DYNAMIC MODELLING AND SIMULATION

INTRODUCTION

INDUSTRIAL EXPERIENCE WITH DYNAMIC SIMULATION

Paul I. Barton
Department of Chemical Engineering
Massachusetts Institute of Technology
Cambridge, MA

© copyright Paul I. Barton, July 1997

“Process modeling is the single technology that has had the biggest impact on our business in the last decade”

Frank Popoff
former CEO, Dow Chemical
April 1996

APPLICATIONS OF DYNAMIC SIMULATION

Dynamic simulation is useful throughout the entire lifetime of a plant: from conception to decommissioning.

Plant design:

a) a priori assessment of intrinsic operability and controllability of a plant - especially for highly integrated plants.

b) design and testing of regulatory control systems - selection of control structures, control algorithms, and initial tuning of loops.

c) design and testing of operating procedures - start-up, shut-down, feed stock changeover, etc.

d) hazard/safety studies - answer important questions raised by HAZOP study.

e) design and testing of protective and relief devices - study plant behaviour and performance of protective system during major deviations from steady-state (need good models!)

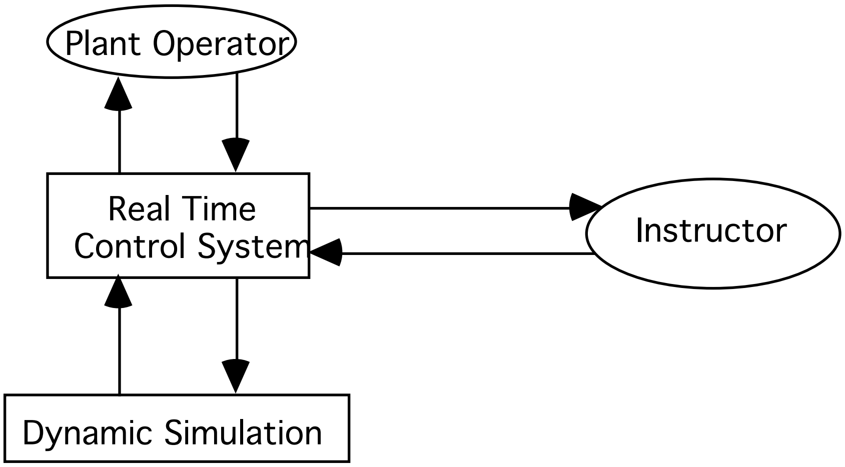
f) environmental studies - predict emissions generated during plant upsets/failures.

g) analysis of intrinsically dynamic processes: batch/semi-continuous processes, periodic processes (e.g., pressure swing adsorption)

APPLICATIONS OF DYNAMIC SIMULATION

Plant operation: typically a dynamic simulation is linked to the plant's real time control and monitoring software:

(a) computer based operator training:

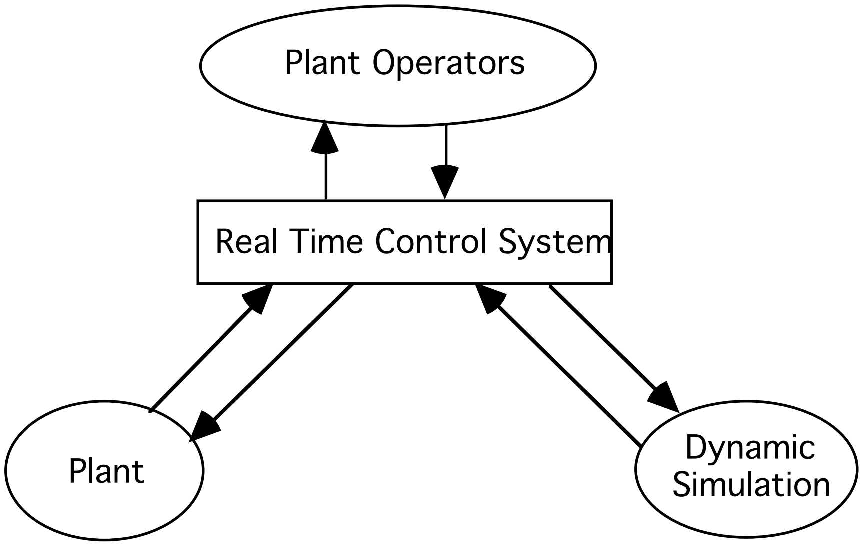


- dynamic simulation runs in real time and mimics behaviour of real plant.
- operator interfaces to control system (not simulator) — more realistic.
- instructor monitors and creates scenarios (e.g., disturbances, failures)

(b) validation of operating and safety procedures — test control software by dynamic simulation before implementing on the real plant!

APPLICATIONS OF DYNAMIC SIMULATION

(c) on-line operator decision support tool (simulation runs in parallel with real plant):



- update current state (initial condition) directly from measurements.
- dynamic predictor — run simulation faster than real time to predict hazards or problems (so that early corrective or preventative action can be taken).
- experimental tool — predict consequences of proposed actions.

CONTROLLABILITY STUDY OF BAYER DISTILLATION PROCESS

(Zitney et al., 1995)

BACKGROUND

- system of eight coupled and heat integrated distillation columns (nearly 1200 trays)
- separates eight crude silanes from a mixture of 40 components
- operation characterized by large holdup and time varying feed composition with no virtually no buffering between columns and upstream process

OBJECTIVES

Study performance of control system in order to:

- increase plant capacity
- and at same time improve quality and reduce energy consumption

APPROACH

- develop dynamic model of the overall system
- use dynamic model for identification and controllability experiments - avoid disrupting production

THE DYNAMIC MODEL

- detailed SPEEDUP (Aspen Technology, 1993) model developed of eight coupled columns

- one year effort for 2-3 engineers: process modeled in stages, so results available throughout this period

- tray/reboiler/condenser mass and energy balances, phase equilibrium modeled by Wilson equation

- differential-algebraic system of 75,000 coupled equations - largest known industrial SPEEDUP application at the time

- simulations performed on CRAY C90 supercomputer - solution speed increased upto 50-fold by exploiting vector parallelization of supercomputer

- simulation runs approx. 10 times faster than real time

APPLICATIONS OF THE MODEL

- steady-state simulation:
    - steady-state is just one initial condition for a dynamic model: always get steady-state model for free in dynamic study!
    - optimize plant for different feed concentrations and loads
    - identify which equipment must be revamped to increase capacity

- plant must be regulated at new optimal operating points - use dynamic model for identification and controllability studies:
    - sensitivity studies to identify necessary process measurements
    - sensitivity studies to define appropriate control structure (matching of sensors to control elements) and tuning of control parameters
    - validation of control system by predicting plant response to a variety of load and setpoint changes

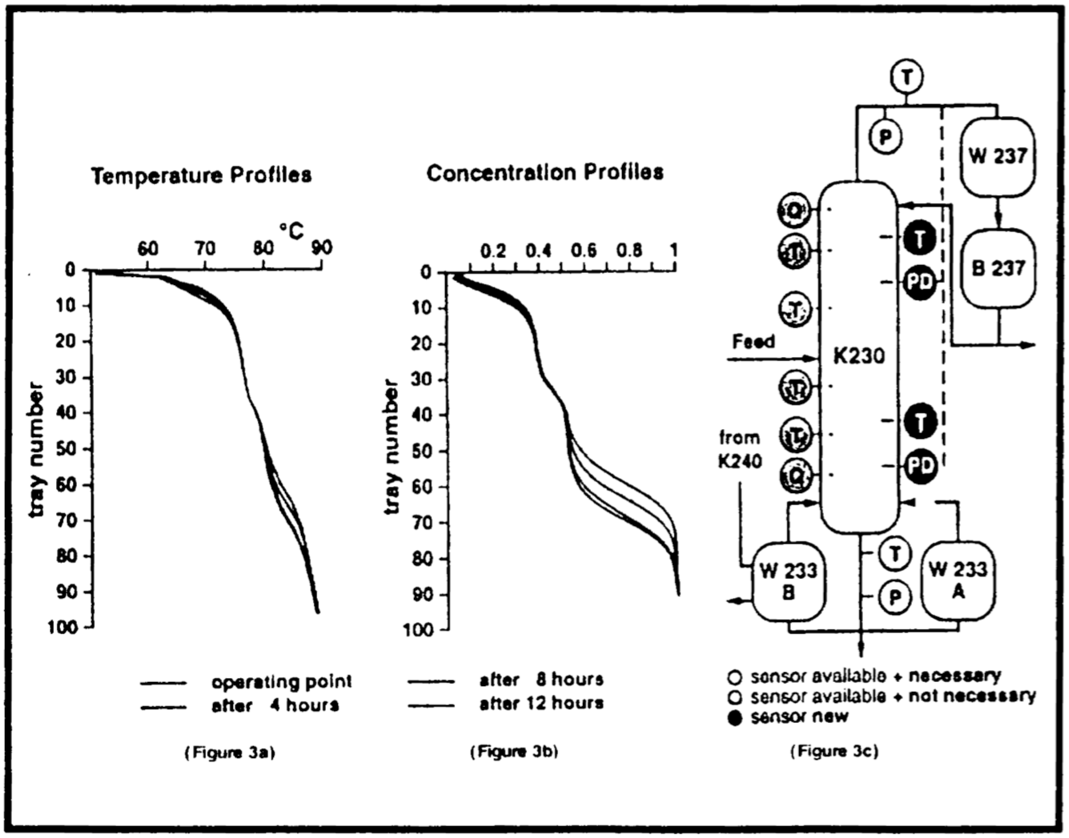
MEASUREMENT SELECTION

- study sensitivity of process outputs to range of common disturbances (see figure)
    - with dynamic model can observe *all* the plant states

- many existing sensors in inappropriate positions

- why use dynamic simulation? - necessary information in steady-state gain matrix
    - much easier to get steady-state gain matrix from SPEEDUP (compared to ASPEN PLUS)
    - think before applying dynamic simulation: effort *must* be justified

SENSITIVITY ANALYSIS FOR HEAT INTEGRATED COLUMN



(Figure 3a)

(Figure 3b)

(Figure 3c)

Figure 3. Sensitivity analysis and measurement selection.

from Zitney *et al.*, "Plantwide Dynamic Simulation on Supercomputers: Modeling a Bayer Distillation Process", *AIChE Symposium Series*, **91**(304), pp. 313-316, 1995.

Reproduced with permission. Copyright © American Institute of Chemical Engineers and Computers Aids for Chemical Engineering Education. All rights reserved.

CONTROLLER DESIGN AND VALIDATION

Given new measurement, completely new control system was designed.

Use dynamic simulations (rather than experiments with real plant) for:

- identification experiments to determine transfer function models for controller design

- controller design: major goal was disturbance rejection:
  - match outputs and controls to minimize interaction
  - tune PID constants using dynamic model

- controller validation: test controllers in wide variety of scenarios before implementation in plant

CONTROLLABILITY STUDY OF BAYER DISTILLATION PROCESS

BENEFITS

- doubling capacity only required revamping half the plant

- one third of the sensors were no longer necessary, and outputs more sensitive to disturbances identified

- decision made to design completely new control system

- controller testing by simulation saved much troubleshooting in plant, and faster and more successful startup of new control system

- certain time constants so large that operator never sees consequences of manual operations during their shift (see figure):
    - operators instructed to leave process in automatic operation as much as possible

- the act/discipline of developing a dynamic model can lead to significant new knowledge and insights concerning the process

EFFECTS OF A DISTURBANCE MAY BE FELT OVER SEVERAL SHIFTS
(VERY LARGE TIME CONSTANTS)

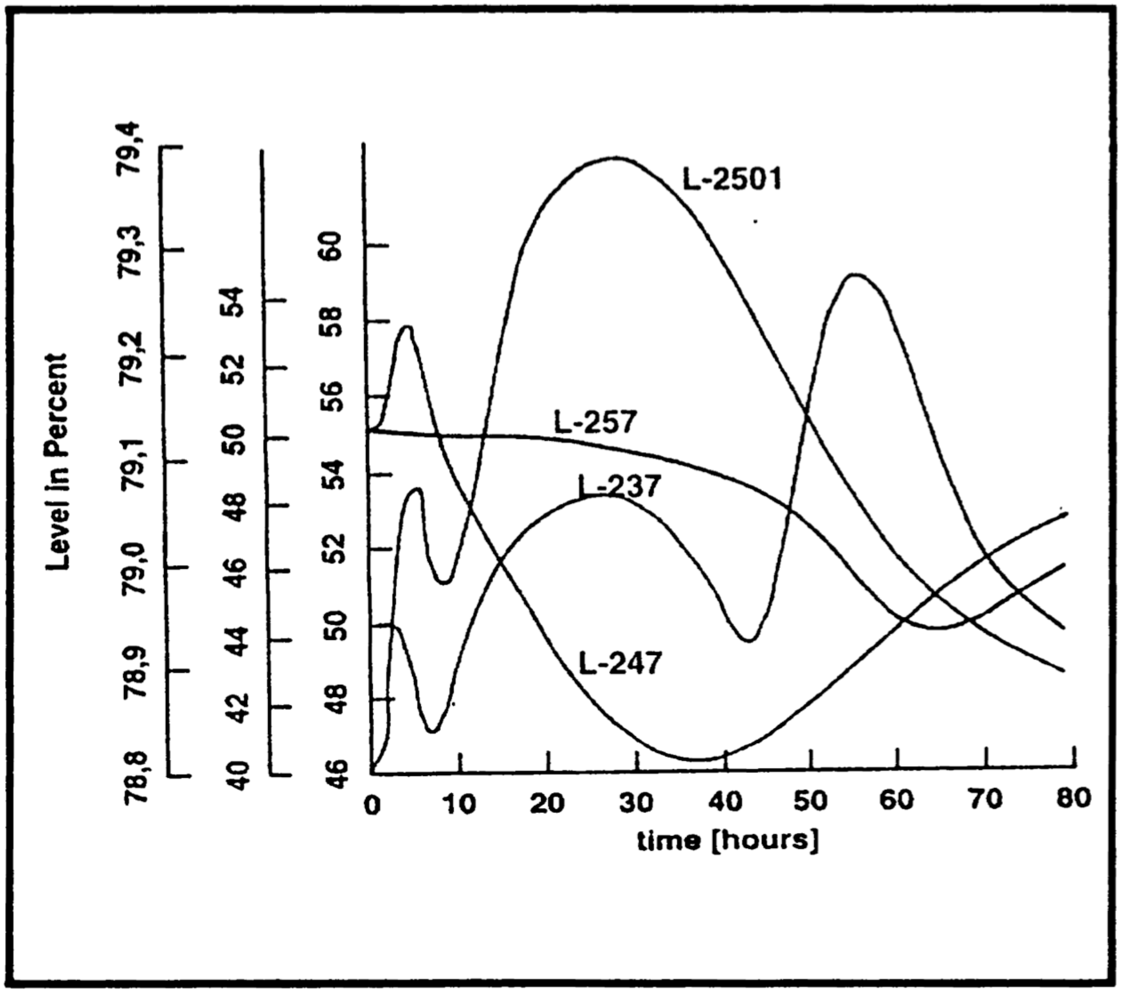


Figure 4. Dynamic plant behavior for a change in feed concentration.

from Zitney *et al.*, “Plantwide Dynamic Simulation on Supercomputers: Modeling a Bayer Distillation Process”, *AIChE Symposium Series*, **91**(304), pp. 313-316, 1995.

Reproduced with permission. Copyright © American Institute of Chemical Engineers and Computers Aids for Chemical Engineering Education. All rights reserved.

RETROFIT OF NORSK HYDRO TOPSIDE OFFSHORE OIL PRODUCTION PROCESS

(Naess et al., 1993)[1]

BACKGROUND

- introduction of flow from Tarbert pipeline may expose the Osberg topside process to slug regimes

- necessary to study process response to slugs without disrupting production
    - more sophisticated approach than "let's just try it and see what happens"!

- is insertion of slugcatcher in front of separation train necessary?

APPROACH

- use simulations with dynamic model to avoid disrupting production

- predict performance of slugcatcher designs

[1] this paper is instructive because it applies dynamic simulation to four different processes at four different stages of their lifetimes. It also compares two dynamic simulators.

SEPARATION AND STABILIZATION TRAIN AT OSBERG FIELD (NORTH SEA)

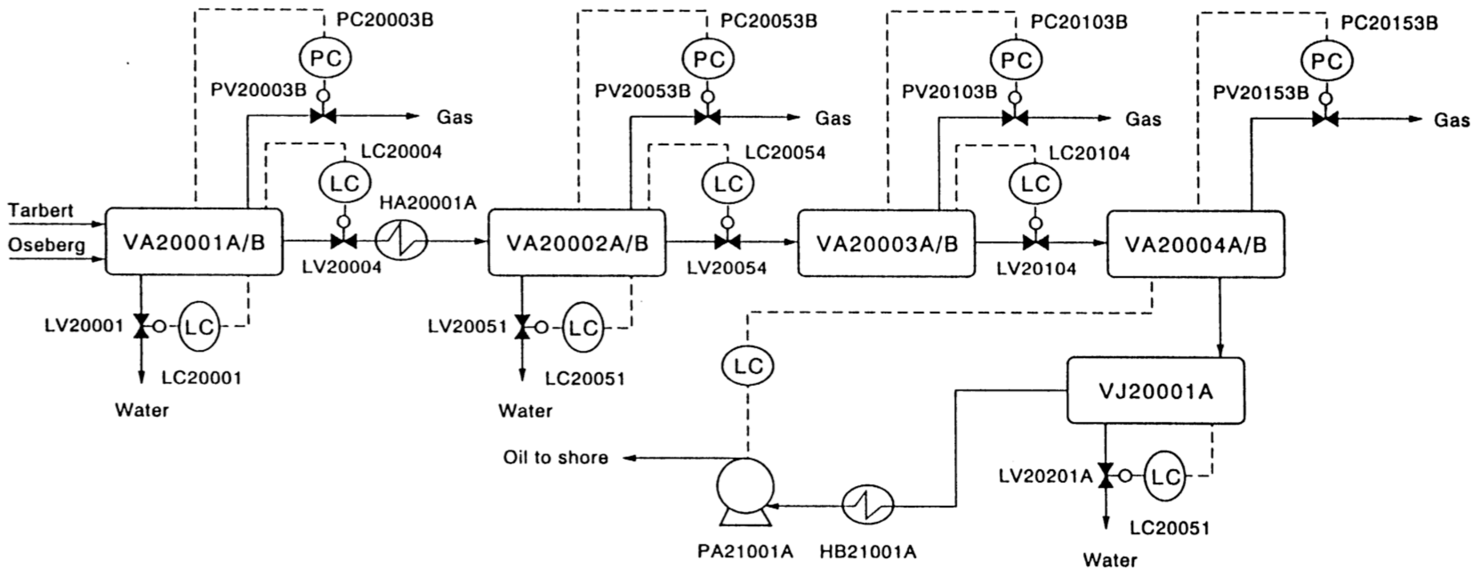


Fig. 3. Separation and stabilization train at the Oseberg field in the North Sea.

from Naess et al., "Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants", Computers Chem. Engng., 17, pp. 585-600, 1993.
(permission requested).

THE DYNAMIC MODEL

- dynamic model absolutely necessary as topside process cannot cope during the *transients*
  - the steady-states may be fine, but the path between them may lead to a plant trip

- SPEEDUP (Aspen Technology, 1993) used

- rigorous equilibrium models of two- and three-phase separators with seven components

- models of PI-controllers with real life tuning parameters

- models identified for control valves including time lag

- dynamics of piping neglected

- interfaced to in-house physical properties package via *procedures*
  - same physical properties as steady-state simulation
  - model validated against steady-state model used by engineering division

CASE STUDY

- use model to predict system response to seven “typical” slugs:
    - production start
    - maximum gas production
    - maximum water production

- worries about pressure trips unwarranted

- gas valve on second stage separator has too little capacity due to back pressure from the gas train

- in worst case (maximum water production) fourth stage separator will trip due to high-high level after 30 minutes

TYPICAL SLUG THROUGH TARBERT PIPELINE TO OSBERG

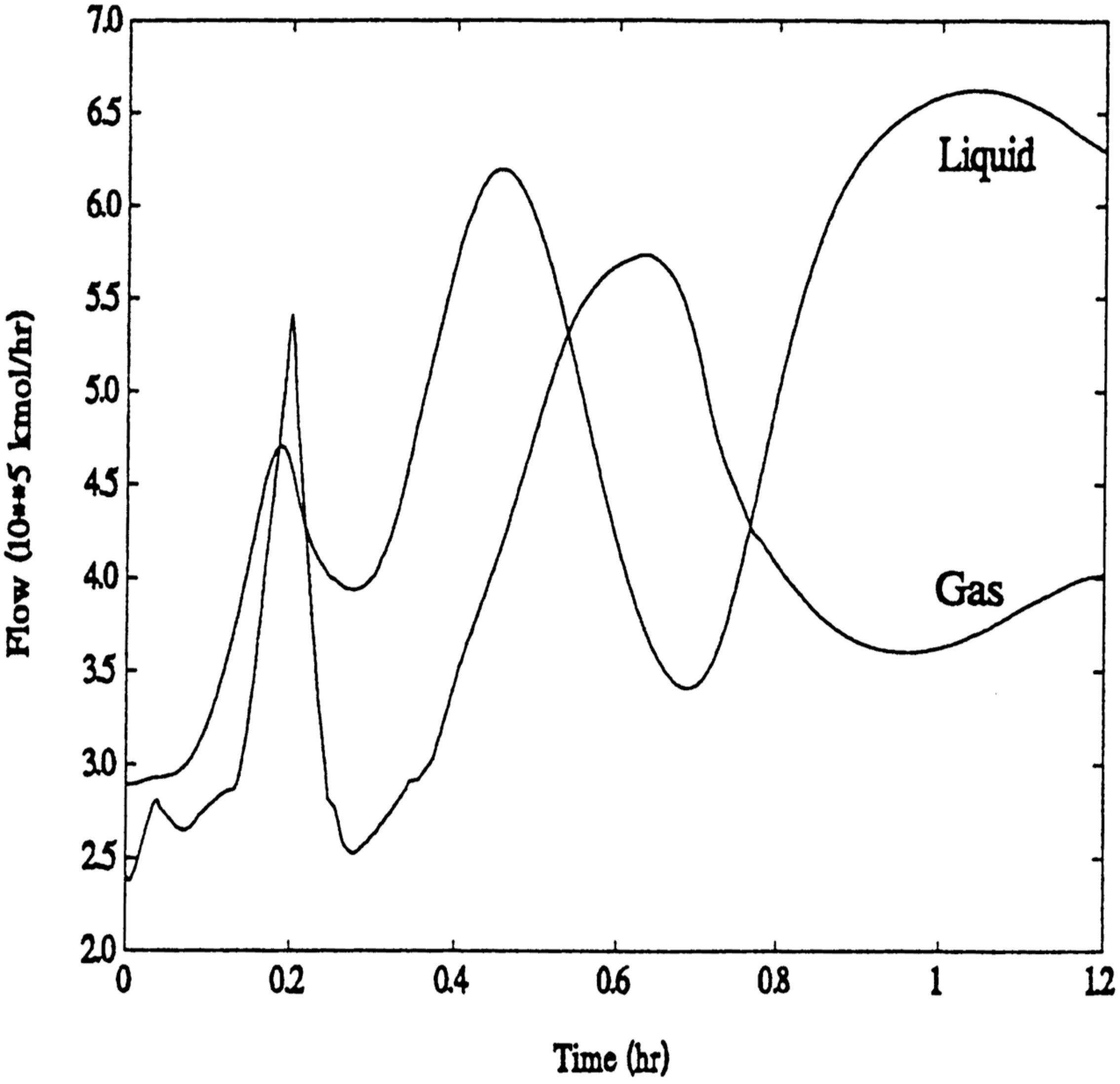


Fig. 4. Typical slug through the Tarbert pipeline to Oseberg.

Reprinted from *Computers Chem. Engng.* **17**, Naess *et al*., "Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants," pp. 585-600, Copyright 1993, with kind permission from Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington OX5 1GB, UK.

