x'_2 = a(t) \]
\[ x_2 = b(t) \]

- All equations and all \( x' \) are connected when it finishes
- Free variable nodes are the real states
- DM decomposition can be applied to the final matching
- Singularities are detected (classic algorithm runs indefinitely)
If only one initial condition is given (which is correct):

\[ x_1' - x_2' = a(t) \]
\[ x_2 = b(t) \]

<table>
<thead>
<tr>
<th>Number of variables: 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of equations: 2</td>
</tr>
<tr>
<td>Number of specifications: 0</td>
</tr>
<tr>
<td>Degrees of freedom: 0</td>
</tr>
<tr>
<td>Structural differential index: 1</td>
</tr>
<tr>
<td>Extra Equations: 1</td>
</tr>
<tr>
<td>Extra Variables: 0</td>
</tr>
<tr>
<td>Dynamic degrees of freedom (states): 1</td>
</tr>
<tr>
<td>Number of initial Conditions: 1</td>
</tr>
</tbody>
</table>
Debugging Techniques
– Applying the New Algorithm: high-index –

\[ x' = w \]  \( (1) \)
\[ y' = z \]  \( (2) \)
\[ w' = T \cdot x \]  \( (3) \)
\[ z' = T \cdot y - g \]  \( (4) \)
\[ x^2 + y^2 = L^2 \]  \( (5) \)

only two states!
Dynamic model of a distillation column for the separation of isobutane from a mixture of 13 compounds

<table>
<thead>
<tr>
<th>N. Trays</th>
<th>N. Variables</th>
<th>Time* (s)</th>
<th>Time / N² (s · 10⁹)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>2157</td>
<td>0.04</td>
<td>9.46</td>
</tr>
<tr>
<td>40</td>
<td>3877</td>
<td>0.14</td>
<td>9.58</td>
</tr>
<tr>
<td>80</td>
<td>7317</td>
<td>0.52</td>
<td>9.79</td>
</tr>
</tbody>
</table>

* Pentium M 1.7 GHz PC with 2 MB of cache memory, Ubuntu Linux 6.06
What is coming next?

- **Tools**
  - Model updating tool and development of virtual analyzer based on Constrained Extended Kalman Filter (CEKF)
  - Model generation tool for predictive controllers

- **Features**
  - Creation of discretization functions for integral-partial differential equations
  - Implementation of MINLP solver interfaces

- **Technologies**
  - Hessian evaluation by reverse-mode automatic differentiation
  - New resources for incremental building of flowsheets in the G.U.I.
Robust strategies for on-line updating of dynamic models

Dynamic data reconciliation and gross error detection

Parameters selection and estimation

Related topics:

- Hybrid and rigorous modeling
- Order reduction of nonlinear models
- Fault diagnosis
- NMPC tuning strategies
Challenges

DAE solvers

- Reliable high-index (>3) solvers
- Automatic/guided selection of feasible set of variables for initial condition
- Index reduction with trajectory projection onto hidden manifold
Challenges

- Multi-level dynamic simulator
- Simultaneous data reconciliation and parameter estimation tool
- Specialist system
- Integrated tool for D-RTO
- Dynamic optimizer with adaptive grid
- Self-tuned nonlinear model predictive controller
Challenges

- Systems Interoperability
  - Truly CAPE-OPEN
    - Unified communication protocol
  - Heterogeneity and multi-platform
  - Multi-processing and shared-memory advantages
Challenges

Complex systems

- Advances in process simulation + CFD
- Multi-scale modeling + simulation tools
- Bifurcation + control system design
- Hybrid modeling