PRESSURE TRANSIENT IN FIRST SEPARATOR IN RESPONSE TO SLUG

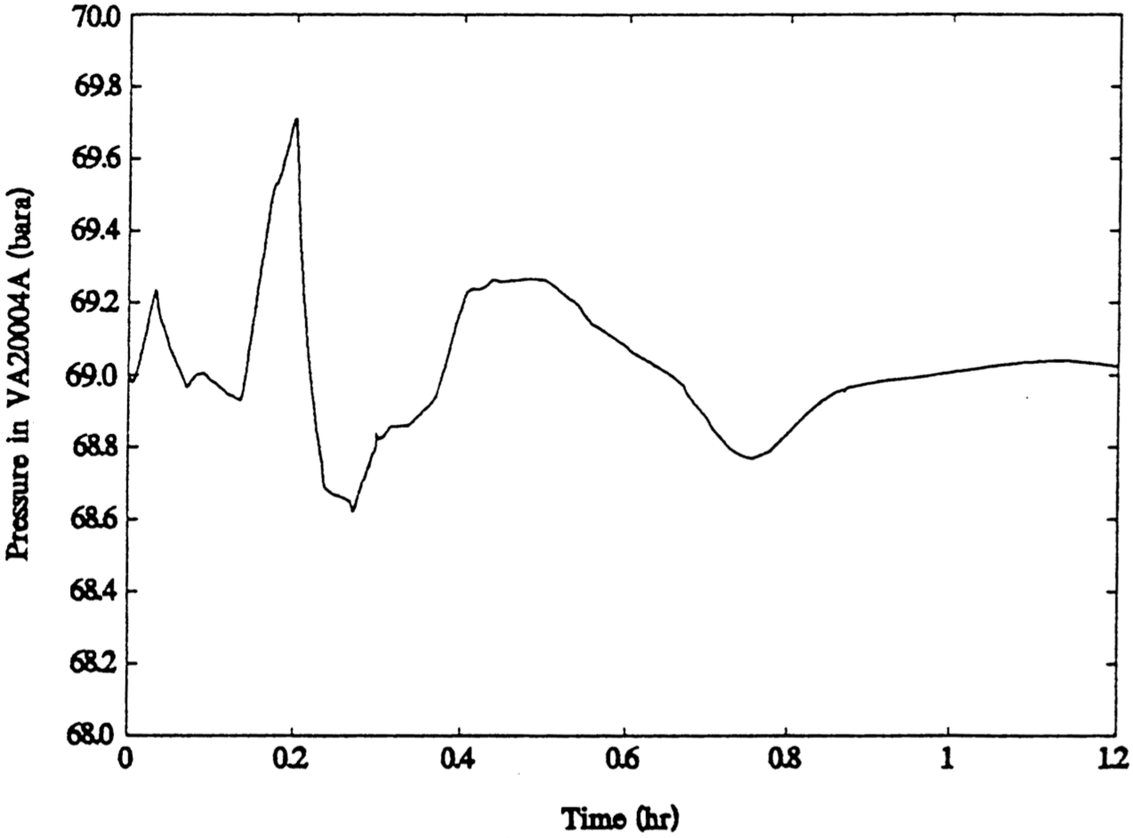


Fig. 5. Pressure in the first stage separator.

Reprinted from *Computers Chem. Engng.* **17**, Naess *et al.*, "Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants," pp. 585-600, Copyright 1993, with kind permission from Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington OX5 1GB, UK.

VALVE STEM POSITION TRANSIENTS IN RESPONSE TO SLUG

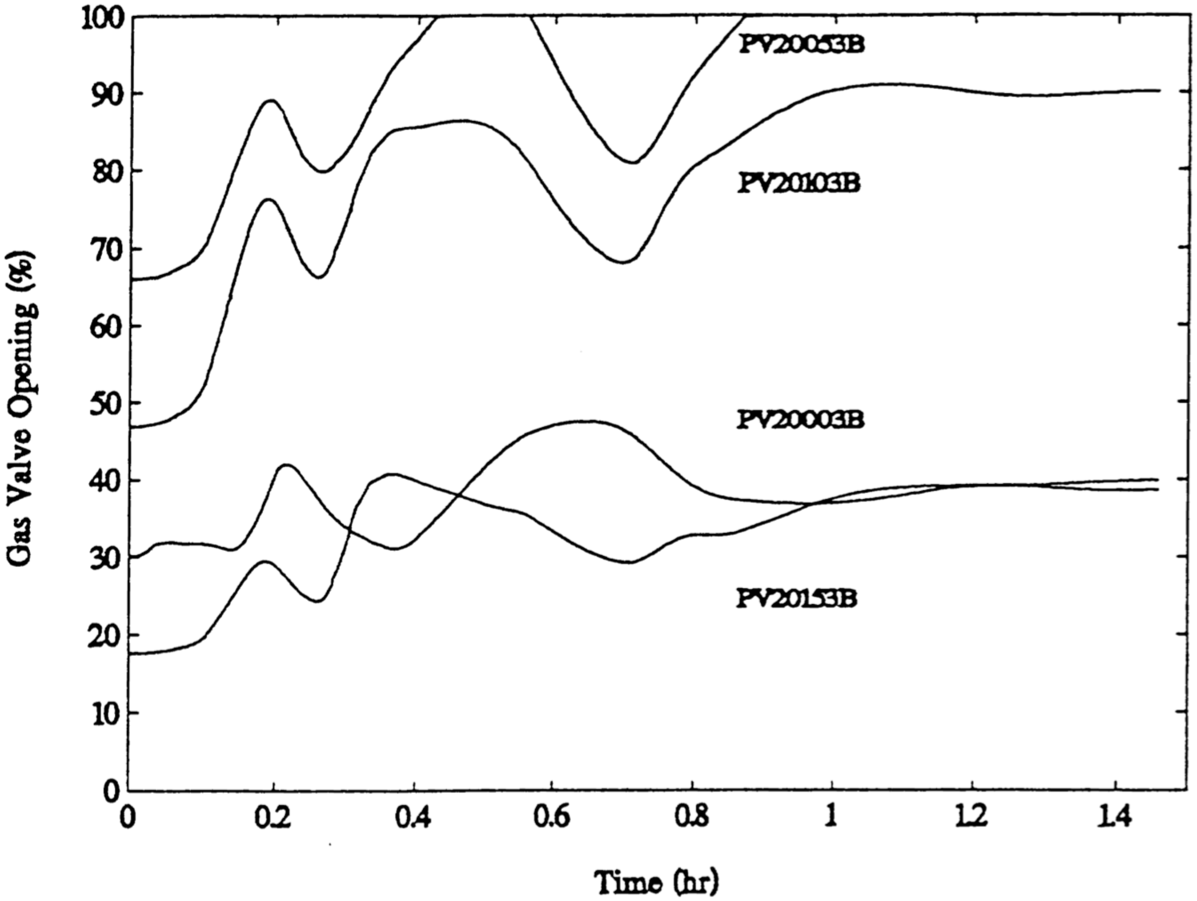


Fig. 6. Opening of the gas valves in the separation train.

Reprinted from *Computers Chem. Engng.* **17**, Naess *et al*., "Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants," pp. 585-600, Copyright 1993, with kind permission from Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington OX5 1GB, UK.

RETROFIT OF NORSK HYDRO TOPSIDE OFFSHORE OIL PRODUCTION PROCESS

BENEFITS

- problems lay elsewhere to what was anticipated: real value added from simulation!
- produce less from Osberg wells during startup of Tarbert pipeline to avoid level trip
- slugcatcher not required due to slow dynamics of slug

DESIGN OF PRODUCT GRADE TRANSITIONS FOR POLYOLEFIN PROCESSES

(Debling et al., 1994)

BACKGROUND

- wide variety of polyolefin products:
    - polypropylene
    - low and high density polyethylene
    - ethylene-propylene rubber

- number of catalytic processes
    - slurry
    - solution
    - gas-phase

...each has particular advantages

- but, each will be required to produce range of product grades to meet demand (typical product campaign varies from less than one day to several weeks)

OBJECTIVE

Design changeover policies that:

- conduct grade transitions as swiftly as possible
- minimize amount of off-spec product

...market conditions may favour one over the other.

DESIGN OF PRODUCT GRADE TRANSITIONS FOR POLYOLEFIN PROCESSES

APPROACH

- design changeover policy to move process between two steady-states: a “path” between these points in the time domain

- use dynamic simulation to predict response of the processes to a particular changeover policy, and experiment with improved changeover policies

Alternative: experiment with a real plant or pilot plant

ADVANTAGES (OF SIMULATION)

- **safety:** path taken by process must satisfy constraints to avoid, for example, temperature or pressure runaway:
  - *optimal changeover policy* subject to *path constraints*
  - constraints can be violated by a simulation without dire consequences: experiment with policies close to runaway

- **environment:** off-spec or vented material not created by a simulation!

- **economic:** process not taken out of production for experiments, can explore more alternatives in hope finding a better policy, potentially faster (opportunity cost), etc.

DESIGN OF PRODUCT GRADE TRANSITIONS FOR POLYOLEFIN PROCESSES

CASE STUDY[1]

- grade transitions in polypropylene homopolymer (PP) and impact copolymer (IPP) studied

- processes studied:

    - gas-phase fluidized bed (Union Carbide/Shell)
    - gas-phase vertical stirred bed (BASF/ICI)
    - gas-phase horizontal stirred bed (AMOCO/Chisso)
    - liquid pool loop (Himont)

- grade transition: increase number average molecular weight from 55,000 to 65,000 (stiffer product)

- use POLYRED - simulator specifically designed for polymer processes developed at the University of Wisconsin

[1] the original paper (Delbing *et al.*, 1994) contains several case studies of different products and processes

DESIGN OF PRODUCT GRADE TRANSITIONS FOR POLYOLEFIN PROCESSES

ALTERNATIVE CHANGEOVER POLICIES

In both cases, try to reduce accumulated hydrogen in the reactor as quickly as possible.

- strategies for gas-phase reactors

  A: i) step change in hydrogen feed rate

  B: i) step change in hydrogen feed rate
  ii) open gas vent for first ten minutes to change gas phase composition quickly

  ...mitigate large hydrogen residence time in gas phase reactors

- strategies for loop reactor

  A: i) step change in hydrogen feed rate

  H: i) step change in hydrogen feed rate
  ii) reduce recycle rate in loop to narrow residence time distribution

  ...narrow residence time distribution for hydrogen.

GAS-PHASE FLUIDIZED BED PROCESS

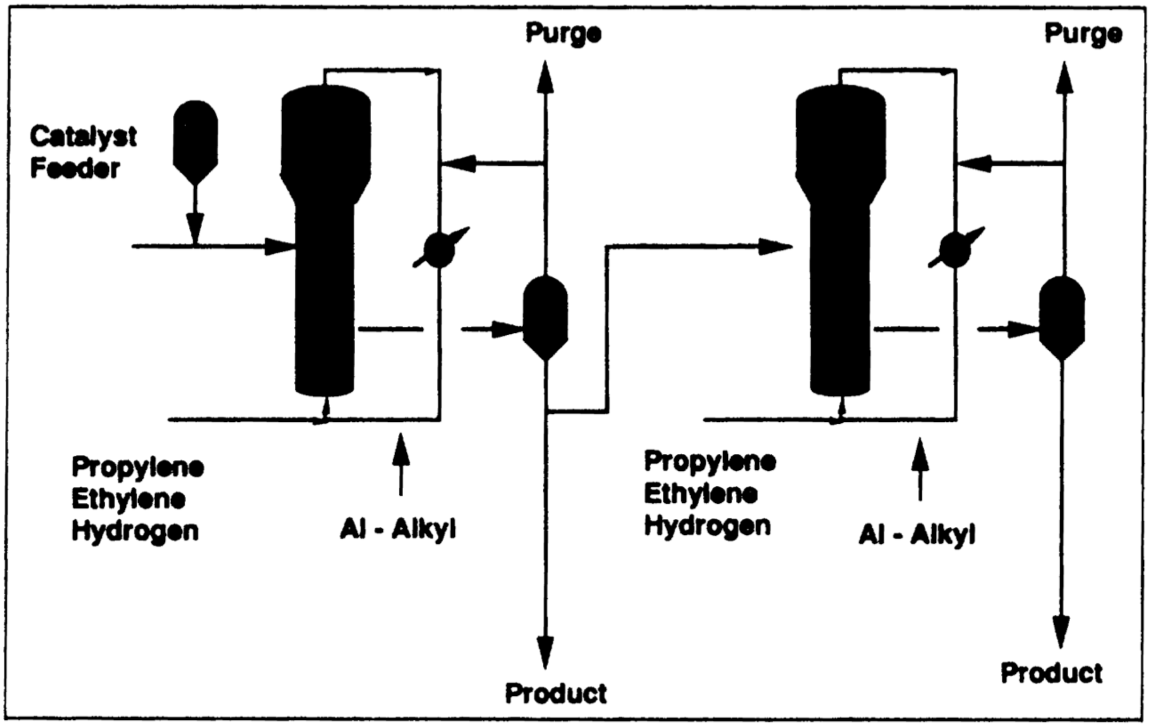


Figure 2. Union Carbide UNIPOL process.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institution of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

GAS-PHASE VERTICAL STIRRED BED

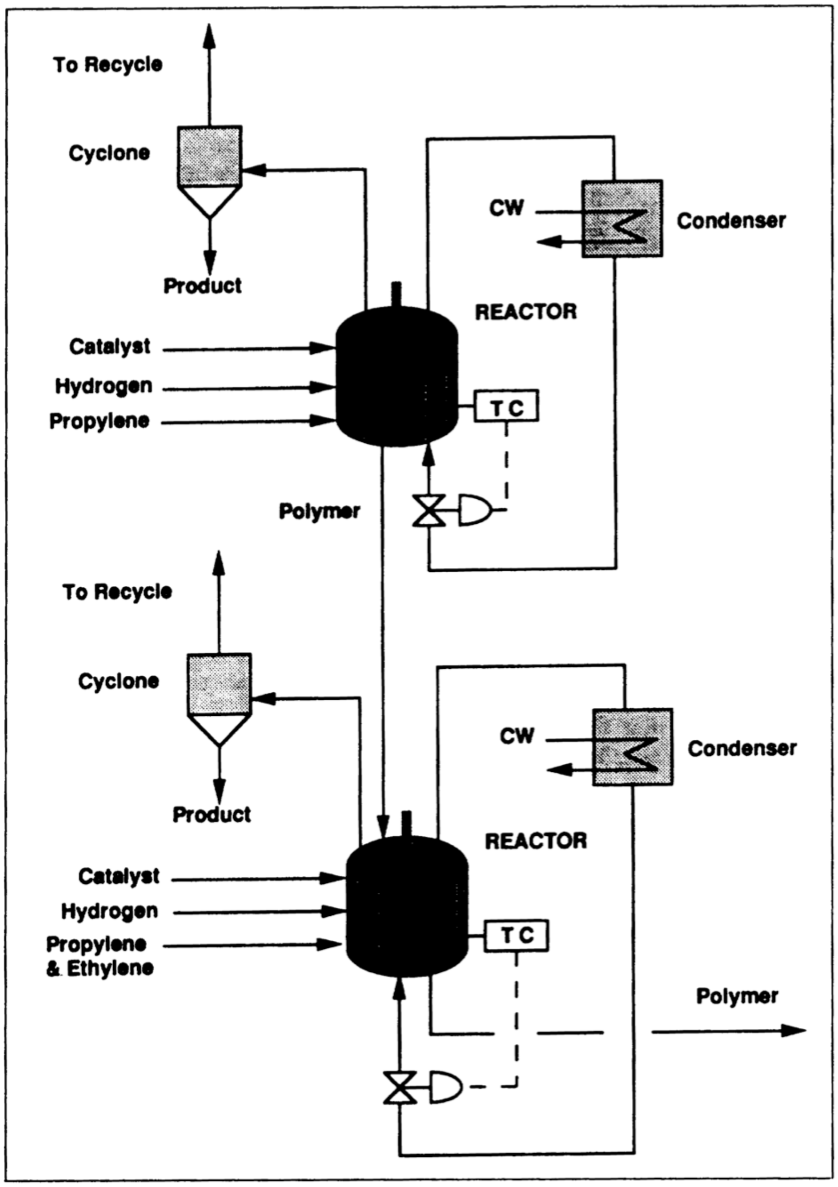


Figure 3. BASF gas-phase vertical stirred bed process.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institution of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

GAS-PHASE HORIZONTAL STIRRED BED

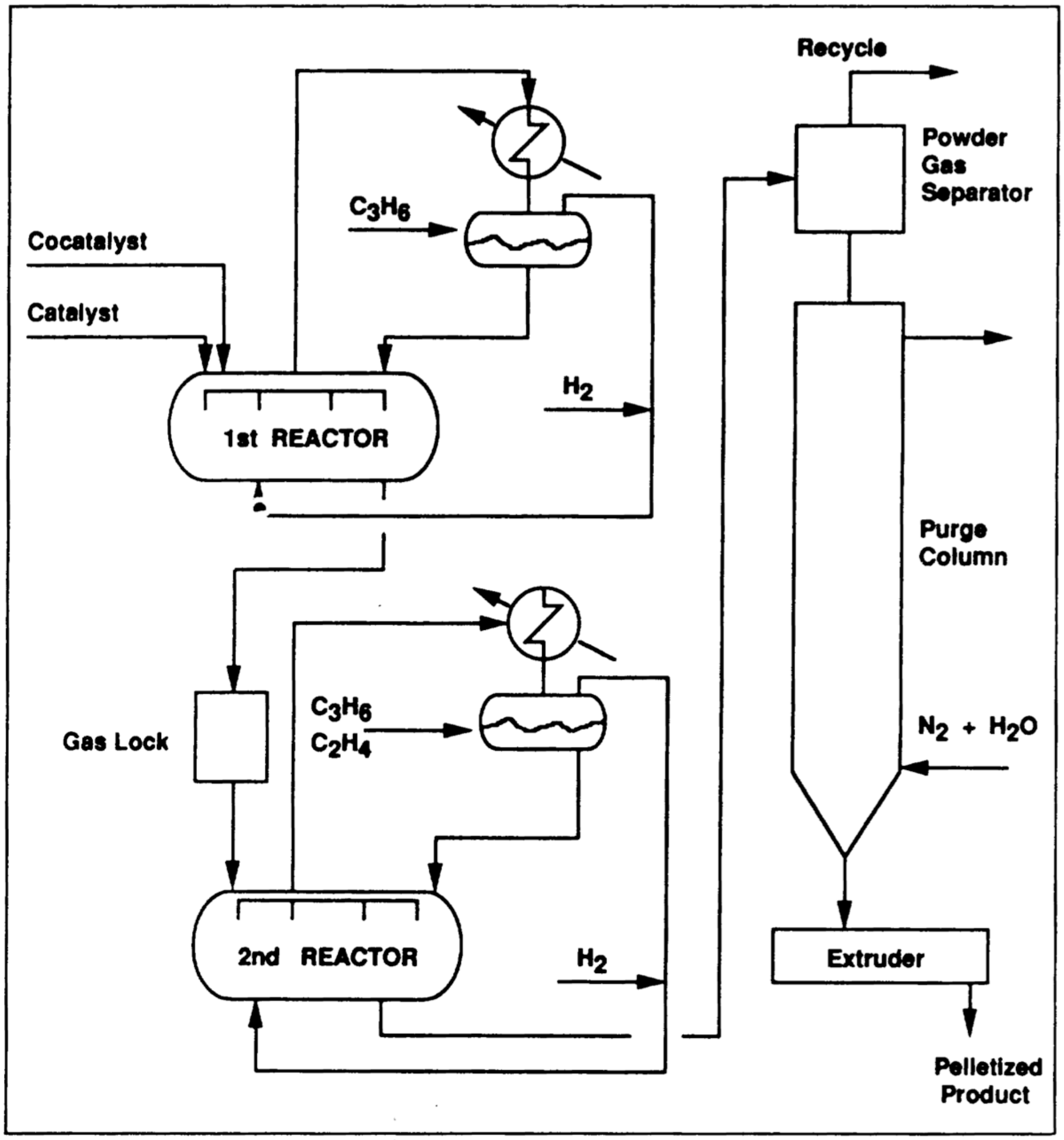


Figure 4. AMOCO/Chisso gas-phase horizontal stirred bed process.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institution of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

LIQUID POOL LOOP

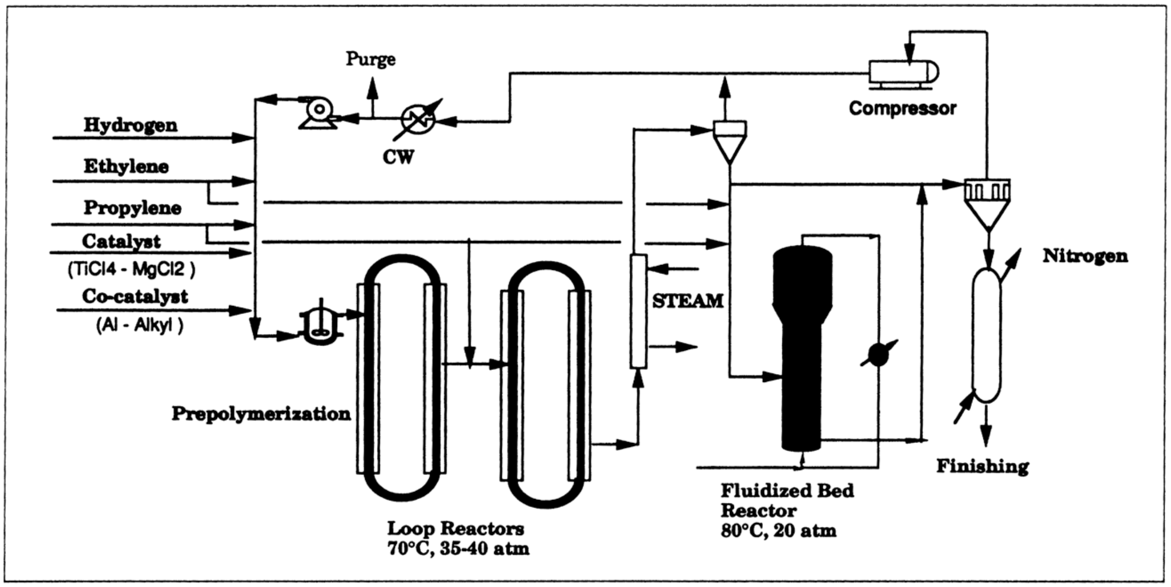


Figure 1. Himont spheripol process.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institute of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

NUMBER AVERAGE MOLECULAR WEIGHT DURING TRANSITION BASE CASE POLICY (A)

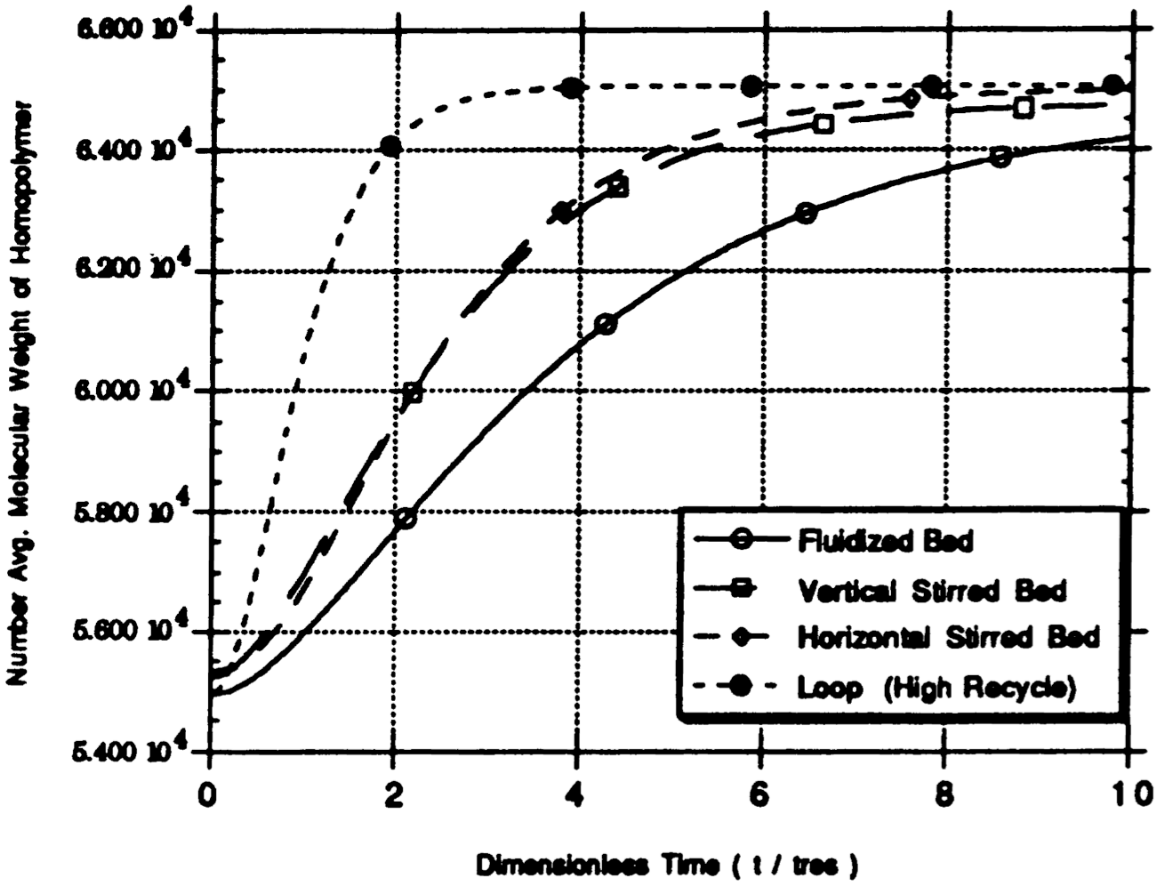


Figure 7. Grade change in molecular weight of polypropylene homopolymer, strategy A.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institution of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

NUMBER AVERAGE MOLECULAR WEIGHT DURING TRANSITION ALTERNATIVE POLICY (B or H)

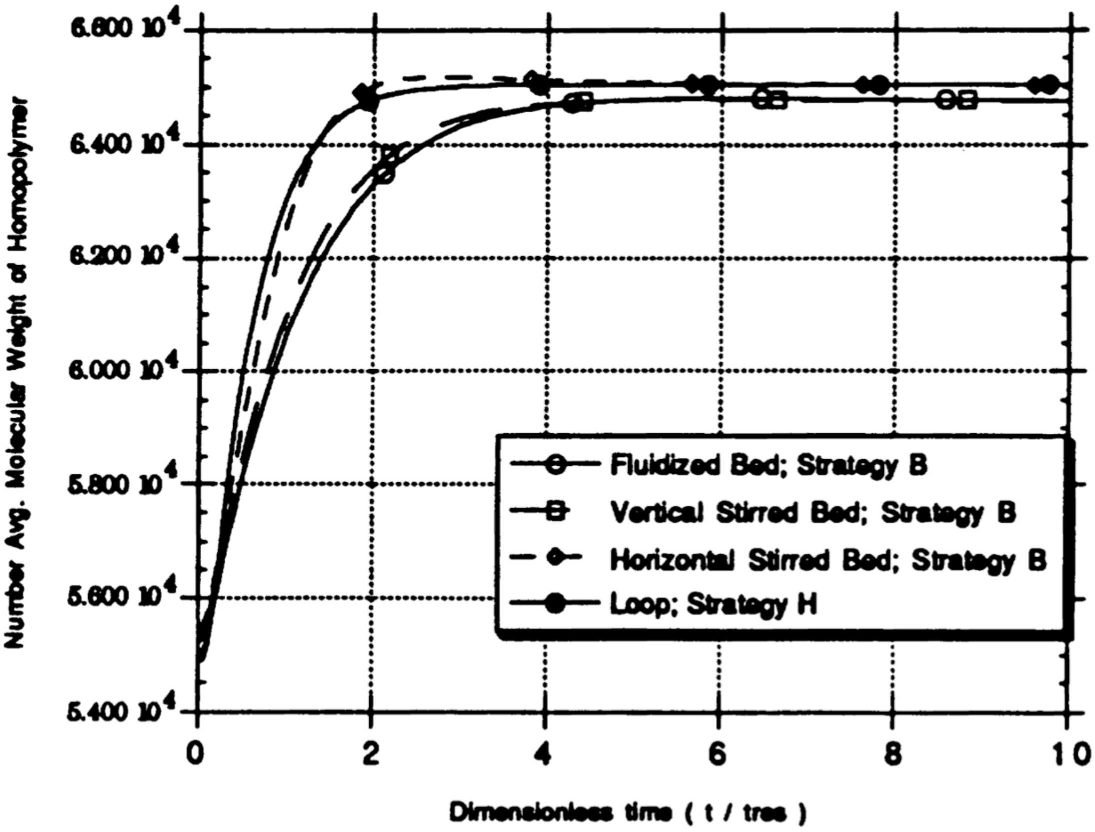


Figure 9. Grade change in molecular weight of polypropylene homopolymer.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institute of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

PERCENT OFF-SPEC PRODUCT DURING TYPICAL CAMPAIGN

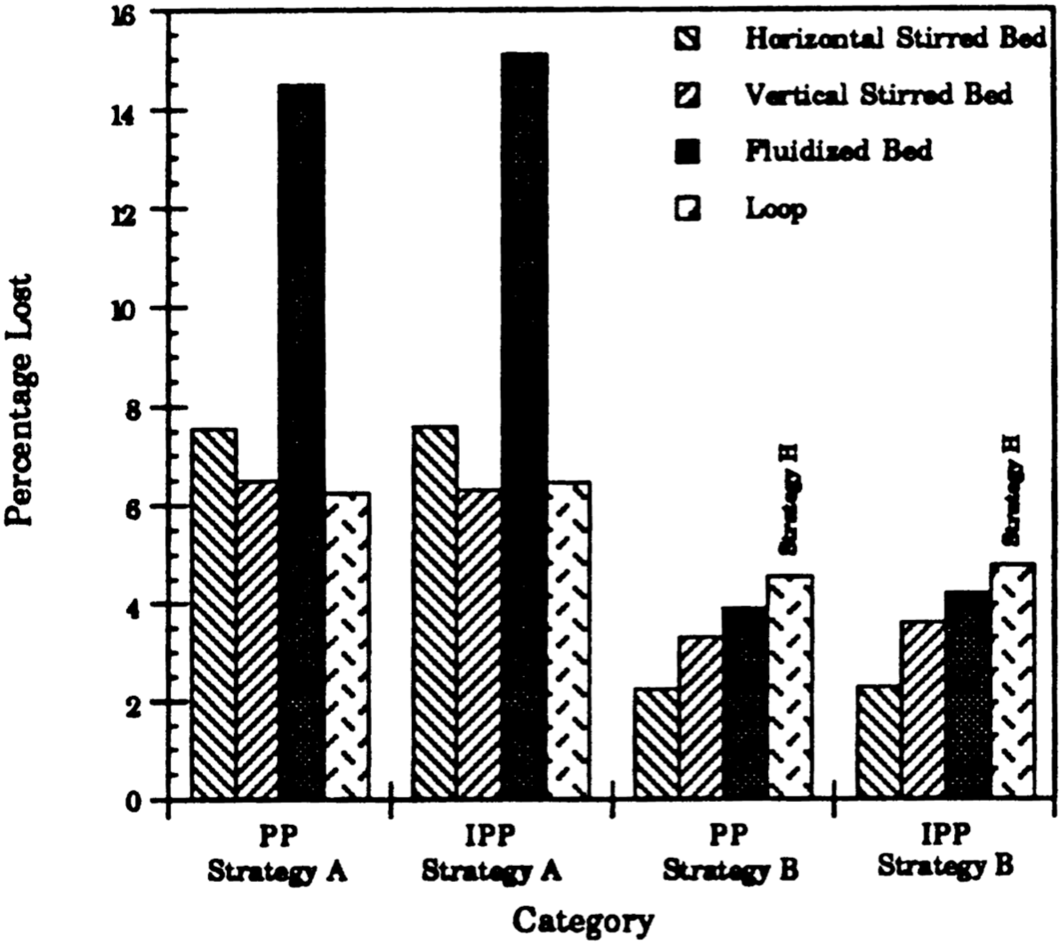


Figure 12. Percent off-spec product based on 1,000 ton of product within specification.

All processes are scaled to production level of 12.5 ton/h PP and 16 ton/h IPP.

from Debling *et al.*, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp.506-520, 1994. Reproduced with permission of the American Institution of Chemical Engineers. Copyright © 1994 AIChE. All rights reserved.

DESIGN OF PRODUCT GRADE TRANSITIONS FOR POLYOLEFIN PROCESSES

CONCLUSIONS

- for gas phase reactors, blow down to reduce hydrogen accumulation can greatly reduce changeover time (holdup/accumulation of hydrogen dominates speed of changeover)

- careful design of operating policies can change relative rankings of processes

  - gas phase reactors now exhibit "natural" residence times

- these conclusions reached in the comfort and safety of the office!

BATCH PROCESS DEVELOPMENT: SILOXANE MONOMER PROCESS

(Allgor et al., 1996; Ahmad and Barton, 1994)

BACKGROUND

- effective process development critical to success of a new specialty chemical or synthetic pharmaceutical.
  - develop efficient process rapidly
  - environmental problems: use and disposal of organic solvents

- can detailed modeling technology help?

- batch processes ⇒ dynamic simulation

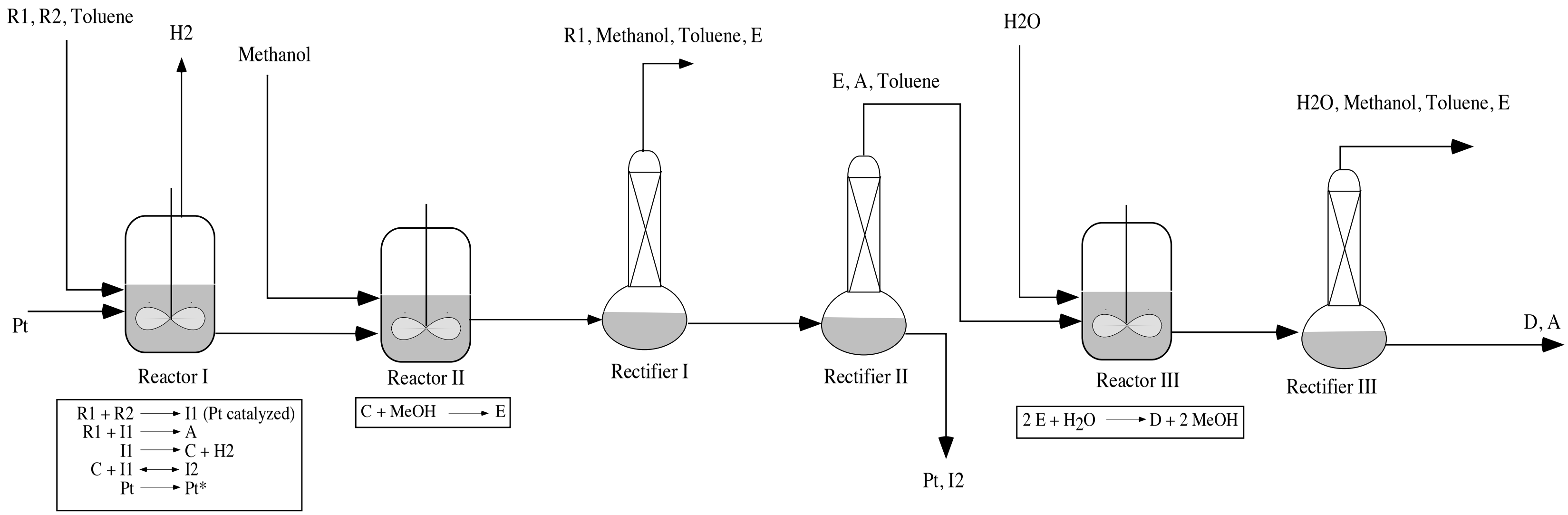
- siloxane monomer process (Allgor et al., 1996):
  - products A and D required in arbitrary ratio: A formed in first reaction, byproduct C further reacted to D.
  - allyl alcohol (R1 in figure), toluene, methanol and water used as solvents and reagents
  - complex reaction kinetics, highly nonideal mixtures

OBJECTIVE

Can detailed dynamic models of a batch process help in the design of a process with better environmental performance?[2]

[2] for more lengthy discussion see Ahmad and Barton (1994)

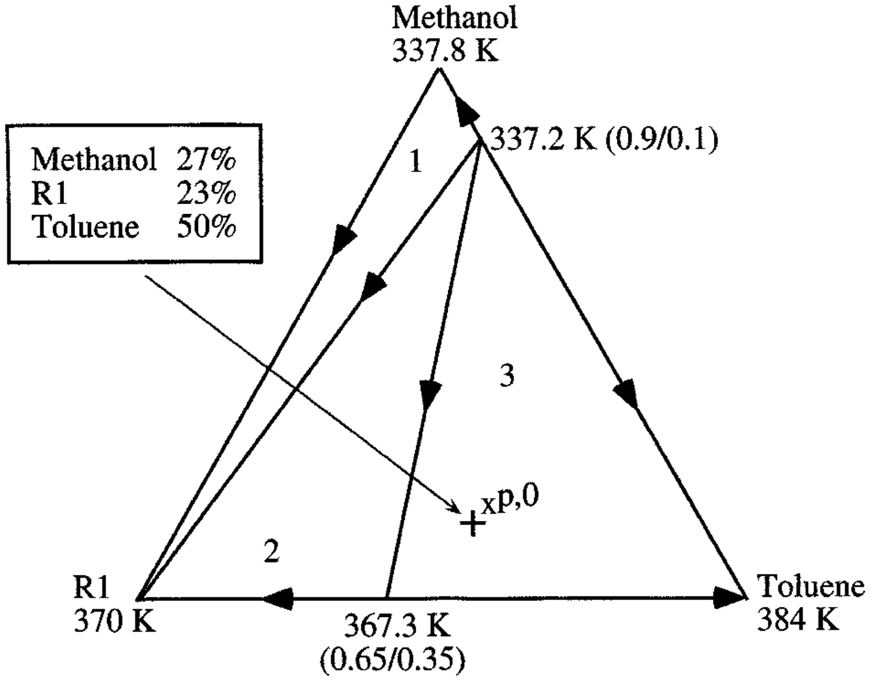
SILOXANE MONOMER PROCESS FLOWSHEET



BASE CASE DESIGN: PROBLEMS

- process developed in pilot plant simulated in ABACUSS to quantify all process streams (on a batch size basis)

- organic waste stream: overhead from Rectifier I. Analyze solvent recovery possible using theory of simple distillation residue curve maps:



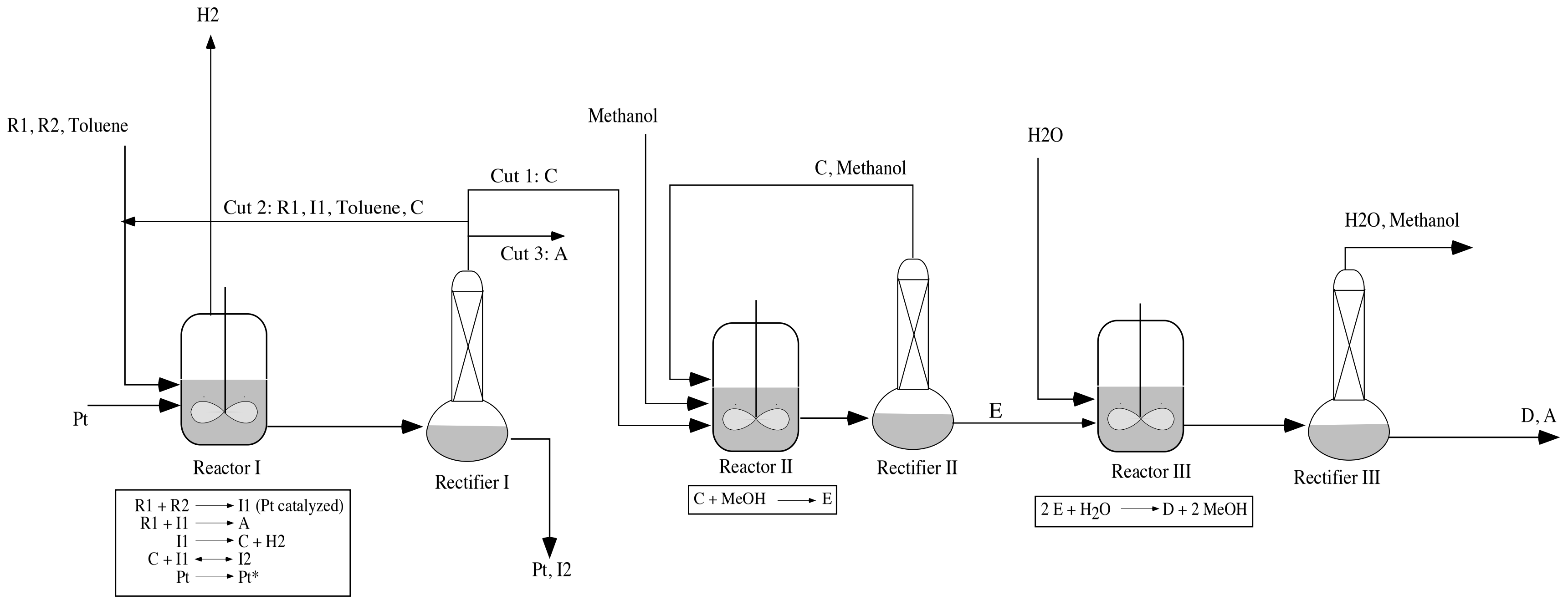
Conclusions:

- only toluene can be recovered in pure form for recycle
- 30% of stream disposed of by incineration.

- aqueous waste stream: solvent recovery complicated by presence of toluene (heterogeneous mixture, azeotropes)

- don't let toluene get downstream from Reactor I?

PROCESS ALTERNATIVE: INSERT RECTIFIER FOLLOWING REACTOR I



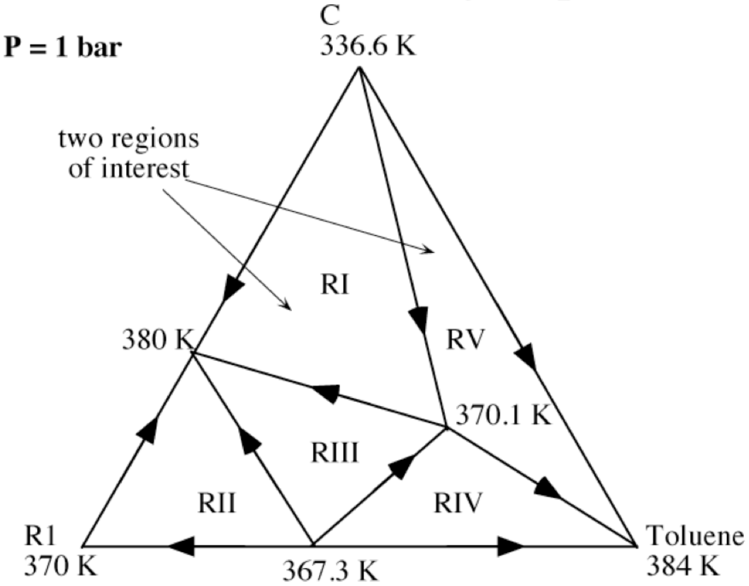
$R1 + R2 \longrightarrow I1$ (Pt catalyzed)
$R1 + I1 \longrightarrow A$
$I1 \longrightarrow C + H2$
$C + I1 \longleftrightarrow I2$
$Pt \longrightarrow Pt^*$

$C + MeOH \longrightarrow E$

$2 E + H_2O \longrightarrow D + 2 MeOH$

IS PROCESS ALTERNATIVE FEASIBLE?

- analyze separation alternatives for mixture leaving Reactor I: is recovery of pure C feasible?



Cut 1: C
RI: Cut 2: C-R1-Toluene
Cut 3: C-R1

Cut 2: C
RV: Cut 2: C-R1-Toluene
Cut 3: Toluene

Conclusions:

- if mixture created by Reaction I lies in regions I or V then process is feasible
- some C will also be recycled in azeotropes with solvent and/or reagent to next batch

- recycle of C will lead to build up of C in the reactor to a *cyclic steady-state*[3] concentration:

$$R1 + R2 \rightarrow I1 \text{ (Pt catalyzed)}$$

$$R1 + I1 \rightarrow A$$

$$I1 \rightarrow C + H2$$

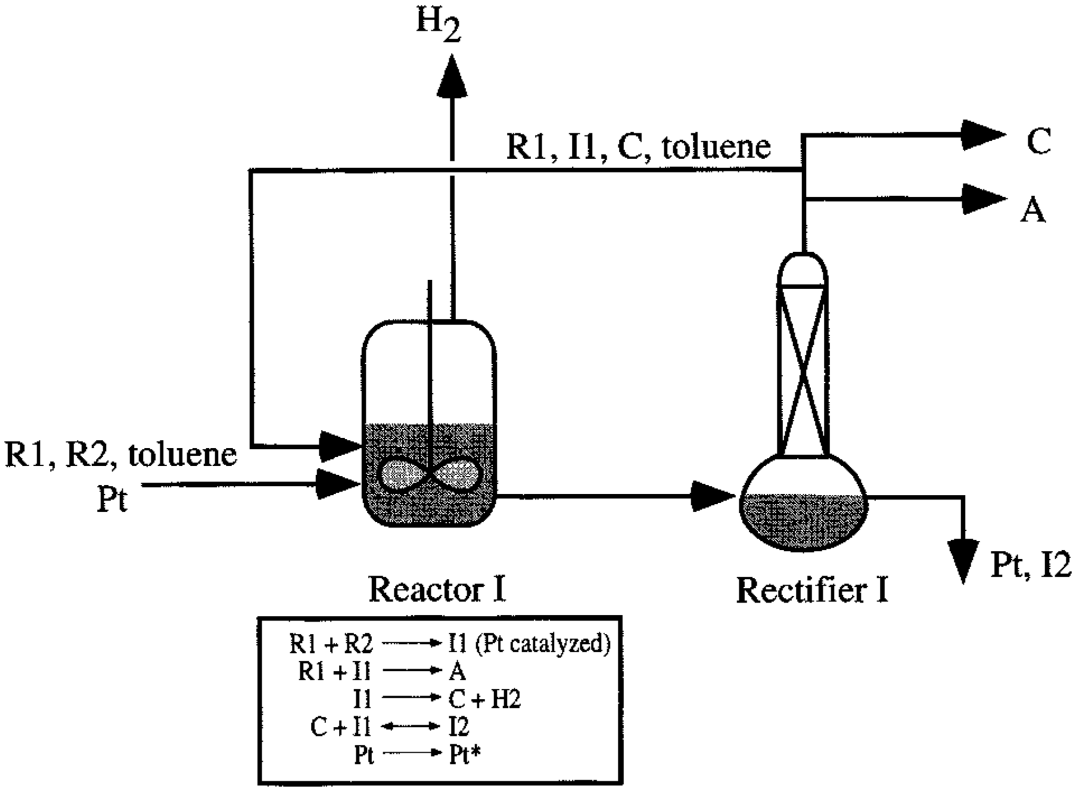
$$C + I1 \leftrightarrow I2$$

$$\text{Pt} \rightarrow \text{Pt}^*$$

...encourages formation of undesired oligmer I2.