References

**DAE Solvers:**

**DASSL:** Petzold, L.R. (1989) http://www.enq.ufrgs.br/enqlib/numeric/numeric.html


**MEBDFI:** Abdulla, T.J. and J.R. Cash (1999), http://www.netlib.org/ode/mebdfi.f


Research Group

GIMSCOP - 2008

Argimiro Resende Secchi, D.Sc.
Evaristo Chalbaud Biscaia Jr, D.Sc.
Jorge Otávio Trierweiler, D.Sc.
Nilo Sérgio Medeiros Cardozo, D.Sc.
Marcelo Farenzena, D.Sc.
Rafael de Pelegrini Soares, D.Sc.
Adriano Giraldi Fisch, M.Sc.
Débora Jung Luizieto, M.Sc.
Edson Cordeiro do Valle, M.Sc.
Eduardo Moreira de Lemos, M.Sc.
Euclides Almeida Neto, M.Sc.
Gabriela Sporlede Straatmann, M.Sc.
Géron Balbueno Bicca, M.Sc.
Gustavo Alberto Neumann, M.Sc.
Luciane da Silveira Ferreira, M.Sc.
Marcelo Escobar, M.Sc.
Nina Paula Gonçalves Salau, M.Sc.
Paula Betio Staudt, M.Sc.
Ricardo Guilherme Duraiski, M.Sc.
Tiago Fiorenzano Finkler, M.Sc.
Anderson de Campos Paim, Eng.
Andrea Cabral Farias, Eng.
Antonio José V. Nascimento, Eng.
Bruna Racoski, Eng.
Cristine Alessandra Kayser, Eng.
Fabio Cesar Diehl, Eng.
Gustavo Rodrigues Sandri, Eng.
Jovani Luiz Fávero, Eng.
Luciano Forgiarini, Eng.
Marcos Lovato Alencastro, Eng.
Rafael Busato Sartor, Eng.
Rodolfo Rodrigues, Eng.
Thais Machado Farias, Eng.
Bruno Cardozo Mohler, I.C.
Caio Felippe Curitiba Marcellos, I.C.
Ivana Martins, I.C.
Josias José Junges, I.C.
Luiza Gueller Zardin, I.C.
Maria Aparecida Paula Lima, I.C.
Sara Scomazzon Masiero, I.C.
Igor Rodacovski, Tec. Inf.
Irma Maria Bueno, Sec.
... thank you for your attention!

http://www.enq.ufrgs.br/alsoc

Process Modeling, Simulation and Control Lab
- Prof. Dr. Argimiro Resende Secchi
- Phone: +55-21-2562-8349
- E-mail: arge@peq.coppe.ufrj.br
- http://www.peq.coppe.ufrj.br/Areas/Modelagem_e_simulacao.html

Process Simulation and Optimization Lab
- Prof. Dr. Rafael de Pelegrini Soares
- Phone: +55-51-3308-4166
- E-mail: rafael@enq.ufrgs.br
- http://www.enq.ufrgs.br/labs/lasim.html
Extra slides
Flash multi-component

\[ F, z, P_f, T_f \]

\[ T, P \]

\[ V, y \]

\[ L, x \]
A liquid-phase mixture of C hydrocarbons, at given temperature and pressure, is heated and continuously fed into a vessel drum at lower pressure, occurring partial vaporization. The liquid and vapor phases are continuously removed from the vessel through level and pressure control valves, respectively. Determine the time evolution of liquid and vapor stream composition and the vessel temperature and pressure, due to variations in the feed stream, keeping the heating rate constant.
• negligible vapor holdup (no dynamics in vapor phase);
• thermodynamic equilibrium (ideal stage);
• no droplet drag in vapor stream;
• negligible heat loss to surroundings;
• $\Delta$(internal energy) $\approx \Delta$(liquid-phase enthalpy);
• perfect mixture in both phases.
Overall mass balance (molar base):

\[
\frac{dm}{dt} = F - V - L \quad (1)
\]

Component mass balance:

\[
\frac{d}{dt}(m x_i) = F z_i - V y_i - L x_i \quad (2) \quad i = 1, 2, \ldots, C
\]

Equilibrium:

\[
y_i = K_i x_i \quad (3) \quad i = 1, 2, \ldots, C
\]

\[K_i = f(T, P, x, y)\]

Molar fraction:

\[
\sum_{i=1}^{C} x_i = 1 \quad (5)
\]
Exercícios de Modelagem

Energy balance:

\[ \frac{d}{dt} (m \ h) = F \ h_f + q - V \ H - L \ h \quad (6) \]

Enthalpies:

\[ h = f(T, P, x) \quad (7) \]
\[ H = f(T, P, y) \quad (8) \]
\[ h_f = f(T_f, P_f, z) \quad (9) \]

Controllers:

\[ L = f(m) \quad (10) \]
\[ V = f(P) \quad (11) \]
### Variable Units of Measurement

<table>
<thead>
<tr>
<th>Variable</th>
<th>Units of Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m )</td>
<td>kmol</td>
</tr>
<tr>
<td>( F, L, V )</td>
<td>kmol s(^{-1} )</td>
</tr>
<tr>
<td>( t )</td>
<td>s</td>
</tr>
<tr>
<td>( x_i, y_i, z_i )</td>
<td>kmol kmol(^{-1} )</td>
</tr>
<tr>
<td>( K_i )</td>
<td>–</td>
</tr>
<tr>
<td>( T, T_f )</td>
<td>K</td>
</tr>
<tr>
<td>( P, P_f )</td>
<td>kPa</td>
</tr>
<tr>
<td>( q )</td>
<td>kJ s(^{-1} )</td>
</tr>
<tr>
<td>( h, H, h_f )</td>
<td>kJ kmol(^{-1} )</td>
</tr>
</tbody>
</table>
variables: $m, F, L, V, t, x_i, y_i, z_i, K_i, T, T_f, P, P_f, q, h, H, h_f \Rightarrow 13+4C$

constants: \( \Rightarrow 0 \)

specifications: $q, t \Rightarrow 2$

driving forces: $F, z_i, T_f, P_f \Rightarrow 3+C$

unknown variables: $m, L, V, x_i, y_i, K_i, T, P, h, H, h_f \Rightarrow 8+3C$

equations: $8+3C$

**Degree of Freedom** = variables – constants – specifications – driving forces – equations = unknown variables – equations = $(13+4C) – 0 – 2 – (3+C) – (8+3C) = 0$

Initial condition: $m(0), x_i(0), T(0) \Rightarrow 2+C$

**Dynamic Degree of Freedom** (index < 2) = differential equations – initial conditions = $(2+C) – (2+C) = 0$
Running EMSO

Note: file Sample_flash_pid.mso has level and pressure controllers.
EQUATIONS

"Overall Molar Balance"
diff(ML) = Inlet.F - OutletL.F - OutletV.F;

"Component Molar Balance"
diff(N) = Inlet.F*Inlet.z - OutletL.F*OutletL.z - OutletV.F*OutletV.z;

"Molar Holdup"
M = ML*OutletL.z;

"Equilibrium"
OutletV.z = K*OutletL.z;

"Equilibrium Constant"
PP.LiquidFugacityCoefficient(OutletL.T,OutletL.P,OutletL.z) =
PP.VapourFugacityCoefficient(OutletV.T,OutletV.P,OutletV.z) * K;

"Mol fraction normalization"
sum(Outlet.z) = sum(OutletV.z);

"Energy Balance"
diff(E) = Inlet.F*Inlet.h - OutletL.F*OutletL.h - OutletV.F*OutletV.h + InletQ.Q;

"Energy Holdup"
E = ML*OutletL.h;

"Thermal Equilibrium"
OutletV.T = OutletL.T;

"Mechanical Equilibrium"
OutletV.P = OutletL.P;

"Vaporization Fraction"
OutletV.F = Inlet.F * vfrac;

"Liquid volume"
VL = PP.LiquidVolume(OutletL.T, OutletL.P, OutletL.z);

switch orientation
case "vertical";
"Cross Section Area"
|  Across = 0.5 * asin(1) * diameter^2;

"Liquid Level"
|  ML * VL = Across * Level;
FlowSheet test_flash

PARAMETERS
PP as Plugin(Brief="Physical Properties", Type="PP",
Components = ['1,3-butadiene', 'isobutene', 'n-pentane',
'1-pentene', '1-hexene', 'benzene'],
LiquidModel = "PR",
VapourModel = "PR"
);
NComp as Integer;

VARIABLES
Q as energy_source (Brief="Heat supplied");

SET
NComp = PP.NumberOfComponents;

DEVICES
fl as ex_flash;
s1 as source;

CONNECTIONS
s1.Inlet to fl.Inlet;
Q.InletQ to fl.InletQ;

EQUATIONS
fl.Inlet.F = 400*sqrt(fl.Level/\(\text{m}\)) * 'kmol/h';