[3] a cyclic dynamic system is said to have reached cyclic steady-state when the variation of the variables (profiles) over a cycle is the same from one cycle to the next

DESIGN OPERATING POLICY FOR COUPLED REACTOR AND COLUMN



DESIGN OBJECTIVES

At cyclic steady-state:

- mixture at the end of Reaction I must be in either batch distillation region I or V
- formation of undesired oligmer I2 must be minimized

KEY DESIGN VARIABLES

- charge and temperature policy for reactor (amount of solvent and reagent at end, control conc. of I1)
- reflux policy for column (recycled cuts do not need to be sharp)

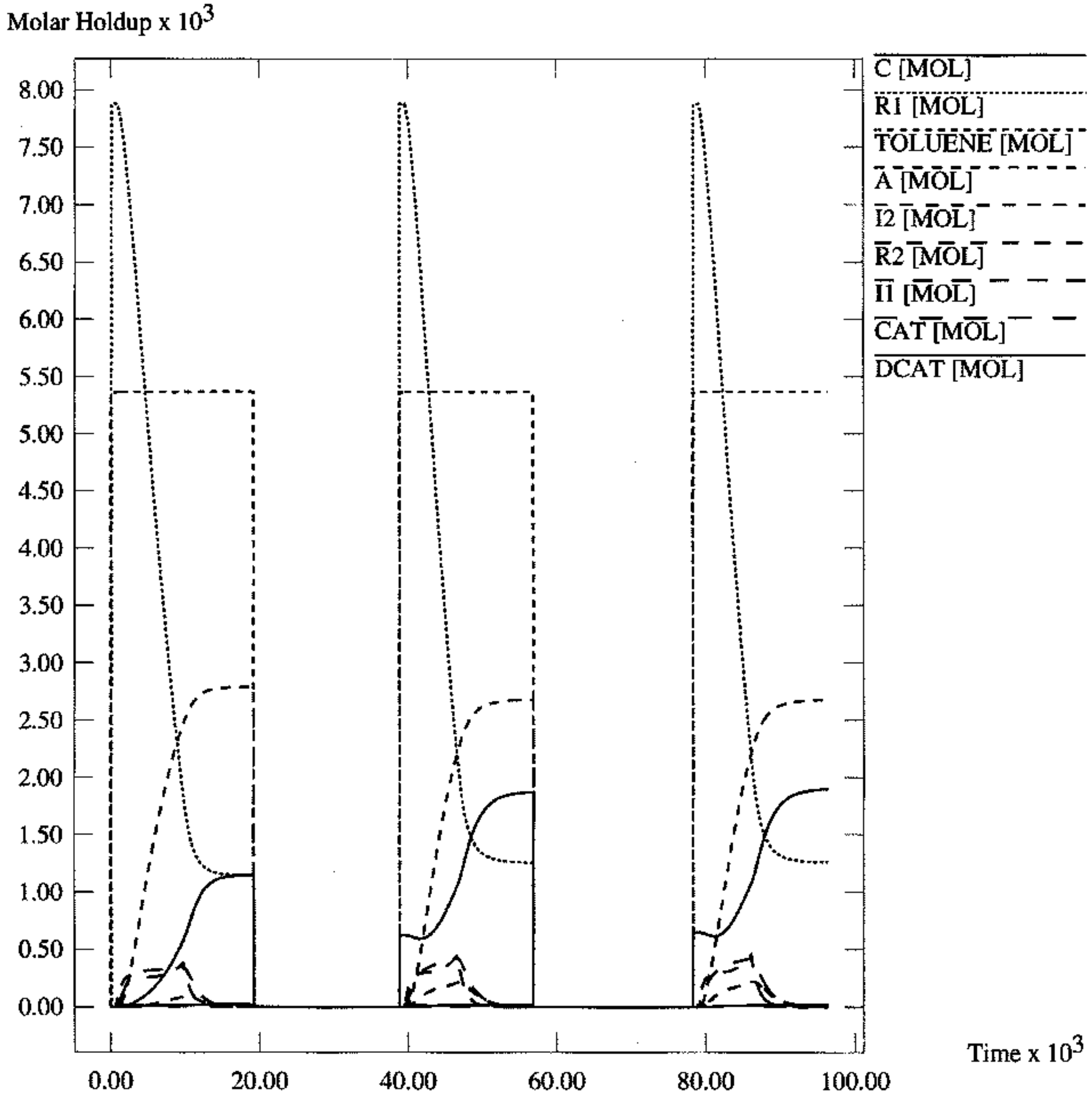
ROLE OF DYNAMIC SIMULATION

- rigorous dynamic model of coupled reactor and column can:
    - predict cyclic steady-state for given operating policy
    - allow experimentation with operating policy to meet design objectives
    - cheaper, safer, quicker, better (more alternatives explored) than pilot plant

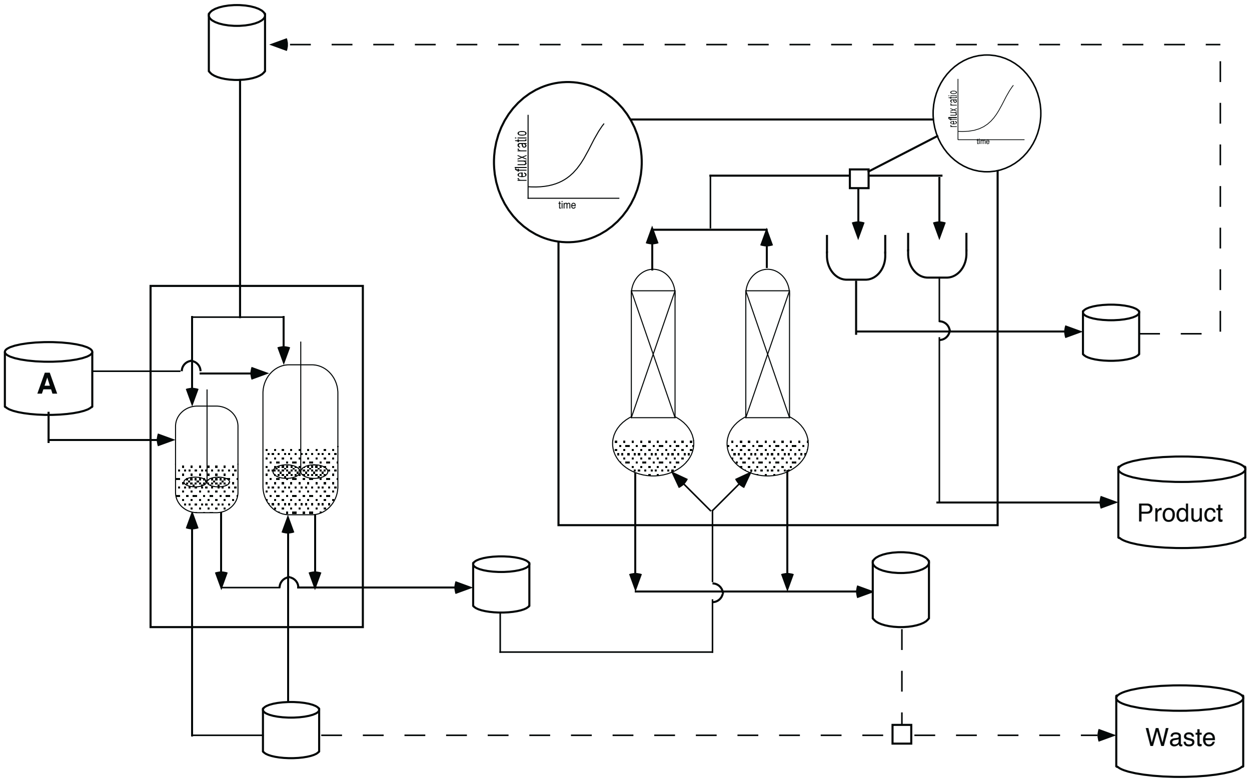
- complex dynamic model of reactor and column must reflect:
    - nonideal thermodynamics: azeotropic mixtures
    - complex reaction kinetics
    - sequential operating policies for reactor and column, and intermediate storage and recycle of solvent/reagent
    - physico-chemical discontinuities: reaction mass boiling, intensive quantities not defined in empty vessel, etc.

- need state-of-the-art combined discrete/continuous dynamic simulator: ABACUSS
    - 2657 equations (including physical properties)
    - design of suitable operating policy achieved

MOLE HOLDUPS IN REACTOR I: APPROACH TO CYCLIC STEADY-STATE



OPTIMAL CYCLIC STEADY-STATE DYNAMIC OPTIMIZATION



BATCH PROCESS DEVELOPMENT: SILOXANE MONOMER PROCESS

BENEFITS

- total organic waste reduced by 88% compared to the base case (with central solvent recovery facility) - only waste as unavoidable consequence of stoichiometry left
  - in process recycling of solvent more effective!

- reduced load on downstream unit operations: only C passed to rest of process, selectivity optimized with respect to A

- raw material saved (solvent/reagent)

- aqueous waste easier to treat

- throughput in same equipment similar

- targeted use of dynamic simulation in conjunction with pilot trials and simpler models (e.g. residue curve maps) leads to effective problem solving strategy

SECTION I

DYNAMIC MODELLING

Paul I. Barton
Department of Chemical Engineering
Massachusetts Institute of Technology
Cambridge, MA

© copyright Paul I. Barton, July 1997

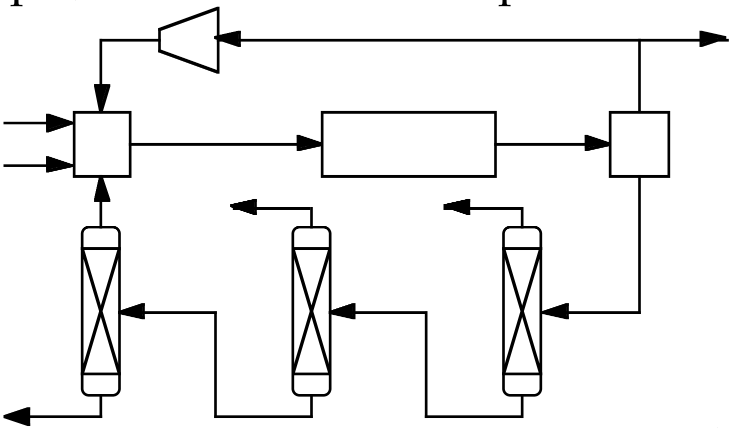
MODELLING DYNAMIC BEHAVIOUR OF PROCESS SYSTEMS

Three key ideas:

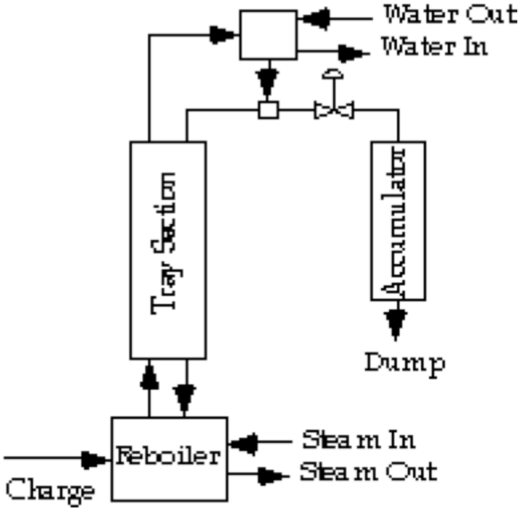
(a) model *system* dynamics

System ≡ complex artifact composed of connected and interacting components.

For example, an entire chemical process:



...or, a complex unit operation - e.g., a batch distillation column:

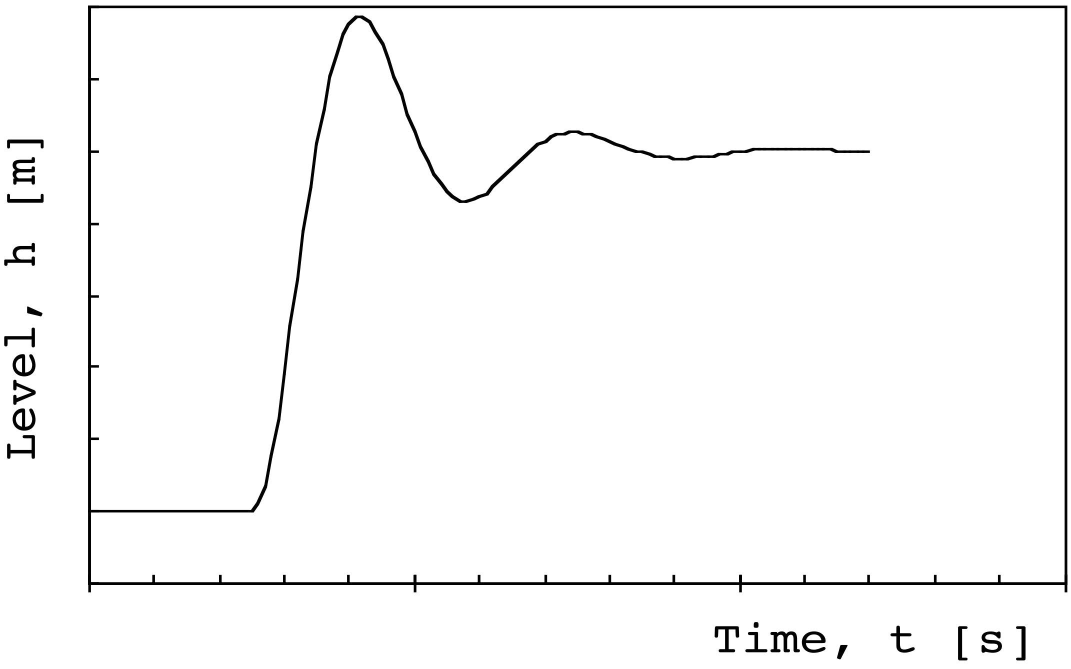


MODELLING DYNAMIC BEHAVIOUR OF PROCESS SYSTEMS

(b) focus on deterministic first principle based (mathematical) models

- emphasis on lumped parameter systems
- model will always rely to a certain extent on empirically derived parameters (e.g. reaction kinetic parameters) and correlations (e.g. distillation tray hydrodynamics).

(c) goal is to predict the evolution of the system behaviour with respect to time. For example, dynamic simulation results are typically presented as time trajectories:



Note that there is more to a transient than the steady-states.....

MODELLING – CAVEATS

Any mathematical model (by definition) is an abstraction of the true system behaviour.[1]

☹ does the model exactly mimic the true behaviour?

☺ does the model predict the aspects of system behaviour (*phenomena*) of interest with sufficient accuracy for the current application?

This implies:

⇒ a model is *only* valid within the context and assumptions it was developed.

⇒ extrapolation of the model beyond this context and assumptions is extremely dangerous - critical reappraisal of the model required (hence need to document the model)

⇒ verify a model against the real system behaviour whenever possible.

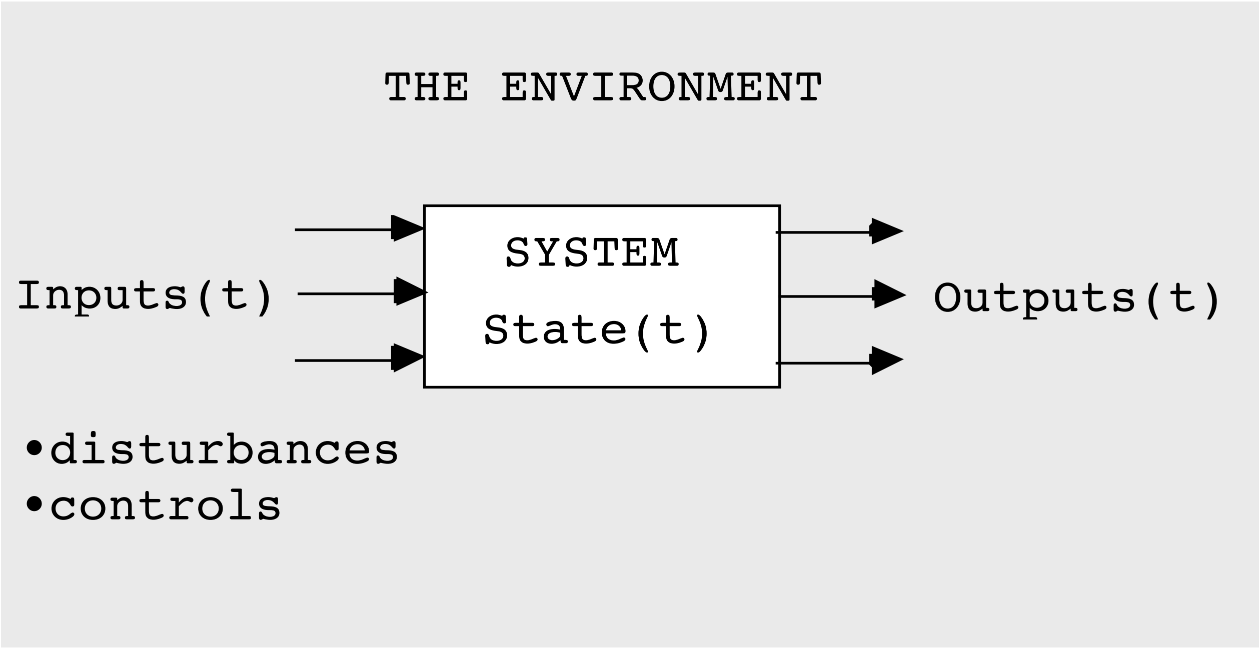
⇒ there are many models for a single system — each represents a different level of abstraction. Key task is to select the *appropriate* level of abstraction.

[1] in fact, any "perfect model" would exhibit such complexity that it would be indistinguishable from the real system.

PRELUDE TO MODELLING

Because of this fundamental nature of models, before and during any modelling activity it is important to clarify and document the following information:

(a) identify the system for which a model is required



Identify:

- boundaries (function of system) - a system is defined by its boundary
- constraints
- quantities describing system behaviour: inputs, states, outputs
- assume: inputs $\neq f$(outputs)

PRELUDE TO MODELLING

Notes:

- we are developing a model for the *system*. Everything else is the *environment*.
- inputs define the influence of the environment on the system.
- while in real life the inputs will be further subdivided into:

  (a) controls — those inputs that can be manipulated in order to control the system behaviour.

  (b) disturbances — those inputs over which we have no control.

  . . . from the point of view of modelling we can play God: we must define the time variation of all the inputs in order to pose a fully determined simulation problem.

- inputs $\neq f$(outputs):

  — more precisely: we can sufficiently decouple the influence of the outputs on the inputs (feedback via the environment) for the purposes of the current exercise.

  — pragmatic view: can only model so much at a given level of detail.

PRELUDE TO MODELLING

(b) what is the intended application of the model?

What questions will be asked about the system?
Begin to identify:

- what phenomena are of interest?
- what quantities describe the system behaviour?
- how detailed should the model be?
- what assumptions can be made?

(c) what data concerning the system is available?

or can be obtained . . . imposes constraints on the phenomena that can be modelled and the accuracy of the simulation results.

Typically relates to parameters and empirical correlations:

- what is available?
- in what range of process conditions are the predictions valid?
- how much uncertainty is there in the predictions?

Examples: non-equilibrium distillation tray models, reaction kinetics for novel synthesis.

PRELUDE TO MODELLING

In conclusion: modelling is a fundamental engineering activity. We must trade the need to obtain an answer against the accuracy (or even validity) of this answer, and the resources required to obtain it (time, manpower, skills, computing resources, data, etc. . . .)

=> principle of *optimum sloppiness* — make as many simplifying assumptions as reasonable without throwing out the baby with the bath water (Luyben, 1990)

...modelling is still very much an art (but research continues).

STEP 1 — HIERARCHICAL DECOMPOSITION

No engineer can grasp simultaneously all the relevant details of a complex system: e.g., how do we construct a mathematical model composed of thousands of simultaneous equations?

Two approaches to manage system complexity in model building:

A: Top Down Approach

(a) identify a series of components from which the system is composed, and the connections between these components (e.g., a flowsheet, an electrical circuit, organs in the human body, etc.).

(b) once identified, concentrate on modelling and testing each individual component in isolation as a separate task.

B: Bottom Up Approach

(a) identify a series of standard *primitive* components required to describe a class of systems. Develop and validate models for these primitive components (archive in library).

(b) develop models of more complex systems by connecting these components together to form structures (systems).

STEP 1 — HIERARCHICAL DECOMPOSITION

In my experience, a hybrid of the top down and bottom up approaches is used:

- system structure is invariably unraveled in a top down manner, but . . .
- model libraries are usually available, and these are kept in mind when selecting a decomposition.

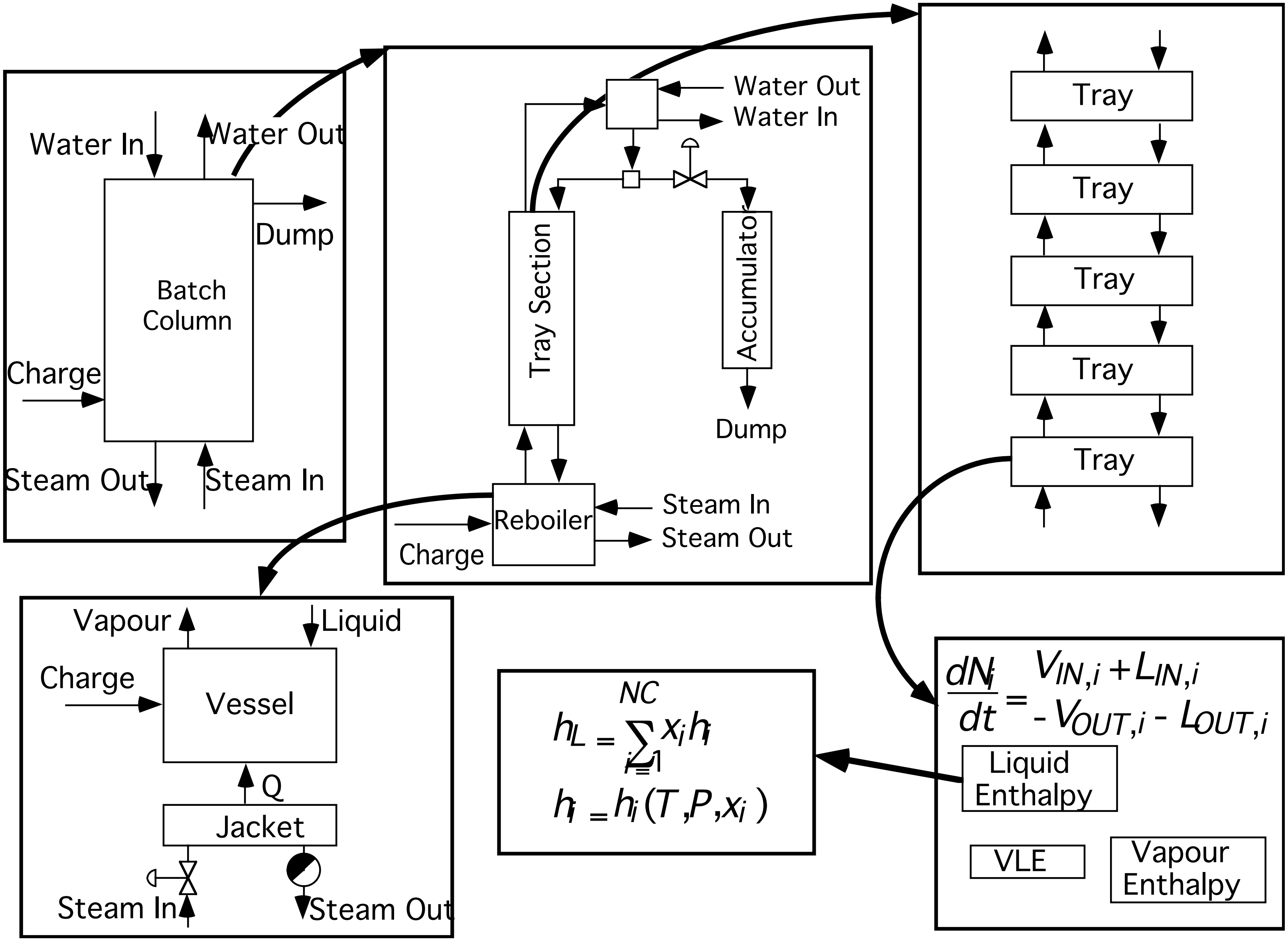
Hierarchical Submodel Decomposition (Elmquist, 1978) introduces the notion of recursion into these approaches — a component model can either be:

(a) *primitive* — e.g., it is entirely described in terms of equations and is not decomposed further.

(b) *composite* — e.g., it is treated as a system itself, and can be therefore described in terms of a set of interconnected components.

. . . the introduction of recursion allows the decomposition of composite models to continue into as many levels as is necessary - hierarchies of arbitrary depth can evolve.

HIERARCHICAL SUBMODEL DECOMPOSITION - BATCH COLUMN EXAMPLE



$$\frac{dN_i}{dt} = \begin{array}{l} V_{IN,i} + L_{IN,i} \\ - V_{OUT,i} - L_{OUT,i} \end{array}$$

$$h_L = \sum_{i=1}^{NC} x_i h_i$$

$$h_i = h_i(T, P, x_i)$$

STEP 2 — MODELLING THE COMPONENTS

Each component model is itself a system (recursion):

(a) identify the boundaries.

- the model may represent a physical artifact — e.g., a plant section, a unit operation, a vessel, or a section of a vessel.
- the model may represent some phenomenological abstraction — e.g., the set of equations defining a mixture enthalpy, or a set of reaction rate expressions.

(b) identify the connections of the model to its environment - this will enable us to connect the model to others in a larger structure.

Connections will typically represent either:

(i) *fluxes* of extensive properties, such as:

- a diffusive flux - e.g., heat transfer through a vessel wall
- a convective flux - e.g., a pipe connecting two vessels (ignoring the holdup of material in the pipe)

Note fluxes will have a *direction* implied.

(ii) information transfer - e.g., pressure and/or voltage signals for control instruments, or intensive properties (e.g., temperature, pressure) to determine driving forces for fluxes

STEP 2 — MODELLING THE COMPONENTS

Structures are defined by establishing equivalence (merging) between the connection of one component model and the connection of another component model (obviously, the two connections must be compatible - e.g., must represent a convective flux)

(c) **define the internal behaviour of the component model** - what describes the "state" of this system, and how is it related to both the inputs and the outputs.

INTERNAL BEHAVIOUR

Ultimately internal behaviour is represented by a set of:

```
                    Inputs
VARIABLES - e.g.    States
                    Outputs
```

...and these variables are related by a set of:

```
                    Mass Balances
                    Energy Balances
EQUATIONS - e.g.    Physical Constraints
                    Thermodynamic Models
                    etc.
```

However, there are several options as to how these two sets can be defined:

(a) the model is primitive — e.g., just define a set of variables and a set of equations

(b) decompose the model further — e.g., define a set components and their connections. The set of variables is then the union of the sets of variables in the sub models, and similarly the set of equations is the union of the sets of equations in the sub models and the equations implied by the connections

(c) a hybrid of (a) and (b) — e.g., both variables and sub models

DERIVING THE EQUATIONS

Once a suitable decomposition has been established, it is necessary to develop all the primitive models - e.g., derive a set of variables and equations that describe the dynamic behaviour of that section of the overall system.

Basically, we must utilize our knowledge of chemical engineering science at this point - control volume analysis, mass conservation, energy conservation, etc.

Experience with teaching this material suggests that it is worthwhile to review the principles frequently used in dynamic modelling - but this discussion is by no means exhaustive (nor is my experience!). The approach is relatively systematic and will allow derivation of component models for a system that are consistent.

DERIVING THE EQUATIONS

The following information should be developed at this stage:

1. **define the control volume(s) for balance equations** (again. . .identify the boundaries)

2. **list the assumptions employed** — under what conditions is the model valid?

   Example: phase transitions

3. **list the set of variables required to describe the system** — e.g., symbol, verbal description, units (*always* try to use a consistent set of units).

4. **derive the set of equations** — *always* check that the units of each term in an equation are consistent.

5. **perform a *degree of freedom analysis*:**

   (a) identify the natural set of input variables to the system.

   (b) given values for this subset of the variables it should be possible to calculate values for all the other variables — e.g. it is necessary that the number of equations equal the number of remaining unknown variables (ignore time derivatives of variables in this analysis).