SPECIFY
fl.Inlet.F = 496.3 * 'kmol/h';
s1.Inlet.T = 338 * 'K';
s1.Inlet.P = 507.1 * 'kPa';
s1.Inlet.z = [0.2379, 0.30820, 0.09958];
fl.Inlet.V = 68.5 * 'kmol/h';
Q.InletQ = 0 * 'kJ/h';

SET
fl.V = 50 * 'm\(^3\)';
fl.diameter = 2 * 'm';
fl.orientation = "horizontal";

INITIAL
fl.Inlet.T = 338 * 'K';
fl.Level = 0.4 * 'm';
fl.InletL.z[1] = 0.1;
fl.InletL.z[2] = 0.1;
fl.InletL.z[3] = 0.1;
fl.InletL.z[4] = 0.1;
fl.InletL.z[5] = 0.1;
fl.InletL.z[6] = 0.5;
Standard Interfaces

CAPE-OPEN

Diagram showing relationships between Simulator, Executive, Libraries, Unit, Numerics, Thermo, External UO, External Thermo, and External Numeric interfaces. Legend:
- U: Unit Interfaces
- T: Thermo Interfaces
- N: Numeric Interfaces
Example of CAPE-OPEN: DyOS (Dynamic Optimization Software) - Marquardt’s group (2000)
Another example of CAPE-OPEN: EMSO (Environment for Modeling, Simulation and Optimization) - Soares and Secchi (2004)

EMSO A

EMSO B

methanol plant

CORBA Object Bus

EMSO
Other available tools and features
Optimization

```plaintext
MAXIMIZE
leves;

FREE
fl.InletL.T;
#fl.InletL.P;

EQUATIONS
fl.InletL.T < 320 * 'K';
fl.InletL.T > 300 * 'K';

OPTIONS
Dynamic = false;
NLPSolveNL = false;
NLPSolver(#File = "optpp_emso",
          #File = "ioppt_emso",
          RelativeAccuracy = 1e-6);

end
```

```plaintext
VARIABLES
x1 as Real(Default=2, Lower=1, Upper=5);
x2 as Real(Default=5, Lower=1, Upper=5);
x3 as Real(Default=1, Lower=1, Upper=5);
end
```

```plaintext
MINIMIZE
x1*x4*(x1+x2+x3) + x3;

EQUATIONS
x1*x2*x3*x4 > 25;
x1*x1 + x2*x2 + x3*x3 + x4*x4 = 40;

OPTIONS
NLPSolver(#File = "ioppt_emso"
          #File = "optpp_emso"
          Dynamic = false;
```

```plaintext
end
```
Parameter Estimation

```
Estimation Biop_NE_Estt5 as Biop_NE_process5t

ESTIMATE

# PARAMETER START LOWER UPPER UNIT
Kss   0.2009  0.004   7      'kg/m3';
Ksn   0.0446  0.005   1      'kg/m3';
min   0.7979  0.1     0.8    '1/s';
alfa  2.0293  1       5      ;
gama  0.08502 0.05    5      '1/s';
K1    0.4059  0.1     3      'kg/kg';
K2    -0.00795 -1      3      '1/s';
Yn    10.62   0.1     18     
kD    0.007   0.0005  1      '1/s';

EXPERIMENTS

# DATA FILE WEIGHT
"Bio.dat" 1;

OPTIONS

Statistics(
  Fits=true,
  Parameters=false,
  Predictions=false
);

NLPSolver(
  MaxIterations = 1000,
  File = "complex"
    #File = "iopt_emso"
);

Dynamic = true;
```

```
Estimation PV_Est as PV_Flow

ESTIMATE

# PAR START LOWER UPPER UNIT
A    1.5    -1   10;
B    1000   800   3000 'K';
C    50     20    200 'K';

EXPERIMENTS

# FILE WEIGHT
"pv_est.dat" 1;

OPTIONS

NumJac = false;

NLPSolver(
  File = "iopt_emso"
    #File = "complex"
);

Dynamic = false;
```
**Data Reconciliation**

```
FlowSheet HeatEx_Flow

VARIABLES
x1 as Real (Default=50.00, Lower=0.00, Upper=150);
x2 as Real (Default=50.00, Lower=0.00, Upper=150);
x3 as Real (Default=50.00, Lower=0.00, Upper=150);
x4 as Real (Default=50.00, Lower=0.00, Upper=150);
x5 as Real (Default=50.00, Lower=0.00, Upper=150);
x6 as Real (Default=50.00, Lower=0.00, Upper=150);

SPECIFY
x1 = 100.91;
x2 = 64.45;
x3 = 34.65;

EQUATIONS
x1 - x2 - x3 = 0;
x2 - x4 = 0;
x3 - x5 = 0;
x4 + x5 - x6 = 0;

OPTIONS
Dynamic = false;
```

```
Reconciliation HeatEx_Rec as HeatEx_Flow

RECONCILE
x1; x2; x3; x4; x5; x6;
#x1; x2; x5; x6;
#x1; x2;
#x1; x6;

FREE
x1; x2;

EXPERIMENTS
FILE
# file "heatEx.dat" 1;
#"heatEx_1.dat" 1;
#"heatEx_2.dat" 1;
#"heatEx_3.dat" 1;
#"heatEx_GE.dat" 1;

OPTIONS
Filter = "mean";
Significance = 0.95;

GrossErrorTests(
    Global = true,
    Nodal = true,
    Measurements = true
);

NLPsolver(
    MaxIterations=1000,
    File = "complex"
    #File = "iopt_emso"
);
Dynamic = false;
```
Interface EMSO-OPC
Interface EMSO-OPC