DERIVING THE EQUATIONS

We will be applying one or more conservation principles to a macroscopic control volume of definite size and shape containing a fluid.

Further, we will assume that the contents of the control volume are well mixed, so intensive properties are uniform throughout the control volume and do not vary with spatial position (or with another independent variables such as polymer chain length).

=> one or more *ordinary differential equations* (ODEs) with time as the independent variable.

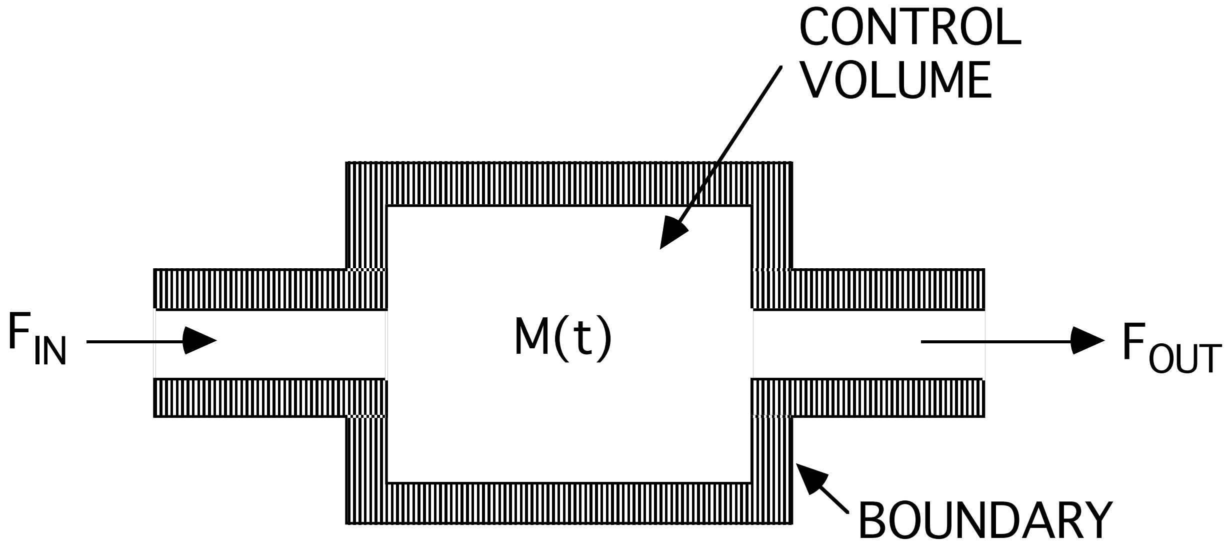
These ODEs will be augmented with *algebraic equations* (e.g., equations not involving time derivatives) that relate variables or model phenomena that take place on a much faster time scale than that of interest, for example:

- a pseudo steady-state assumption
- see discussion of phase equilibrium models later on

Note that in the absence of a vacuum, material will expand (possibly with phase change) to completely fill the control volume.

MASS CONSERVATION

Mass is always conserved[2]:



Mass balance:

$$\begin{bmatrix}\text{Rate of}\\ \text{accumulation of}\\ \text{mass in control}\\ \text{volume}\end{bmatrix} = \begin{bmatrix}\text{Flow of}\\ \text{mass into}\\ \text{control volume}\end{bmatrix} - \begin{bmatrix}\text{Flow of}\\ \text{mass out of}\\ \text{control volume}\end{bmatrix}$$

$$\left[\text{kg s}^{-1}\right] \qquad\qquad \left[\text{kg s}^{-1}\right] \qquad\qquad \left[\text{kg s}^{-1}\right]$$

. . . what comes in must come out (eventually).

Notes:

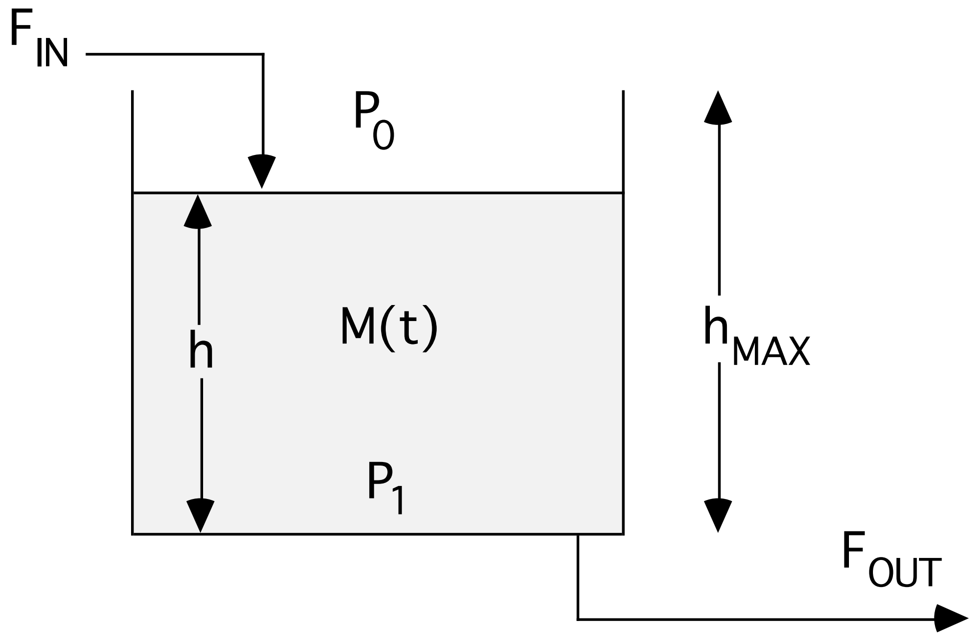
(a) units of each term must be consistent

(b) terms in balance equations are always extensive quantities.

[2] barring nuclear reactions and relativistic effects

MASS CONSERVATION — EXAMPLE

Buffer tank open to the atmosphere:



Control volume: liquid in tank — e.g., boundary is not rigid, so size and shape of control volume will change as the level rises and sinks

Assumptions:

- isothermal system => no need for energy balance
- single chemical species => fluid density constant
- inlet flow defined as an input (or by an upstream model) — e.g., $F_{IN}(t)$ known.
- vessel has uniform cross sectional area
- model valid for interval $0 \le h \le h_{MAX}$

MASS CONSERVATION — EXAMPLE

Variables (what quantities are we interested in?)**:**

M*	mass of fluid in vessel	[kg]
ρ	density of fluid in vessel	[kg m^{-3}]
V*	volume of fluid in vessel	[m^3]
F_{IN}	volumetric flow of inlet stream	[m^3s^{-1}]
F_{OUT}	volumetric flow of outlet stream	[m^3s^{-1}]
P_0	atmospheric pressure	[Nm^{-2}]
P_1	pressure at bottom of vessel	[Nm^{-2}]
A	cross sectional area of vessel	[m^2]
h*	liquid level in vessel	[m]

Note (*): in steady-state models we do not usually worry about quantities describing the "state" of the control volume — we are only concerned with the quantities crossing the boundary and the fact that they must balance at steady-state. This is a major reason why dynamic models are more complex — we now have to relate how this state changes to the quantities crossing the boundary. Clearly, the whole point of doing dynamic simulation is to determine how this state changes with time.

MASS CONSERVATION — EXAMPLE

Equations:

Mass Conservation

$$\frac{dM}{dt} = \rho F_{IN} - \rho F_{OUT} \tag{1}$$

Relate volume and mass

$$\rho V = M \tag{2}$$

Relate volume and liquid level (constant cross sectional area)

$$hA = V \tag{3}$$

Hydrostatic pressure

$$P_1 = P_0 + \rho g h \tag{4}$$

$g \equiv$ gravitational acceleration (9.81ms^{-2})

Flow pressure relationship — flow out driven by hydrostatic pressure in vessel

$$F_{OUT} = k\sqrt{P_1 - P_0} \tag{5}$$

$k \equiv$ loss coefficient (value known)

MASS CONSERVATION — EXAMPLE

Degree of freedom analysis:

Total number of quantities = 11

Time invariant quantities (model parameters):

$$A, \rho, k, g \quad (= 4)$$

Natural input set:

$$F_{IN}(t), P_0(t) \qquad (= 2)$$

Note: these functions are defined by the "environment" which could be either other models, or the engineer.

Remaining variables:

$$M, V, h, P_1, F_{OUT} \qquad (= 5)$$

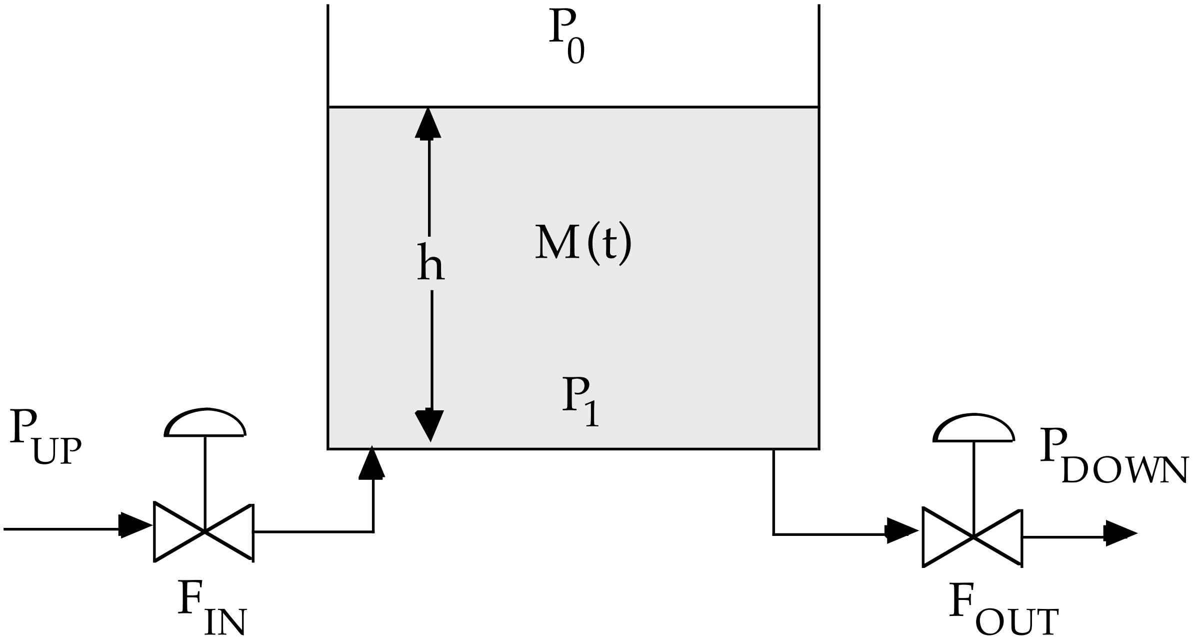
Equations = 5

=> degrees of freedom are satisfied given specification of input set above.

Note: due to the assumptions made above, equation (2) could be substituted into (1) to eliminate M - leading to a "volume balance." This is an error that is frequently made - volume is *not* a conserved quantity (e.g., consider a non isothermal and/or multi-component system) whereas mass is.

MASS CONSERVATION — EXERCISE

Develop a model for the following system:



Note:

(a) the inlet pipe has moved to a different physical location — consider how this changes the model.

(b) the control valves can be modelled by the following equation (see later):

$$F = C_v x \sqrt{\frac{P_{IN} - P_{OUT}}{\rho / \rho_w}}$$

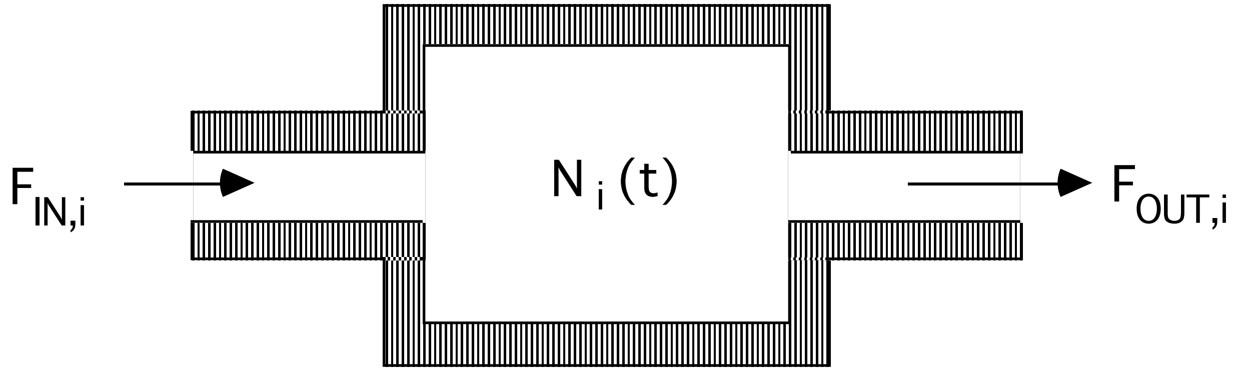
MASS CONSERVATION — EXERCISE

Variables:

F	= flow through valve
C_V	= valve sizing coefficient
x	= valve stem position (control)
P_{IN}	= inlet pressure to valve
P_{OUT}	= outlet pressure from valve
ρ	= density of fluid
ρ_W	= density of water at reference temperature

SPECIES BALANCES

Unlike mass, chemical species are *not* conserved: if a reaction takes place inside a control volume, reactants will be consumed and products generated.



A species balance can be written for each chemical species in the system (note we are now using moles rather than mass):

$$\underset{[\text{mol s}^{-1}]}{\begin{bmatrix}\text{Rate of}\\ \text{accumulation}\\ \text{of species i in}\\ \text{system}\end{bmatrix}} = \underset{[\text{mol s}^{-1}]}{\begin{bmatrix}\text{Flow of}\\ \text{species i}\\ \text{into the}\\ \text{system}\end{bmatrix}} - \underset{[\text{mol s}^{-1}]}{\begin{bmatrix}\text{Flow of}\\ \text{species i}\\ \text{out of the}\\ \text{system}\end{bmatrix}} + \underset{[\text{mol s}^{-1}]}{\begin{bmatrix}\text{Rate of}\\ \text{generation}\\ \text{of species i}\\ \text{by reaction}\end{bmatrix}}$$

$$\forall i = 1 \ldots NC$$

SPECIES BALANCES

Further, we can sum these NC[3] species balances to derive a total mole balance.

$$\begin{bmatrix}\text{Rate of}\\\text{accumulation}\\\text{of moles in}\\\text{system}\end{bmatrix}=\begin{bmatrix}\text{Flow of}\\\text{moles}\\\text{into}\\\text{the system}\end{bmatrix}-\begin{bmatrix}\text{Flow of}\\\text{moles out}\\\text{of the}\\\text{system}\end{bmatrix}+\begin{bmatrix}\text{Net rate}\\\text{of generation}\\\text{of moles by}\\\text{reaction}\end{bmatrix}$$

$$\left[\text{mol s}^{-1}\right] \qquad \left[\text{mol s}^{-1}\right] \qquad \left[\text{mol s}^{-1}\right] \qquad \left[\text{mol s}^{-1}\right]$$

Therefore, we can derive three different balance equations:

(a) NC species balances

(b) a total mole balance

(c) a mass balance

. . . clearly these are not independent — the total number of moles and the mass in the system can be related algebraically to the number of moles of each species, e.g.:

$$N_T = \sum_{i=1}^{NC} N_i$$

$$M = \sum_{i=1}^{NC} m_i N_i$$

m_i = molecular weight of species i [kg mol^{-1}]

[3] NC will be used to denote the number of chemical species in a system.

SPECIES BALANCES

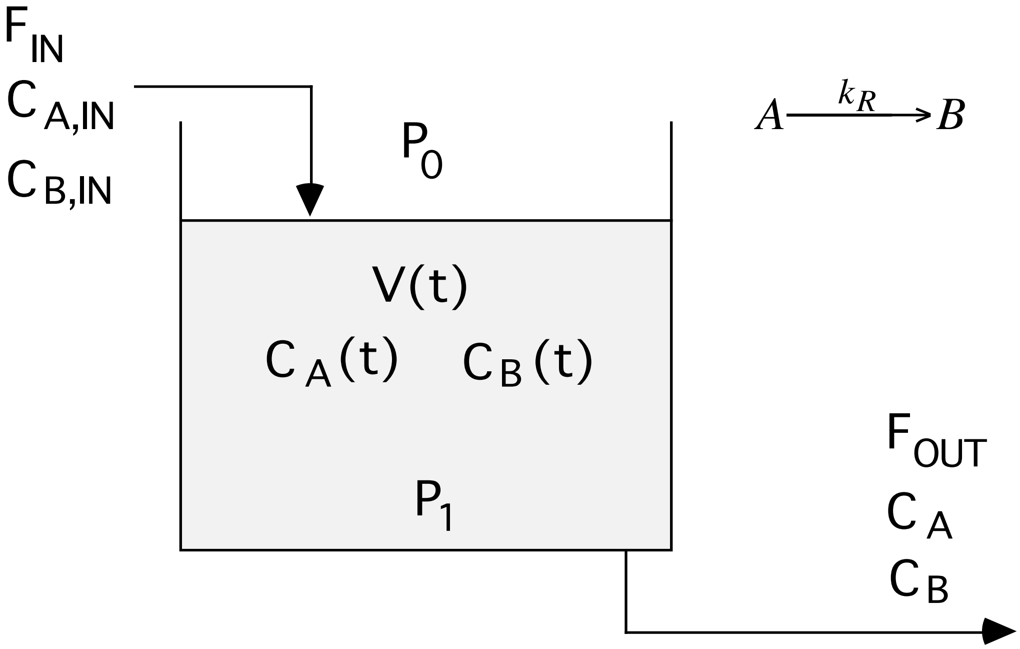
In general, it is best to derive a model in terms of the minimal number of independent species balances (usually NC) and derive other quantities via algebraic equations. The rationale behind this statement should become clearer when we discuss numerical solution of dynamic simulation problems.

SPECIES BALANCES — EXAMPLE

Consider the same tank as before, but the following liquid phase first order irreversible isomerization reaction takes place (i.e. isothermal CSTR):

$$A \xrightarrow{k_R} B$$

Species A and B are present in dilute solution (i.e. NC = 3; A, B, and the solvent).



Assumptions:

- vessel contents well mixed
- isothermal, species A and B present in dilute solution => fluid density constant (e.g., neglect density changes due to presence of A and B)
- model valid for interval $0 < h \leq h_{MAX}$ (note difference)
- otherwise, same as above

SPECIES BALANCES — EXAMPLE

Variables:

V	volume of fluid in vessel	$[m^3]$
ρ	density of fluid in vessel	$[kg\ m^{-3}]$
F_{IN}	volumetric flow of inlet stream	$[m^3\ s^{-1}]$
F_{OUT}	volumetric flow of outlet stream	$[m^3\ s^{-1}]$
P_0	atmospheric pressure	$[N\ m^{-2}]$
P_1	pressure at bottom of vessel	$[N\ m^{-2}]$
A	cross sectional area of vessel	$[m^2]$
h	liquid level in vessel	$[m]$
C_A	concentration of species A in vessel	$[mol\ m^{-3}]$
C_B	concentration of species B in vessel	$[mol\ m^{-3}]$
$C_{A,IN}$	concentration of species A in inlet stream	$[mol\ m^{-3}]$
$C_{B,IN}$	concentration of species B in inlet stream	$[mol\ m^{-3}]$
g	gravitational acceleration	$[m\ s^{-2}]$
k	loss coefficient	
k_R	reaction rate constant	$[s^{-1}]$
r	reaction rate	$[mol\ m^{-3}\ s^{-1}]$

SPECIES BALANCES — EXAMPLE

Equations:

Mass Conservation

$$\frac{dV}{dt} = F_{IN} - F_{OUT} \tag{1}$$

Species balances for A and B

$$\frac{d(VC_A)}{dt} = C_A \frac{dV}{dt} + V \frac{dC_A}{dt} = F_{IN} C_{A,IN} - F_{OUT} C_A - Vr \tag{2}$$

$$\frac{d(VC_B)}{dt} = C_B \frac{dV}{dt} + V \frac{dC_B}{dt} = F_{IN} C_{B,IN} - F_{OUT} C_B + Vr \tag{3}$$

Notes:

- use of volume in mass balance is *only* valid for the assumptions above
- concentrations in outlet stream equal to bulk concentration because vessel contents well mixed
- we have derived NC (=3) mass and species balances — this is sufficient to define the state of the system: all other quantities can be related to {V, C_A,C_B} via algebraic relationships. Note that an alternative would have been to derive a species balance for the solvent instead of the mass balance.

SPECIES BALANCES — EXAMPLE

Relate volume and liquid level

$$hA = V \tag{4}$$

Hydrostatic pressure

$$P_1 = P_0 + \rho g h \tag{5}$$

Flow pressure relationship

$$F_{OUT} = k\sqrt{P_1 - P_0} \tag{6}$$

Reaction rate

$$r = k_R C_A \tag{7}$$

. . . and additional equations to define $N_A \ (= C_A V)$, $N_B \ (= C_B V)$, $M \ (= \rho V)$ if desired.

Note: if significant density changes occur due to reaction (e.g., not sufficiently dilute) then it is better to derive NC species balances and derive the volume from the molar volumes in solution.

SPECIES BALANCES — EXAMPLE

Degree of freedom analysis:

Total number of quantities = 16

Time invariant parameters:

$$A, \rho, g, k, k_R \qquad (= 5)$$

Natural input set:

$$F_{IN}(t), C_{A,IN}(t), C_{B,IN}(t), P_0(t) \qquad (=4)$$

Remaining variables:

$$V, F_{OUT}, P_1, h, C_A, C_B, r \qquad (=7)$$

Equations = 7

=> degrees of freedom satisfied given specification of input set above

ENERGY CONSERVATION

According to the First Law of Thermodynamics, *energy* is a conserved quantity. For an open system, this can be expressed as:

$$\underset{[\mathrm{J\ s^{-1}}]}{\begin{bmatrix}\text{rate of}\\ \text{accumulation}\\ \text{of energy in}\\ \text{system}\end{bmatrix}} = \underset{[\mathrm{J\ s^{-1}}]}{\begin{bmatrix}\text{rate of addition}\\ \text{of heat from the}\\ \text{environment}\end{bmatrix}} + \underset{[\mathrm{J\ s^{-1}}]}{\begin{bmatrix}\text{rate of work}\\ \text{done on the}\\ \text{system by the}\\ \text{environment}\end{bmatrix}} +$$

$$\underset{[\mathrm{J\ s^{-1}}]}{\begin{bmatrix}\text{rate of energy}\\ \text{addition to system}\\ \text{by material flow}\\ \text{into system}\end{bmatrix}} - \underset{[\mathrm{J\ s^{-1}}]}{\begin{bmatrix}\text{rate of energy discharge}\\ \text{from the system by}\\ \text{material flow out of the}\\ \text{system}\end{bmatrix}}$$

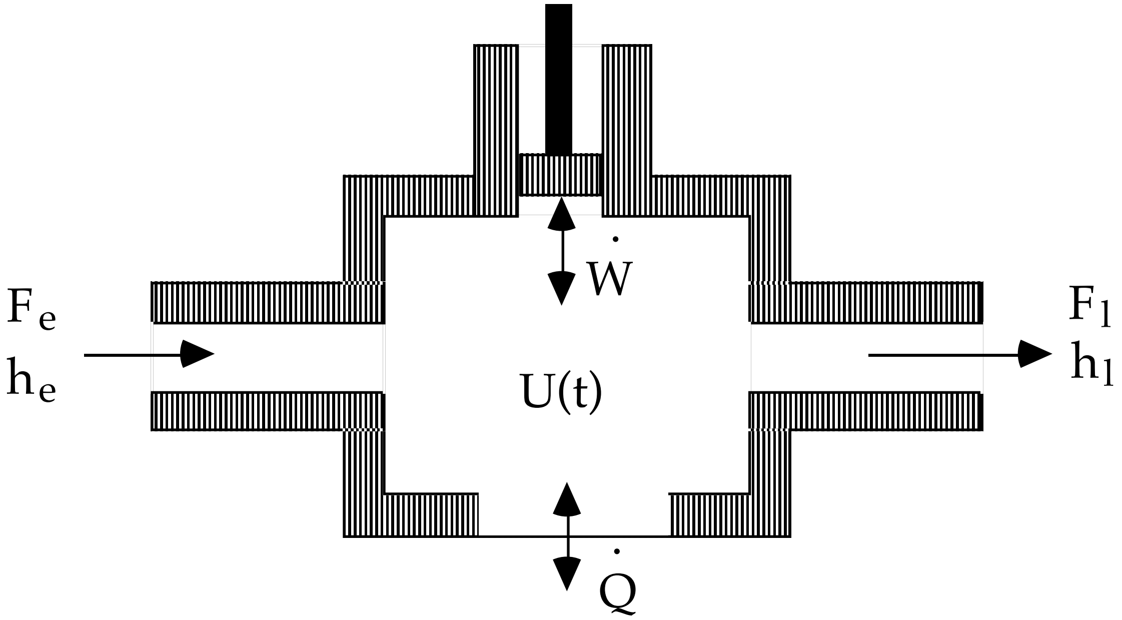
Warning: always start from the First Law when deriving energy balances!!