Simulator

Units Converter

Conversion Rule:

\[ [\text{Tag Units}] = [\text{Variable Units}] \times A + B \]

Variable Units: \( \text{kmol/h} \)
Tag Units: \( \text{kmol/h} \)
Enter A: \( 1 \)
Enter B: \( 0 \)

Cancel  OK

Select Variable

Plant
Interface EMSO-AUTO

EMSO → System of equations
                Jacobian matrix
                       First steady-state solution → AUTO

using "types";
FlowSheet ab_dae

PARAMETERS
p1 as Real;
p2 as Real;
p3 as Real;

VARIABLES
u1 as Real;
u2 as Real;
u3 as Real;

SET
p1 = 0;
p2 = 14;
p3 = 2;

EQUATIONS
diff(u1) = -u1 + p1 * (1 - u1) * u3;
diff(u2) = -u2 + p1 * p2 * (1 - u1) * u3 - p3 * u2;
u3 = exp(u2);
Interface EMSO-MATLAB
Interface EMSO-MATLAB

```plaintext
using "types";
FlowSheet bio
PARAMETERS
mMax as frequency (Brief="Monod parameter", DisplayUnit="1/h");
Km as conc_mass (Brief="Monod parameter", Default=0.12);
K1 as inv_conc_mass (Brief="Monod parameter", Default=0.4545);
Y as coefficient (Brief="Yield biomass/substrate", Default=0.4);
x2F as conc_mass (Brief="Substrate feed concentration", Default=1);
D as frequency (Brief="Dilution rate F/V", DisplayUnit="1/h");
VARIABLES
biomass as conc_mass (Brief="Biomass concentration");
substrate as conc_mass (Brief="Substrate concentration");
m as frequency (Brief="Specific growth rate", Lower=-1e-3, DisplayUnit="1/h");

EQUATIONS
"Specific growth rate"
m = mMax*substrate/(Km+substrate+(K1*substrate^2));
"Biomass production"
diff(biomass)/biomass = m;
"Substrate consumption"
diff(substrate) = D*(2F*substrate)-(biomass*m/ν);
SET
mMax = 0.51 * '1/h';
x2F = 4 * 'kg/(m³)';
D = 0.1 * '1/h';

INITIAL
biomass = 1 * 'kg/m³';
substrate = 0.5 * 'kg/(m³)';
OPTIONS
Dynamic = true;
TimeStep = 0.01;
TimeEnd = 11;
TimeUnit = 'h';
SparseAlgebra = true;
end
```

For Help, pre Ln 1, Col. 1, CW

UNI
Interface EMSO-CFD

Energy Balance
Mass Balance

Overall heat transfer coefficient evaluation

Momentum Balance

\[ \rho, \mu, C_p, k \rightarrow U \]

\[ \rho, \mu \rightarrow V = f(z, r) \]