Here, energy is the summation of internal energy (e.g., that associated with translation, rotation and vibration of molecules), kinetic energy, and potential energy.

In most process simulation applications, it is usually reasonable to neglect kinetic and potential energy (or to perform separate balances for internal energy and these other forms of energy) — but always check this assumption.

ENERGY CONSERVATION

Given this assumption, the differential form of the First Law of Thermodynamics for the following "general" open system (see Modell and Reid (1983); pp. 39-41 for a complete derivation) reduces to:



$$\frac{dU}{dt} = \dot{Q} + \dot{W} + \sum_e F_e h_e - \sum_l F_l h_l$$

...where we have introduced time as the independent variable.

Note that the conserved quantity is the *internal energy of the control volume contents* (not the enthalpy!). The work term is composed of two contributions:

$$\dot{W} = \dot{W}_s - P\frac{dV}{dt}$$

...the summation of the shaft work done on the system (e.g., an impeller to keep the contents well mixed) and the PV work due to changes in volume of the system.

ENERGY CONSERVATION

Variables:

U	internal energy of control volume contents	[J]
$\dot{Q}$	rate of heat addition to control volume from environment	[J s^{-1}]
$\dot{W}$	rate of work done on the control volume by the environment	[J s^{-1}]
F_i	rate of material addition/discharge	[kg s^{-1}] or [mol s^{-1}]
h_i	specific enthalpy of material stream	[J kg^{-1}] or [J mol^{-1}]

ENERGY CONSERVATION

The energy balance can therefore be expressed in two equivalent forms:

$$\frac{dU}{dt}=\dot{Q}+\dot{W}_s-P\frac{dV}{dt}+\sum_e F_e h_e-\sum_l F_l h_l \qquad \text{(E1)}$$

or dU/dt can be eliminated by substitution of the differential form of the definition of enthalpy ($H=U+PV$):

$$\frac{dH}{dt}=\frac{dU}{dt}+P\frac{dV}{dt}+V\frac{dP}{dt}$$

which leads to:

$$\frac{dH}{dt}=\dot{Q}+\dot{W}_s+V\frac{dP}{dt}+\sum_e F_e h_e-\sum_l F_l h_l \qquad \text{(E2)}$$

where,

$H\equiv$ enthalpy of the control volume contents [J]

Each form is convenient under certain assumptions. If the volume of the system is constant, energy balance (E1) reduces to:

$$\frac{dU}{dt}=\dot{Q}+\dot{W}_s+\sum F_e h_e-\sum F_l h_l$$

and if the pressure of the system is constant, energy balance (E2) reduces to:

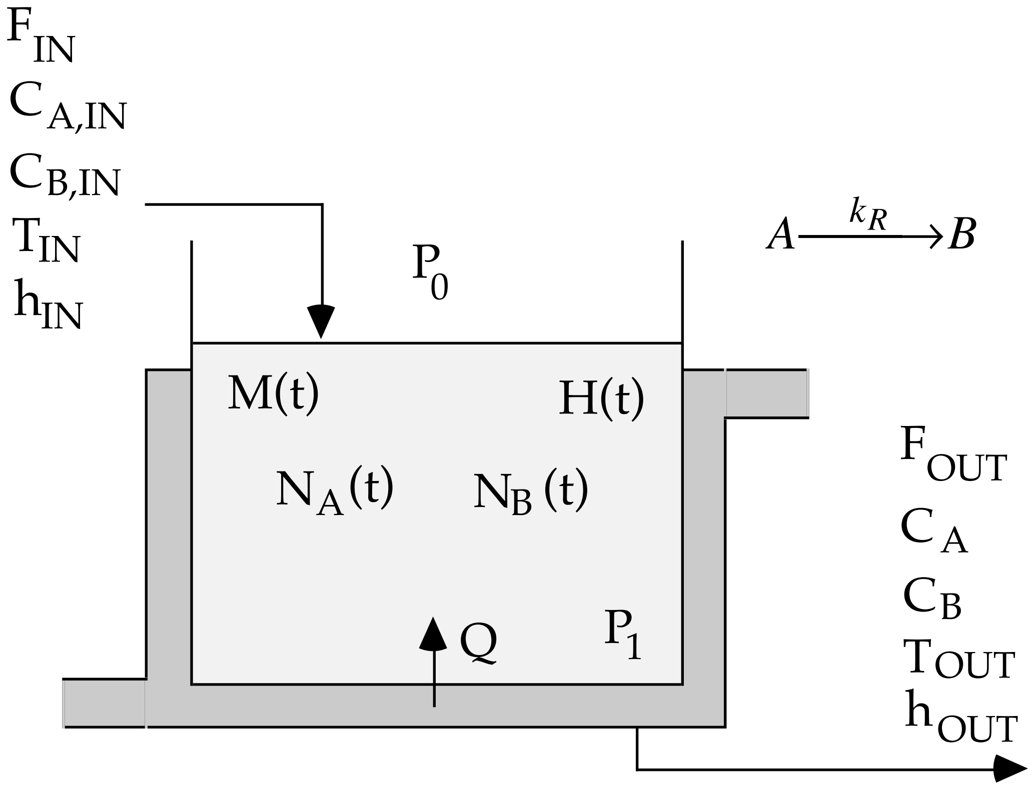
$$\frac{dH}{dt}=\dot{Q}+\dot{W}_s+\sum_e F_e h_e-\sum_l F_l h_l$$

ENERGY CONSERVATION

Of course, there are many situations in which neither the volume or the pressure remains constant over the time horizon of interest. In this case, we must be able to express the rate of change of the volume or pressure explicitly, or have this implied by the algebraic equations (which leads to difficulties - see discussion of "high index" problems later).

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Consider the CSTR again — the reaction is now exothermic and heat is removed by a jacket containing a vaporizing medium, so an energy balance is necessary.



Assumptions:

- vessel contents well mixed
- species A and B present in dilute solution
- atmospheric pressure is constant (e.g., $P_0 \neq f(t)$)
- neglect shaft work of impeller
- neglect heat interaction with atmosphere
- otherwise, same as above.

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Variables:

M	mass of fluid in vessel	[kg]
V	volume of fluid in vessel	[m^{3}]
N_A	number of moles of species A in vessel	[mol]
N_B	number of moles of species B in vessel	[mol]
C_A	concentration of species A in vessel	[mol m^{-3}]
C_B	concentration of species B in vessel	[mol m^{-3}]
ρ	density of fluid in vessel	[kg m^{-3}]
A	cross sectional area of vessel	[m^{2}]
A_J	heat transfer area of jacket	[m^{2}]
U_J	overall heat transfer coefficient for jacket	[J m^{-2} K^{-1} s^{-1}]
H	enthalpy of vessel contents	[J]
$C_{A,IN}$	concentration of species A in inlet stream	[mol m^{-3}]
$C_{B,IN}$	concentration of species B in inlet stream	[mol m^{-3}]
F_{IN}	volumetric flow of inlet stream	[m^{3} s^{-1}]
F_{OUT}	volumetric flow of outlet stream	[m^{3} s^{-1}]
T_{IN}	temperature of inlet stream	[K]
T_{OUT}	temperature of outlet stream	[K]
P_0	atmospheric pressure	[Nm^{-2}]
P_1	pressure at bottom of vessel	[Nm^{-2}]

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

g	gravitational acceleration	[m s^{-1}]
k	loss co-efficient	
k_R	reaction rate constant	[s^{-1}]
r	reaction rate	[mol m^{-3} s^{-1}]
ρ_{IN}	density of inlet stream	[kg m^{-3}]
ρ_{OUT}	density of outlet stream	[kg m^{-3}]
$\dot{Q}$	heat transfer to fluid in vessel from jacket	[J s^{-1}]
h_{IN}	specific enthalpy of inlet stream	[J kg^{-1}]
h_{OUT}	specific enthalpy of outlet stream	[J kg^{-1}]
ΔE_R	activation energy	[J mol^{-1}]
R	universal gas constant	[J mol^{-1} K^{-1}]
h	liquid level in vessel	[m]
T	temperature of vessel contents	[K]
T_J	temperature of vaporizing medium	[K]

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Equations:

Mass conservation

$$\frac{dM}{dt} = \rho_{IN} F_{IN} - \rho_{OUT} F_{OUT} \tag{1}$$

...note we are unable to assume ρ is constant

Species balances for A and B

$$\frac{dN_A}{dt} = F_{IN} C_{A,IN} - F_{OUT} C_A - Vr \tag{2}$$

$$\frac{dN_B}{dt} = F_{IN} C_{B,IN} - F_{OUT} C_B + Vr \tag{3}$$

Energy balance (constant pressure formulation — e.g., it is most convenient to use enthalpy in the accumulation term)

$$\frac{dH}{dt} = \dot{Q} + F_{IN} \rho_{IN} h_{IN} - F_{OUT} \rho_{OUT} h_{OUT} \tag{4}$$

Reaction rate

$$r = k_R e^{-\Delta E_R / RT} C_A \tag{5}$$

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Relate volume and mass

$$\rho V = M \tag{6}$$

Relate volume and liquid level

$$hA = V \tag{7}$$

Relate mole numbers and concentration

$$N_A = VC_A \tag{8}$$
$$N_B = VC_B \tag{9}$$

Hydrostatic pressure

$$P_1 = P_0 + \rho g h \tag{10}$$

Flow pressure relationship

$$F_{OUT} = k\sqrt{P_1 - P_0} \tag{11}$$

Contents well mixed

$$\rho = \rho_{OUT} \tag{12}$$
$$T = T_{OUT} \tag{13}$$

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Define enthalpy holdup (implies temperature of contents)

$$H = Mh_{OUT} \tag{14}$$

Physical Properties (abstract functions)

$$\rho = \rho(T) \tag{15}$$

$$\rho_{IN} = \rho(T_{IN}) \tag{16}$$

$$h_{IN} = h(T_{IN}, C_{A,IN}, C_{B,IN}) \tag{17}$$

$$h_{OUT} = h(T, C_A, C_B) \tag{18}$$

Heat transfer

$$\dot{Q} = U_J A_J (T_J - T) \tag{19}$$

A commonly asked question is: how is the temperature in the reactor determined? In fact, the temperature is determined by the simultaneous solution of the complete set of implicit relationships above. One can view equation (4) as determining the extensive enthalpy *H* of the vessel contents, equation (14) determining the intensive enthalpy h_{OUT}, and equation (18) implicitly determining *T* given h_{OUT}.

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Degree of freedom analysis:

Total number quantities = 33

Time invariant parameters:

$$A, A_J, U_J, g, K, k_R, \Delta E_R R, P_0 \qquad (=9)$$

Natural input set:

$$F_{IN}(t), C_{A,IN}(t), C_{B,IN}(t), T_{IN}(t), T_J(t) \qquad (=5)$$

Remaining variables:

$$\begin{gathered} M, V, N_A, N_B, C_A, C_B, \rho, H, F_{OUT}, T_{OUT}, \\ P_1, r, \rho_{IN}, \rho_{OUT}, \dot{Q}, h_{IN}, h_{OUT}, h, T \end{gathered} \qquad (=19)$$

Equations = 19

=> degrees of freedom satisfied given specification of input set above.

ENERGY BALANCE — CONSTANT PRESSURE EXAMPLE

Note: in many textbooks it is common practice to include a reaction term in the energy balance. The control volume approach (equation (E1) or (E2)) clearly shows that no energy crosses the system boundary due to a reaction taking place, so a reaction term should not appear in the energy balance.

If a reaction is taking place in an isolated system, the total energy of the system remains unchanged, but the distribution of energy between "heat of formation" energy and "sensible heat" energy changes as the reaction progresses (e.g., the temperature will rise or drop if the reaction is exothermic or endothermic).

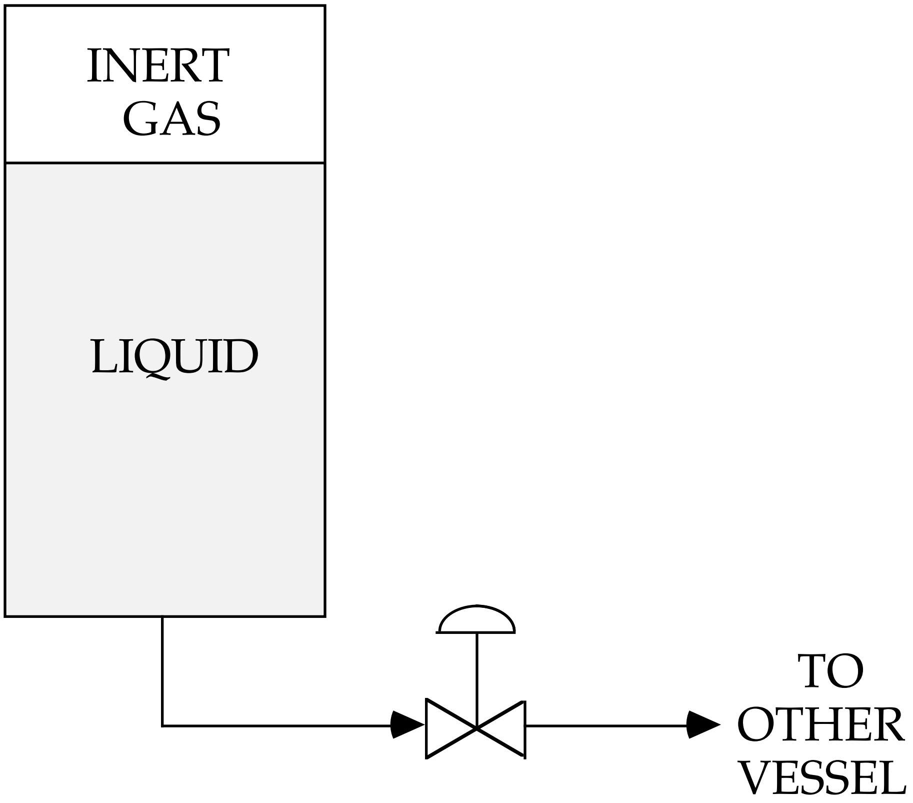
To reflect this, the constitutive equation defining the specific enthalpy (18) must include the contribution of both the heat of formation and the sensible heat of each species to the total system energy. So, the zero energy reference state for equation (18) must be defined as the elements making up the chemical species in their standard states at some temperature and pressure.

If heats of formation are not included in equation (18), a heat of reaction term must be added to the energy balance, and the heat of reaction must be calculated as a function of temperature.

Overall, I consider it clearer and simpler to work with the first law and include heats of formation in species enthalpies (obviously, none of this is necessary if no chemical reactions occur in the system of interest).

ENERGY BALANCE — EXAMPLE OF SYSTEM IN WHICH PRESSURE AND VOLUME VARY

Consider a system in which an inert gas at high pressure is employed to charge liquid rapidly to another vessel:



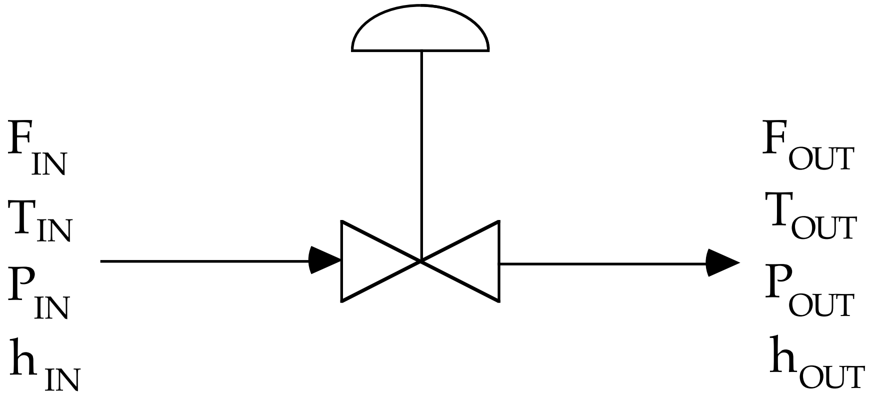
System decomposition: system is composed of two components:

(a) the storage vessel
(b) the control valve

... we must develop models for each.

MODEL OF CONTROL VALVE

For a more complete discussion see Luyben (1990), pp. 213-222:



Assumptions:

- fluid density constant
- irreversible adiabatic expansion
- rate of accumulation of material and energy in piping negligible in comparison to material and energy flows => static (pseudo steady-state) mass and energy balances
- value has linear inherent characteristic
- non flashing liquid phase.
- no significant time lag of stem position in response to changes in the control signal

Control valve dynamics are particularly important in control related studies (see Norsk Hydro example).

MODEL OF CONTROL VALVE

Variables:

F_{IN}	volumetric flow into valve	[m^3s^{-1}]
C_V	valve sizing coefficient	
ρ	density of liquid	[kg m^{-3}]
ρ_w	density of water at reference temperature	[kg m^{-3}]
x	valve stem position	
F_{OUT}	volumetric flow out of valve	[m^3s^{-1}]
T_{IN}	temperature of inlet stream	[K]
T_{OUT}	temperature of outlet stream	[K]
h_{IN}	specific enthalpy of inlet stream	[J kg^{-1}]
h_{OUT}	specific enthalpy of outlet stream	[J kg^{-1}]
P_{IN}	pressure of inlet stream	[N m^{-2}]
P_{OUT}	pressure of inlet stream	[N m^{-2}]

MODEL OF CONTROL VALVE

Equations:

Mass conservation (static - constant fluid density)

$$F_{IN} = F_{OUT} \tag{V1}$$

Energy conservation (static - irreversible adiabatic expansion)

$$F_{IN}\rho h_{IN} = F_{OUT}\rho h_{OUT} \tag{V2}$$

Flow pressure relationship (linear characteristic)

$$F_{IN} = C_V x \sqrt{\frac{P_{IN} - P_{OUT}}{\rho / \rho_w}} \tag{V3}$$

Physical properties (assume h_{IN}, T_{IN} calculated from equations in upstream unit)

$$h_{OUT} = h(T_{OUT}) \tag{V4}$$

MODEL OF CONTROL VALVE

Degree of freedom analysis:

Total number of quantities = 12

Time invariant parameters:

$$C_V ,\rho ,\rho_w \text{ (=3)}$$

Natural input set:

$$x(t), T_{IN}(t), h_{IN}(t), P_{IN}(t), P_{OUT}(t) \qquad \text{(=5)}$$

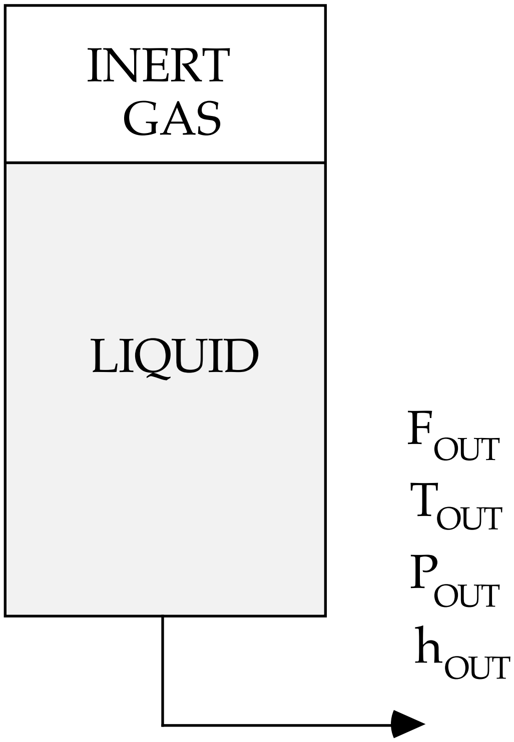
Remaining variables:

$$F_{IN} ,F_{OUT} ,T_{OUT} ,h_{OUT} \qquad \text{(=4)}$$

Equations = 4

=> degrees of freedom satisfied for input set above.

MODEL OF STORAGE VESSEL



Control Volumes: control volume for gas phase, *and* control volume for liquid phase. Both will vary in size and shape as liquid is ejected.

Assumptions:

- liquid density constant
- ideal gas equation of state
- adiabatic vessel (in time horizon of interest)
- negligible heat and mass transfer between liquid and inert gas (in time horizon of interest)
- uniform cross sectional area
- liquid phase well mixed and gas phase well mixed
- valid in interval $0 < h < h_{max}$

MODEL OF STORAGE VESSEL

Variables:

M^V	mass of gas in vessel	[kg]
M^L	mass of liquid in vessel	[kg]
ρ	density of liquid in vessel	[kg m^{-3}]
F_{OUT}	volumetric flow of liquid from vessel	[m^{3}s^{-1}]
U^V	internal energy of gas phase	[J]
U^L	internal energy of liquid phase	[J]
P	pressure of gas	[Nm^{-2}]
V^V	volume of gas phase	[m^{3}]
V^L	volume of liquid phase	[m^{3}]
h_{OUT}	specific enthalpy of outlet stream	[J kg^{-1}]
MW	molecular weight of gas	[kg mol^{-1}]
R	universal gas constant	[J mol^{-1}k^{-1}]
T^V	temperature of gas	[K]
h	liquid level in vessel	[m]
A	cross sectional area of vessel	[m^{2}]
P_{OUT}	pressure of outlet stream	[Nm^{-2}]
g	gravitational acceleration	[ms^{-2}]
V_{TOT}	vessel volume	[m^{3}]
T^L	temperature of liquid	[K]
T_{OUT}	temperature of outlet stream	[K]
h^L	specific enthalpy of liquid phase	[J kg^{-1}]
h^V	specific enthalpy of gas phase	[J kg^{-1}]

MODEL OF STORAGE VESSEL

Equations

Mass balance on gas phase (mass of gas remains constant)

$$\frac{dM^V}{dt} = 0 \tag{S1}$$

Mass balance on liquid phase

$$\frac{dM^L}{dt} = -\rho F_{OUT} \tag{S2}$$

Energy balance on gas phase (neither pressure or volume constant)

$$\frac{dU^V}{dt} = -P\frac{dV^V}{dt} \tag{S3}$$

Energy balance on liquid phase

$$\frac{dU^L}{dt} = -P\frac{dV^L}{dt} - F_{OUT}\rho h_{OUT} \tag{S4}$$

Relate mass and volumes of each phase

$$\rho V^L = M^L \tag{S5}$$

Ideal gas equation of state

$$PV^V = \frac{M^V}{MW}RT^V \tag{S6}$$

MODEL OF STORAGE VESSEL

Relate liquid volume and liquid level

$$hA = V^L \tag{S7}$$

Hydrostatic pressure

$$P_{OUT} = P + \rho g h \tag{S8}$$

Volume constraint

$$V_{TOT} = V^L + V^V \tag{S9}$$

Well mixed liquid

$$T_{OUT} = T^L \tag{S10}$$

$$h_{OUT} = h^L \tag{S11}$$

Define internal energy holdups (implies temperatures of phases)

$$U^V + PV^V = M^V h^V \tag{S12}$$
$$U^L + PV^L = M^L h^L \tag{S13}$$

Physical properties (abstract functions)

$$h^V = h^V(T^V, P) \tag{S14}$$
$$h^L = h^L(T^L, P) \tag{S15}$$

MODEL OF STORAGE VESSEL

Degree of freedom analysis:

Total number of quantities = 22

Time invariant parameters:

$$\rho, MW, R, A, g, V_{TOT} \qquad (=6)$$

Natural input set:

$$F_{OUT}(t) \qquad (=1)$$

Remaining variables:

$$\begin{matrix} M^V, M^L, U^V, U^L, P, V^V, V^L, h_{OUT}, \\ T^V, T^L, h, P_{OUT}, T_{OUT}, h^L, h^V \end{matrix} \quad (=15)$$

Equations = 15

=> degrees of freedom satisfied for input set above.

PUTTING IT ALL TOGETHER — FORMULATING A SYSTEM MODEL

We now have models for each component in our system — these must be connected to form a *system* model.

Create *instances* of the storage vessel and control valve models, and connect the outlet stream of the storage vessel to the inlet stream of the control valve. This adds the following equations to the system model:

$$\begin{aligned} Vessel.F_{OUT} &= Valve.F_{IN} \\ Vessel.T_{OUT} &= Valve.T_{IN} \\ Vessel.P_{OUT} &= Valve.P_{IN} \\ Vessel.h_{OUT} &= Valve.h_{IN} \end{aligned} \qquad \text{(C1-4)}$$

Note the use of a *pathname* mechanism:

```
unit_name.variable_name
```

to identify uniquely variables in different models (hence the same variable name may be used in many different models).

Degree of freedom analysis for system model:

Number of Equations = 23 (V1-4, S1-15, C1-4)

Total number of quantities = 34

Number of time invariant parameters = 9

=> Degrees of freedom = 34 - 9 - 23=2

PUTTING IT ALL TOGETHER — FORMULATING A SYSTEM MODEL

So, what simulation could we specify?

- values for all the time invariant quantities (geometry, physical characteristics of system)
- specify the pressure of the downstream vessel ($Valve.P_{OUT}(t)$) and the stem position of the control valve ($Valve.x(t)$)

=> we can solve the system model for the time variation of all the other quantities — e.g., open the valve and determine the system behaviour as liquid is ejected by the high pressure gas.

Notes:

(a) although the material flow in this system is unidirectional from the storage vessel to the valve (assuming $Vessel.P > Valve.P_{OUT}$), the information flow in the model is not: we specify the downstream pressure and calculate the upstream flow rate. This means that the models of the valve and the storage vessel must be solved simultaneously — in general, the decomposition techniques applied to solve steady-state simulations cannot be applied to dynamic simulations even when material flow in the flowsheet is unidirectional.

(b) the introduction of the PV work terms in the energy balances causes problems with consistent initialization of this model (see discussion of "high index" problems later on).

PUTTING IT ALL TOGETHER — FORMULATING A SYSTEM MODEL

Exercise: what is the qualitative time variation of the temperature of the gas phase if the liquid is ejected rapidly from the vessel:

(a) predicted by the model above

(b) predicted by a model in which we neglect the PV work terms in the energy balances.

...bearing this qualitative behaviour in mind, and the fact that the vessel is constructed from carbon steel:

(i) what safety concerns would you have about this operation if we started it with the gas at room temperature?

(ii) in conducting safety studies, which model above ((a) or (b)) would you recommend?

MOMENTUM BALANCE

It is sometimes necessary to model the velocity (or momentum) of the contents of a control volume. As flows can in general be three dimensional, velocity is a vector quantity with components corresponding to the velocity resolved into the coordinate directions of the chosen coordinate system. So, in principle we can formulate a momentum balance for each co-ordinate direction (three balances).

Applying Newton's Second Law to a control volume, we obtain:

$$\underset{\left[\text{kg ms}^{-2}\right]}{\begin{bmatrix}\text{Rate of accumulation}\\ \text{of momentum in the}\\ \text{ith direction in control}\\ \text{volume}\end{bmatrix}} = \underset{\left[\text{kg ms}^{-2}\right]}{\begin{bmatrix}\text{Rate at which}\\ \text{momentum in}\\ \text{ith direction}\\ \text{flows into}\\ \text{control volume}\end{bmatrix}} - \underset{\left[\text{kg ms}^{-2}\right]}{\begin{bmatrix}\text{Rate at which}\\ \text{momentum in}\\ \text{the ith direction}\\ \text{flows out of}\\ \text{control volume}\end{bmatrix}}$$

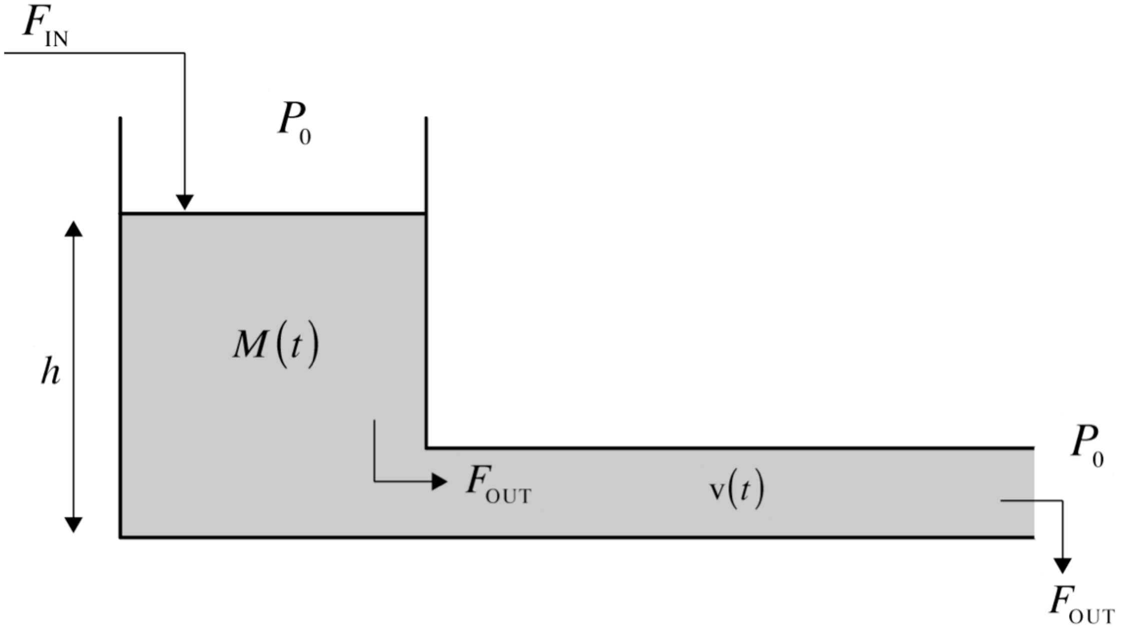
$$+ \underset{[\text{N}] \ \equiv \ [\text{kg ms}^{-2}]}{\begin{bmatrix}\text{Sum of forces}\\ \text{applied to}\\ \text{control volume}\\ \text{in ith direction}\end{bmatrix}}$$

. . . for each direction i in which material is flowing.

Note: momentum is defined as the product of mass and velocity. Care should be taken if both the mass and the velocity of the control volume are changing with respect to time.

MOMENTUM BALANCE EXAMPLE

Consider the buffer tank open to the atmosphere, but now the fluid flows out into a long pipeline:



Control volumes: (a) liquid in tank, (b) liquid in pipeline

Assumptions:

- same as for original buffer tank example
- one dimensional plug flow in pipeline and incompressible liquid => velocity uniform throughout pipeline (macroscopic control volume)

MOMENTUM BALANCE EXAMPLE

Variables:

M	mass of fluid in vessel	[kg]
ρ	density of fluid in vessel	[kg m^{-3}]
F_{IN}	volumetric flow of inlet stream	[m^{3}s^{-1}]
F_{OUT}	volumetric flow of inlet stream	[m^{3}s^{-1}]
A_P	cross sectional area of pipeline	[m^2]
L	length of pipeline	[m]
v	velocity of fluid in pipeline (uniform)	[ms^{-1}]
F_H	hydraulic force on fluid	[N]
F_F	frictional force resisting flow	[N]
g	gravitational acceleration	[ms^{-2}]
h	level of liquid in vessel	[m]
V	volume of liquid in vessel	[m^3]
k_F	constant related to Fanning friction factor	
A	cross sectional area of vessel	[m^2]

MOMENTUM BALANCE EXAMPLE

Equations:

Mass conservation in tank

$$\frac{dM}{dt} = \rho F_{IN} - \rho F_{OUT} \tag{1}$$

. . . mass balance on pipeline unnecessary - incompressible liquid in fixed volume (Flow in = Flow out)

Momentum conservation on pipeline - axial direction only

$$\frac{d(A_p L \rho v)}{dt} = \rho F_{OUT} v - \rho F_{OUT} v + F_H - F_F$$

$$\Rightarrow A_P L \rho \frac{dv}{dt} = F_H - F_F \tag{2}$$

Relate velocity and volumetric flow

$$A_P v = F_{OUT} \tag{3}$$

Hydraulic force

$$F_H = A_P \rho g h \tag{4}$$

Frictional force (proportional to the square of the velocity and the length of the pipe — large Reynolds numbers)

$$F_F = k_F L v^2 \tag{5}$$

MOMENTUM BALANCE EXAMPLE

Relate volume and mass

$$\rho V = M \tag{6}$$

Relate volume and liquid level

$$hA = V \tag{7}$$

Degree of freedom analysis:

Total number of quantities = 14

Time invariant parameters:

$$\rho, A_p, L, g, k_F, A \qquad (=6)$$

Natural input set:

$$F_{IN}(t) \qquad (=1)$$

Remaining variables:

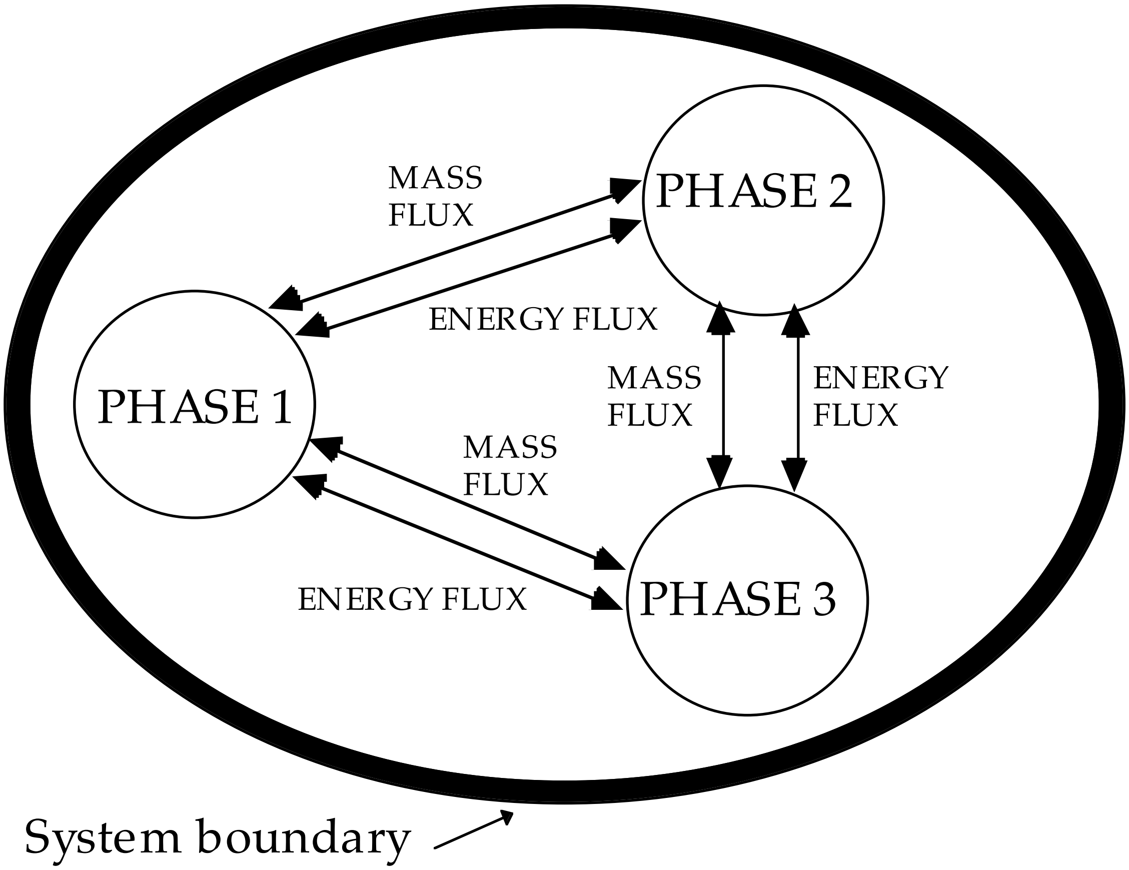
$$M, F_{OUT}, v, F_H, F_F, h, V \qquad (=7)$$

Equations = 7

=> degrees of freedom satisfied

MULTI-PHASE SYSTEMS

Many systems encountered in chemical processes can be abstracted as multiple phases that exchange material and energy, for example:



It is normally assumed that intensive properties inside each phase are uniform (so macroscopic balances can be performed on each phase).

Together, the phases form a simple thermodynamic system that may be constrained at a known volume or pressure.

MULTI-PHASE SYSTEMS

Clearly there are two assumptions we can make about the conditions in this system:

(a) the phases are in phase equilibrium with each other — leading to "equilibrium" models.

(b) the phases are not in phase equilibrium — leading to "non equilibrium" models.

. . . we will consider both cases.

EQUILIBRIUM SYSTEMS

When modelling a system composed of multiple phases at phase equilibrium, the following general comments can be made:

(a) typically, we are assuming that phase equilibrium is reached on a much faster time scale than the other transients of interest.

(b) in general, the conditions for phase equilibrium for multi-component (NC species) simple system composed of π phases can be expressed as:

$$T^1 = T^2 = \ldots = T^\pi$$
$$P^1 = P^2 = \ldots = P^\pi$$
$$\mu_i^1 = \mu_i^2 = \ldots = \mu_i^\pi \qquad \forall i = 1 \ldots NC$$

where:

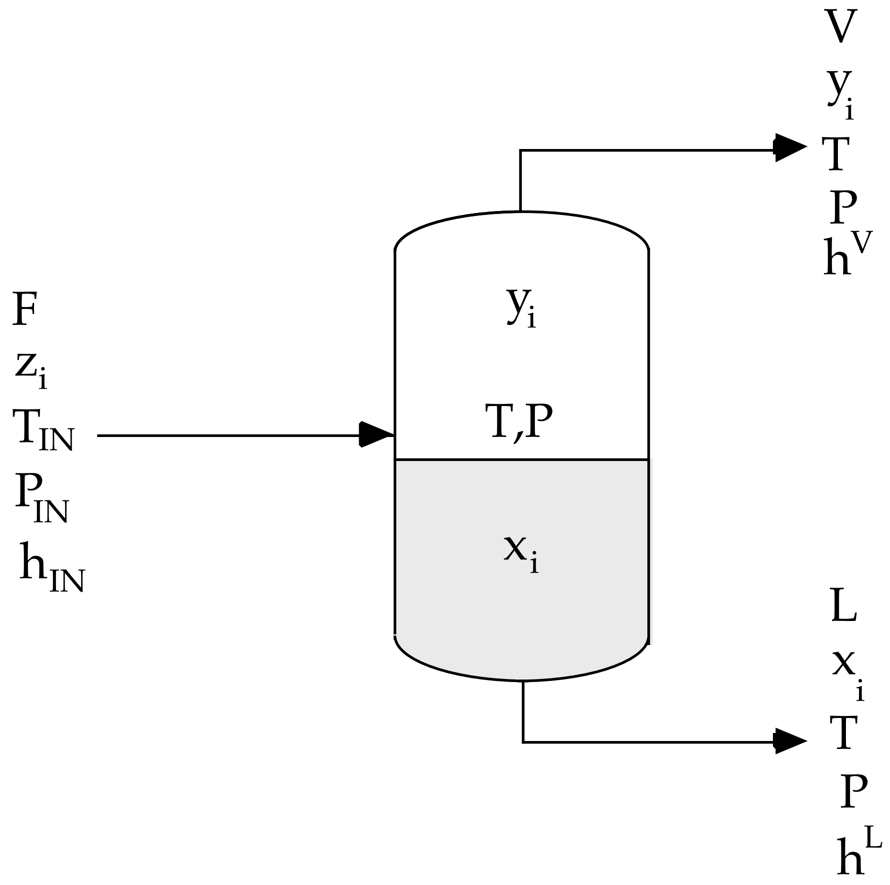
T^j = temperature in phase j
P^j = pressure in phase j
μ_i^j = chemical potential of species i in phase j

In other words, the condition of phase equilibrium enforces $(\pi - 1)(NC + 2)$ equations in the system model.

(c) when formulating species and energy balances consider a control volume encompassing all the phases in equilibrium. The reasons for this relate to the *index* of the resulting model (see Ponton and Gawthrop (1991) for a full explanation), but the consequence is that we cannot calculate the mass and energy fluxes between the phases explicitly.

EQUILIBRIUM SYSTEM — FLASH DRUM

Consider a model of a flash drum in which it is assumed there are vapour and liquid phases in phase equilibrium with each other at all times:



Control volume: the entire vessel contents (e.g. encompassing both phases). Phases constrained to occupy volume of the vessel.

EQUILIBRIUM SYSTEM — FLASH DRUM

Assumptions:

- phases at phase equilibrium
- individual phases well mixed
- vessel adiabatic (no particular need for this — just makes model development easier)
- both phases present at all times — e.g. model valid while $T_{BUBBLE} \leq T \leq T_{DEW}$
- bulk flows into and out of vessel specified or determined by upstream and downstream models.

Variables:

N_i	number of moles of species i in vessel	[mol]
F	total molar flow to vessel	[mol s^{-1}]
L	total liquid flow from vessel	[mol s^{-1}]
V	total vapour flow from vessel	[mol s^{-1}]
z_i	mole fraction of component i in feed stream	
x_i	mole fraction of component i in liquid phase	
y_i	mole fraction of component i in vapour phase	
U	internal energy of vessel contents	[J]
h_{IN}	molar enthalpy of feed stream	[J mol^{-1}]
h^V	molar enthalpy of vapour phase	[J mol^{-1}]
h^L	molar enthalpy of liquid phase	[J mol^{-1}]
N^V	total number of moles in vapour phase	[mol]

EQUILIBRIUM SYSTEM — FLASH DRUM

N^L	total number of moles in liquid phase	[mol]
P	system pressure (uniform)	[Nm^{-2}]
T	system temperature (uniform)	[K]
V_{TOT}	vessel volume	[m^3]
v^V	molar volume of vapour phase	[m^{3}mol^{-1}]
v^L	molar volume of liquid phase	[m^{3}mol^{-1}]
P_{IN}	pressure of inlet stream	[Nm^{-2}]
T_{IN}	temperature of inlet stream	[K]

EQUILIBRIUM SYSTEM — FLASH DRUM

Equations:

Species balances

$$\frac{dN_i}{dt} = Fz_i - Lx_i - Vy_i \qquad \forall i = 1 \ldots NC \tag{1}$$

Energy balance (constant volume formulation)

$$\frac{dU}{dt} = Fh_{IN} - Vh^V \;\; - Lh^L \tag{2}$$

Phase equilibrium (the temperature and pressures of each phase have already been eliminated) can be expressed in several different ways (each of which is a specialization of the general reactions given above):

(a) single chemical species

$$P = P^{SAT}(T)$$

(b) ideal-gas and ideal-solution in liquid phase (Raoult's Law)

$$y_i P = x_i P_i^{SAT}(T) \qquad \forall i = 1 \ldots NC$$

Note this reduces to the relationship above for a single chemical species.

EQUILIBRIUM SYSTEM — FLASH DRUM

(c) for low pressure gases, non-ideal liquid solutions:

$$y_i P = \gamma_i (T, \underline{x}) x_i P_i^{SAT}(T) \qquad \forall i = 1 \ldots NC$$

(d) vapour-liquid equilibrium distribution coefficients (k-values):

$$y_i = k_i(T, P, \underline{x}, \underline{y}) x_i \qquad \forall i = 1 \ldots NC \tag{3}$$

...where the K-values are calculated by any suitable VLE model.

Definition of molar holdups

$$N_i = N^V y_i + N^L x_i \qquad \forall i = 1 \ldots NC \tag{4}$$

Definition of energy holdup

$$U + PV_{TOT} = N^V h^V + N^L h^L \tag{5}$$

Summation of mole fractions (implicitly define N^L and N^V)

$$\sum_{i=1}^{NC} x_i = 1 \tag{6}$$

$$\sum_{i=1}^{NC} y_i = 1 \tag{7}$$

EQUILIBRIUM SYSTEM — FLASH DRUM

Volume constraint

$$V_{TOT} = N^V v^V + N^L v^L \tag{8}$$

Physical Properties (abstract functions)

$$h^V = h^V(T, P, \underline{y}) \tag{9}$$

$$h^L = h^L(T, P, \underline{x}) \tag{10}$$

$$v^V = v^V(T, P, \underline{y}) \tag{11}$$

$$v^L = v^L(T, P, \underline{x}) \tag{12}$$

Degree of freedom analysis:

Total number of quantities = 4NC + 16

Time invariant parameters

$$V_{TOT} \qquad (= 1)$$

Natural input set:

$$F(t), z_i(t), T_{IN}(t), P_{IN}(t), h_{IN}(t), V(t), L(t) \qquad (= \text{NC+16})$$

Remaining variables:

$$N_i, x_i, y_i, U, h^V, h^L, N^V, N^L, P, T, v^V, v^L \qquad (= \text{3NC+9})$$

Equations: 3NC+9

=> degrees of freedom satisfied for this input set.

EQUILIBRIUM SYSTEM — FLASH DRUM

Exercise: derive an alternative model for the equilibrium flash vessel in which separate species and energy balances are written for both liquid and vapour phases. With reference to the discussion of the index of differential-algebraic equations (see later):

(a) what are the mathematical properties of this model?

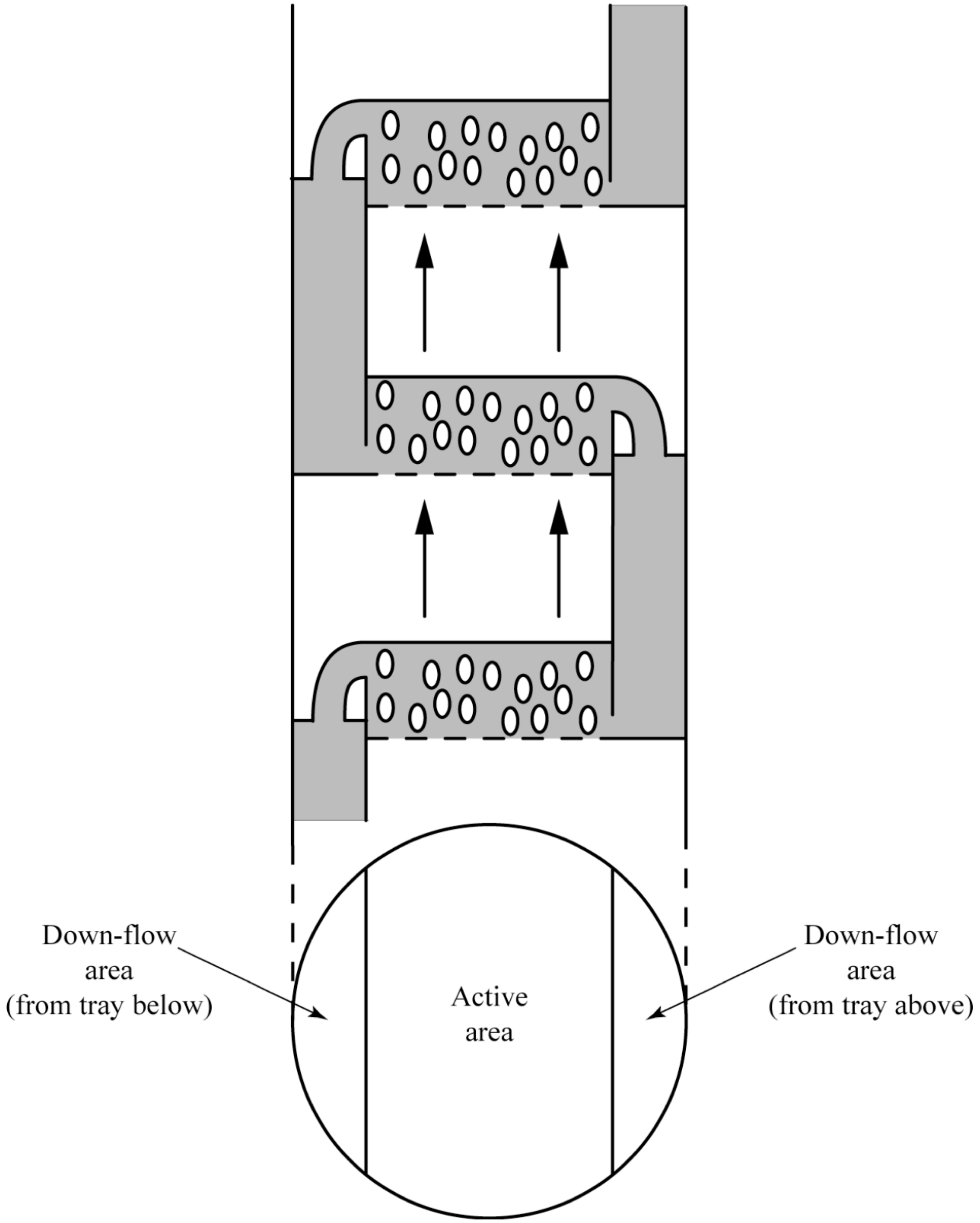
(b) what additional information does this model calculate?

(c) from a practical point of view, which of the two models is appropriate for implementation in SpeedUp?

Hint: it is useful to introduce variables in the balance equations representing the flow of energy and each chemical species between the two phases.

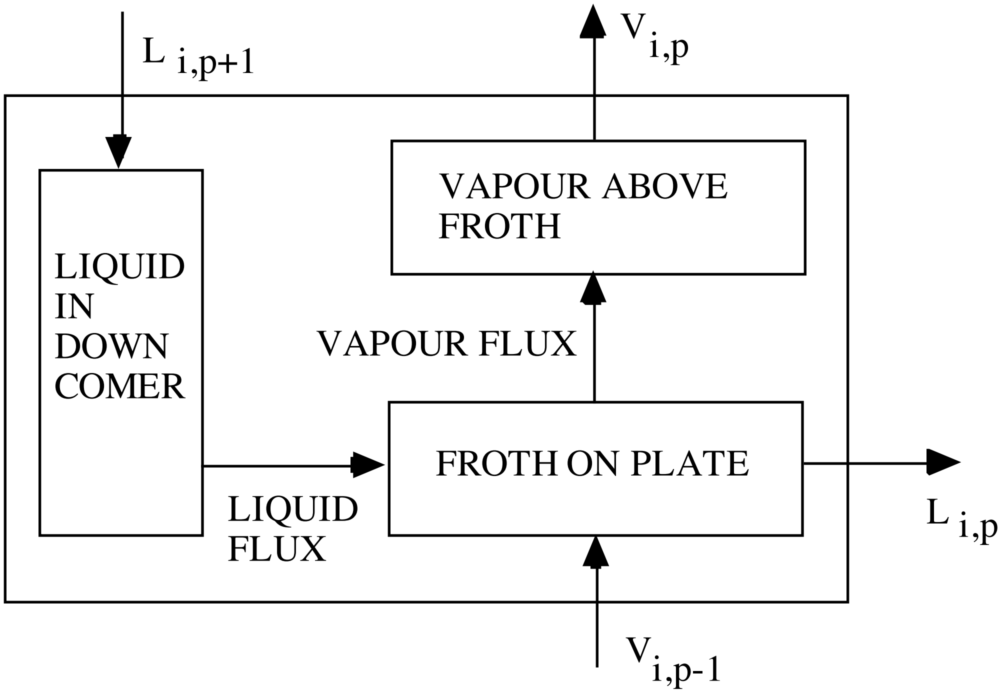
EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Consider the following abstraction of a distillation tower:


Down-flow area (from tray below)
Active area
Down-flow area (from tray above)

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Each stage has the following regions of material and energy accumulation:



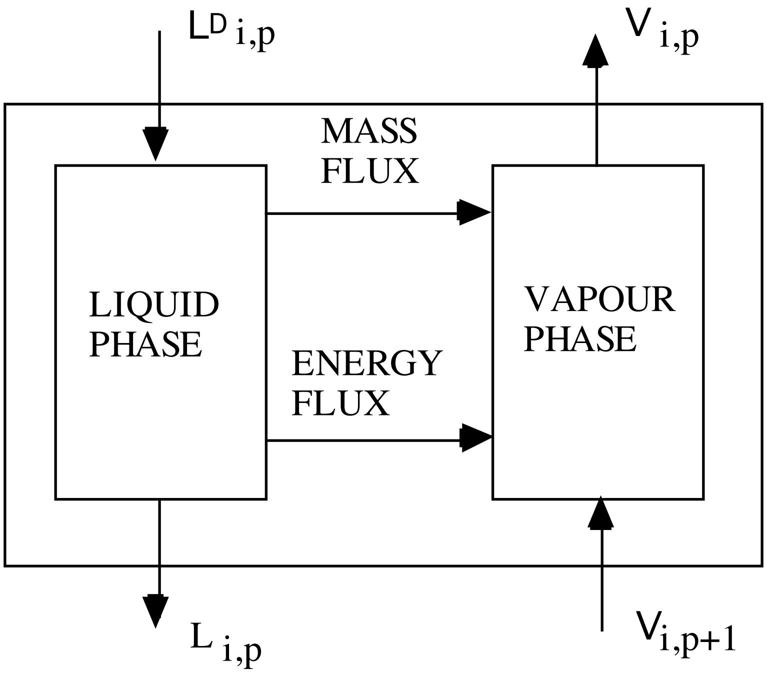
. . . having identified this decomposition, we can focus on modelling each region in isolation.

Here, we will derive a model for the froth on the plate, which is presumably designed to approach phase equilibrium as closely as possible.

Note trays are numbered from the top of the column downwards.

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Froth on plate:



Control volume: the froth on the plate (e.g., encompassing both phases)

Assumptions:

- phases at phase equilibrium
- individual phases well mixed
- adiabatic
- both phases present at all times — e.g., model valid while $T_{BUBBLE} \le T \le T_{DEW}$
- number of moles in vapour phase negligible in comparison to liquid phase
- PV work terms in energy balance negligible

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Variables (i= species number, p = stage number)**:**

N_p	number of moles in froth on stage p	[mol]
L_p^D	total molar flow of liquid from down comer of stage p	[mol s^{-1}]
V_{p+1}	total molar flow of vapour from stage below	[mol s^{-1}]
L_p	total molar flow of liquid leaving froth	[mol s^{-1}]
V_p	total molar flow of vapour leaving froth	[mol s^{-1}]
$x_{i,p}$	mole fraction of component i in liquid phase of froth	
$y_{i,p}$	mole fraction of component i in vapour phase of froth	
$x_{i,p}^D$	mole fraction of component i in down comer	
$y_{i,p+1}$	mole fraction of component i in vapour rising from stage below	
U_p	internal energy of froth	[J]
h_p^D	molar enthalpy of liquid in down comer	[J mol^{-1}]
h_{p+1}^V	molar enthalpy of vapour rising from stage below	[J mol^{-1}]
h_p^L	molar enthalpy of liquid phase of froth	[J mol^{-1}]

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

h_p^V	molar enthalpy of vapour phase of froth	[J mol^{-1}]
T_p	temperature of froth	[K]
P_p	pressure of froth	[Nm^{-2}]
v_p^L	molar volume of liquid in froth	[m^{3}mol^{-1}]
P_{p+1}	pressure on stage below	[Nm^{-2}]

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Equations:

Total mole balance

$$\frac{dN_p}{dt} = L_p^D + V_{p+1} - L_p - V_p \tag{1}$$

NC-1 species balances

$$\frac{d(x_{i,p} N_p)}{dt} = x_{i,p}\frac{dN_p}{dt} + N_p \frac{dx_{i,p}}{dt} = \\ L_p^D x_{i,p}^D + V_{p+1} y_{i,p+1} - L_p x_{i,p} - V_p y_{i,p} \\ \forall i = 1 \ldots NC - 1 \tag{2}$$

Energy balance (both pressure and volume of froth can vary — see assumptions above)

$$\frac{dU_p}{dt} = L_p^D h_p^D + V_{p+1} h_{p+1}^V - L_p h_p^L - V_p h_p^V \tag{3}$$

Phase equilibrium

$$y_{i,p} = k_{i,p}(T_p, P_p, \underline{x}_p, \underline{y}_p) x_{i,p} \qquad \forall i = 1 \ldots NC \tag{4}$$

Definition of energy holdup

$$U_p + P_p N_p v_p^L = N_p h_p^L \tag{5}$$

EQUILIBRIUM SYSTEM — DISTILLATION TRAY

Summation of mole fractions

$$\sum_{i=1}^{NC} x_{i,p} = 1 \tag{6}$$

$$\sum_{i=1}^{NC} y_{i,p} = 1 \tag{7}$$

Flow over weir (e.g., Francis weir formula)

$$L_p = f(N_p, v_p^L) \tag{8}$$

Relate vapour flow into plate to pressure drop across plate (e.g., hydrostatic head and dry plate losses)

$$V_{p+1} = f(P_p, P_{p+1}, N_p) \tag{9}$$

Physical properties

$$h_p^V = h^V(T_p, P_p, \underline{y}_p) \tag{10}$$

$$h_p^L = h^L(T_p, P_p, \underline{x}_p) \tag{11}$$

$$v_p^L = v^L(T_p, P_p \underline{x}_p) \tag{12}$$

See Pantelides *et. al.* (1988) for a more thorough discussion of the properties of this model.

NON EQUILIBRIUM SYSTEMS

(a) still a developing area - active research at the moment

(b) undoubtedly more appropriate for dynamic models; in many cases it is not reasonable to assume that phase equilibrium is reached at a much faster rate than the time constants of interest.

(c) when deriving a model for a simple system:

(i) in a simple system we can assume the pressures equilibrate:

$$P^1 = P^2 = \ldots . P^\pi$$

i.e. (π-1) equations.

(ii) derive separate species balances for each species in each phase — include species flux terms to the other phases

(iii) derive separate energy balances for each phase — include energy flux terms to the other phases.

(d) species and energy fluxes between each phase are determined by multi-component heat and mass transfer relationships:

(i) these are extremely difficult to generalize — must be derived on a case by case basis considering geometry, flow patterns, etc.

(ii) usually dependent on empirically derived correlations which may only be valid in a very small operating region — e.g., at or near steady-state. If the dynamics take the system outside this region, model assumptions are no longer valid!

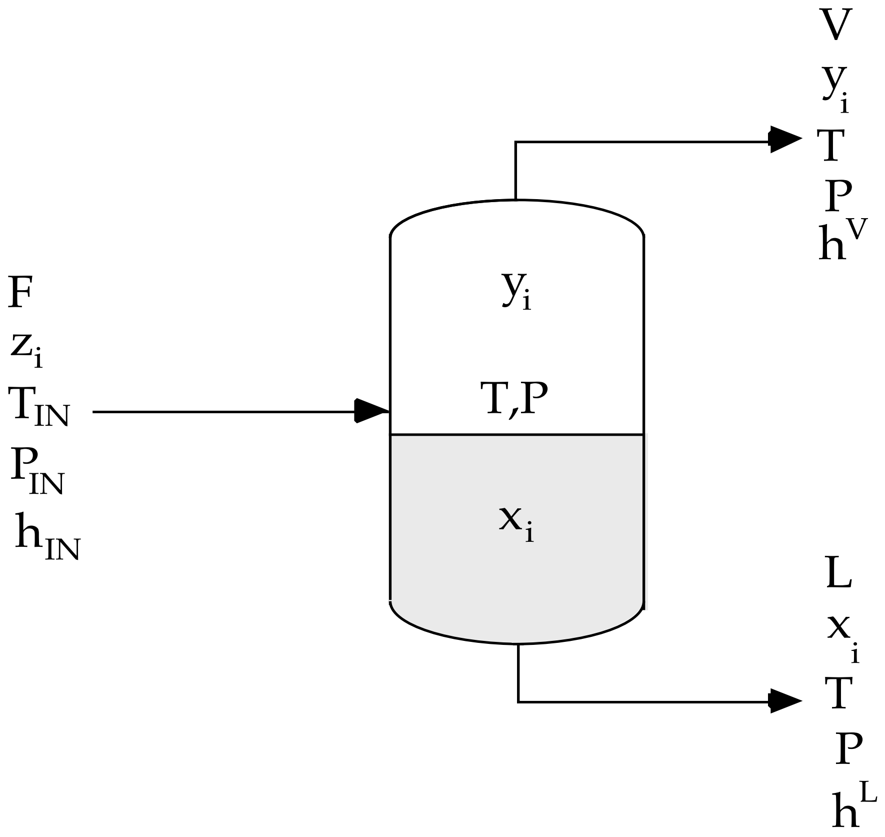
NON EQUILIBRIUM SYSTEMS

(e) the models get larger and more complicated, and more detailed information concerning geometry, internal features of the vessel, etc., must be considered during model development.

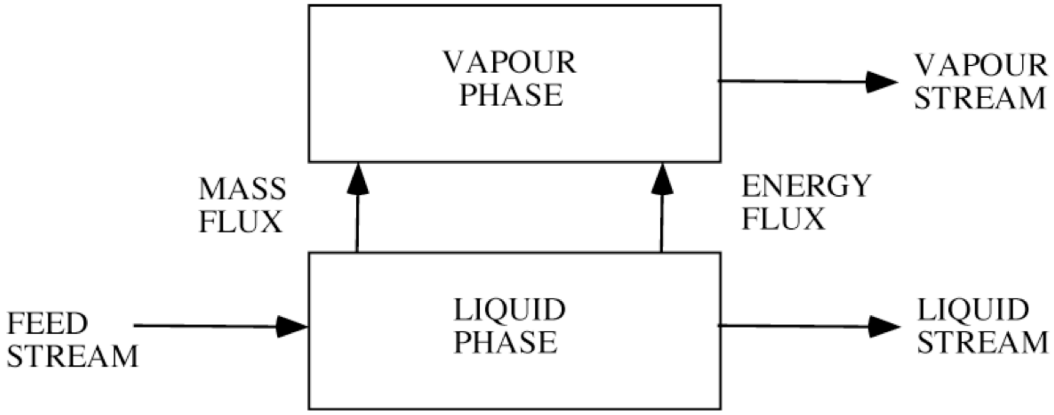
. . . research continues.

NON EQUILIBRIUM SYSTEM - FLASH DRUM

Consider a model of a flash drum in which it cannot be assumed that the vapour and liquid phases are in phase equilibrium:



The vessel can be abstracted as two well-mixed phases constrained within a fixed volume. Material and energy fluxes exist between the phases:



NON EQUILIBRIUM SYSTEM - FLASH DRUM

Control volumes: control volume for gas phase *and* control volume for liquid phase. Both will vary in size and shape as the liquid level rises and falls.

Assumptions:

- individual phases are well-mixed.
- vessel adiabatic
- liquid feed stream well-mixed with the liquid phase
- both phases present at all times
- bulk flows into and out of vessel specified by user or determined by upstream and downstream models

These assumptions are the simplest assumptions that can be made. It is interesting to note that to assume anything else would require a detailed knowledge of the internal topology of the vessel *and* physico-chemical discontinuities (e.g. the level of the feed pipe relative to that of the liquid level, the area of heating coil/jacket immersed in the liquid phase, etc.).

NON EQUILIBRIUM SYSTEM - FLASH DRUM

Variables:

N_i^L	number of moles of component i in liquid phase	[mol]
N_i^V	number of moles of component i in vapour phase	[mol]
F	total molar flow to vessel	[mol s^{-1}]
z_i	mole fraction of component i in feed stream	
A	area of phase boundary	[m^2]
ϕ_i	molar flux of component i from liquid to vapour phase	[mol $m^{-2}s^{-1}$]
L	total liquid flow from vessel	[mol s^{-1}]
x_i	mole fraction of component i in liquid phase	
V	total vapour flow from vessel	[mol s^{-1}]
y_i	mole fraction of component i in vapour phase	
H^L	enthalpy of liquid phase	[J]
h_{IN}	molar enthalpy of feed stream	[J mol^{-1}]
N^L	total number of moles in liquid phase	[mol]
v^L	molar volume of liquid phase	[m^3 mol^{-1}]
P	pressure of vessel contents	[Nm^{-2}]
ϕ_H	energy flux from liquid to vapour phase (heat + material flow)	[$Jm^{-2}s^{-1}$]
h^L	molar enthalpy of liquid	[J mol^{-1}]
H^V	enthalpy of vapour phase	[J]

NON EQUILIBRIUM SYSTEM - FLASH DRUM

N^V	total number of moles in vapour phase	[mol]
V^V	molar volume of vapour phase	[J mol^{-1}]
h^V	molar enthalpy of vapour phase	[J mol^{-1}]
x_i^*	mole fraction of component i in liquid phase at phase boundary	
y_i^*	mole fraction of component i in vapour phase at phase boundary	
k_i^L	mass transfer coefficient for component i in the liquid phase	[mol m^{-2}s^{-1}]
k_i^V	mass transfer coefficient for component i in the vapour phase	[mol m^{-2}s^{-1}]
k_H^L	heat transfer coefficient in the liquid phase	[J m^{-2}k^{-1}s^{-1}]
k_H^V	heat transfer coefficient in the vapour phase	[J m^{-2}k^{-1}s^{-1}]
T^L	temperature of liquid phase	[K]
T^V	temperature of vapour phase	[K]
$T*$	temperature at phase boundary	[K]
V_{TOT}	vessel volume	[m^3]

NON EQUILIBRIUM SYSTEM - FLASH DRUM

Equations:

Species balance for each species in the liquid phase:

$$\frac{dN_i^L}{dt} = Fz_i - A\phi_i - Lx_i \qquad \forall i = 1 \ldots NC \qquad (1)$$

Species balance for each species in the vapour phase:

$$\frac{dN_i^V}{dt} = A\phi_i - Vy_i \qquad \forall i = 1 \ldots NC \qquad (2)$$

Energy balance for the liquid phase (pressure and volume can vary)

$$\frac{dH^L}{dt} = Fh_{IN} + N^L v^L \frac{dP}{dt} - A\phi_H - Lh^L \qquad (3)$$

Energy balance for the vapour phase:

$$\frac{dH^V}{dt} = A\phi^H + N^V v^V \frac{dP}{dt} - Vh^V \qquad (4)$$

NON EQUILIBRIUM SYSTEM - FLASH DRUM

Interphase mass and heat transfer relations:

$$\phi_i = k_i^L\left(x_i - x_i^*\right) \qquad \forall i = 1\ldots NC \qquad (5)$$

$$\phi_i = k_i^V\left(y_i^* - y_i\right) \qquad \forall i = 1\ldots NC \qquad (6)$$

$$\phi_H = k_H^L(T^L - T^*) + h^L \sum_{i=1}^{NC} \phi_i \qquad (7)$$

$$\phi_H = k_H^V(T^* - T^V) + h^V \sum_{i=1}^{NC} \phi_i \qquad (8)$$

Assume equilibrium at the phase boundary:

$$y_i^* = K_i^*\left(T^*, P, \underline{x}^*, \underline{y}^*\right) x_i^* \qquad \forall i = 1\ldots NC \qquad (9)$$

Definitions of molar holdups

$$N_i^L = N^L x_i \qquad \forall i = 1\ldots NC \qquad (10)$$

$$N_i^V = N^V y_i \qquad \forall i = 1\ldots NC \qquad (11)$$

Definitions of enthalpy holdups

$$H^L = N^L h^L \qquad (12)$$

$$H^V = N^V h^V \qquad (13)$$

NON EQUILIBRIUM SYSTEM - FLASH DRUM

Volume constraint:

$$V_{TOT} = N^V v^V + N^L v^L \tag{14}$$

Summation of mole fractions

$$\sum_{i=1}^{NC} x_i = 1 \tag{15}$$

$$\sum_{i=1}^{NC} y_i = 1 \tag{16}$$

Physical Properties (abstract functions)

$$v^L = v^L\left(T^L, P, \underline{x}\right) \tag{17}$$

$$v^V = v^V\left(T^V, P, \underline{y}\right) \tag{18}$$

$$h^L = h^L\left(T^L, P, \underline{x}\right) \tag{19}$$

$$h^V = h^V\left(T^V, P, \underline{y}\right) \tag{20}$$

DYNAMIC MODELLING — CONCLUSIONS

Dynamic modelling is a difficult task that must be addressed on a case by case basis:

(i) understand the system under investigation — what are details of the internals of the system:

  (a) geometry

  (b) internal topology

  (c) phases present

  (d) when behaviour changes — e.g., buffer tank overflows.

(ii) what are the contributions to the dynamics — control volume analysis: mass, energy, and/or momentum dynamics.

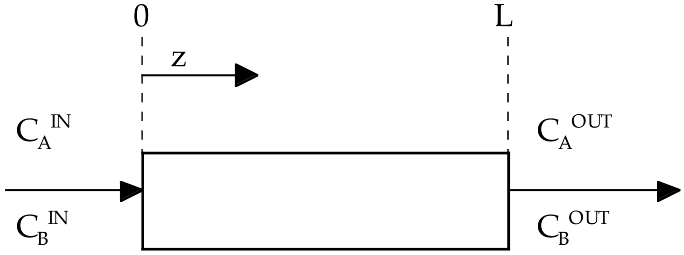
(iii) apply chemical engineering principles to derive a complete model.

(iv) develop and maintain full model documentation — *never* extrapolate model beyond region in which assumptions are valid.

DISTRIBUTED SYSTEMS

All the above discussion is based on the premise that it is reasonable to assume that the macroscopic control volumes employed to derive the model equations are well mixed. Whether or not this assumption is reasonable, our options are pretty limited by the capabilities of current process modelling technology. In this section we will briefly discuss the current options when the above assumption breaks down.

Example: isothermal plug flow tubular reactor with axial dispersion:



2A→B

Equations: species balances derived from microscopic control volume analysis:

$$\frac{\partial C_A}{\partial t} = D\frac{\partial^2 C_A}{\partial z^2} - v\frac{\partial C_A}{\partial z} - 2kC_A^2$$
$$\frac{\partial C_B}{\partial t} = D\frac{\partial^2 C_B}{\partial z^2} - v\frac{\partial C_B}{dz} + kC_A^2$$

Note that these are now *partial differential equations* (PDEs) rather than ordinary differential equations, with the following independent variables:

- time
- axial position along the reactor (in the domain $z \in [O,L]$).

The concentrations of the species in the reactor are now a function of both time and spatial position in the above domain (i.e., $C_A(z,t)$). We need to solve for these functions (or *fields*).

Note that this problem is somewhat more difficult than one in reaction engineering where we would explicitly define the boundary conditions as functions and solve the reactor model in isolation. Here, we do not know the boundary conditions explicitly because the inlet and outlet streams are coupled to upstream and downstream units, and the boundary conditions are determined implicitly by numerical solution of these other units. As the solution to the reactor model (at least) influences the downstream boundary condition (and the upstream one if there is a recycle stream), all the units must be solved simultaneously. This gives rise to a process model composed of a mixed system of partial differential, ordinary differential, and algebraic equations (PDAEs). Further, the PDEs in this system may be distributed on multiple spatial domains, each corresponding to the spatial dimensions of different vessels in the flowsheet. Finally, as noted above, the boundary conditions of these multiple domains are coupled.

Current technology will not accept a PDE model in the form above. *One* approach is to apply what is called the *Method of Lines*. Here, the PDE model is reduced to a system of ODEs by discretization of the spatial domain. The system of ODEs is then coded into the simulation package and the overall process model is solved with general purpose ODE/DAE solvers.

In the example above, discretization of the spatial domain yields a set of ODEs at a series of interior (or mesh) points in the domain. At these points, the spatial derivatives are replaced by a discrete approximation, for example a centered differentiation formula on a regular mesh:

$$\frac{\partial C_{A,i}}{\partial z} \approx \frac{C_{A,i+1} - C_{A,i-1}}{2h}$$
$$\frac{\partial^2 C_{A,i}}{\partial z^2} \approx \frac{C_{A,i+1} - 2C_{A,i} + C_{A,i-1}}{h^2}$$

where the subscript *i* denotes the spatial mesh point, and *h* the distance between mesh points.

Substituting these into the PDE at each mesh point yields the system of ODEs:

$$\frac{dC_{A,i}}{dt} = D\frac{(C_{A,i+1} - 2C_{A,i} + C_{A,i-1})}{h^2} - v\frac{(C_{A,i+1} - C_{A,i-1})}{2h} - kC_{A,i}{}^2$$
$$\forall i = 1 \ldots Nmesh$$

Note that in the discretized model we solve for the time variation of the concentrations at a series of fixed spatial positions (i.e., $C_{A,i}(t)$). The above equations have to be tweaked to take into account the boundary conditions.

The gPROMS simulator (Barton, 1992; Oh, 1995) can now perform this discretization on a fixed spatial mesh automatically. Note that only the derivation of the ODEs is automated, so all the observations below also apply to gPROMS.

However, this approach should be applied with extreme caution (i.e., these notes are not sufficient, study this in more detail). Two important issues are:

- an appropriate discretization formula is very dependent on the physics of the system (e.g., the centered difference formula is appropriate for systems with dispersion, but *not* for purely convective systems).
- the spatial discretization introduces error into the solution. With a fixed regular mesh, the number of mesh points must be sufficiently large so that this discretization error is below the desired tolerance during all transients of interest. This can lead to very big systems of ODEs.

The above approach is called the finite difference method. There also two widely used alternatives: finite element methods and spectral methods. Research continues . . .

SECTION II

DYNAMIC SIMULATION

Paul I. Barton
Department of Chemical Engineering
Massachusetts Institute of Technology
Cambridge, MA

© copyright Paul I. Barton, July 1999

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

We will be concerned with *explicit first-order systems of ordinary differential equations* (ODEs) of the form:

$$\frac{dx_1}{dt} = \dot{x}_1 = f_1(x_1, x_2 ...., x_n, t)$$
$$\frac{dx_2}{dt} = \dot{x}_2 = f_2(x_1, x_2 ...., x_n, t)$$
$$\vdots \qquad\qquad \vdots$$
$$\frac{dx_n}{dt} = \dot{x}_n = f_n(x_1, x_2 ...., x_n, t)$$

or in vector notation:

$$\frac{d\underline{x}}{dt} = \dot{\underline{x}} = \underline{f}(\underline{x}(t), t) \qquad \text{(ODE)}$$

Note: in general, $\underline{f}$ is a n vector of nonlinear functions involving the *dependent (or state) variables* $\underline{x}(t)$ and the *independent variable* t (which we will always assume to be time).

First-order refers to the fact that only first derivatives of the independent variables appear. *Explicit* refers to the fact that each time derivative is expressed individually as an explicit function of the variables (therefore the Jacobian with respect to $\dot{\underline{x}}$ is the identity matrix $\underline{\underline{I}}$).

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

The more general *implicit form* is defined in terms of general functional relationships between $\dot{\underline{x}}$, $\underline{x}$, and t, e.g.:

$$\underline{F}(\dot{\underline{x}},\underline{x},t)=0 \qquad \text{(IODE)}$$

with the condition that the Jacobian (matrix of partial derivatives) of these functions with respect to $\dot{\underline{x}}$ is nonsingular everywhere in the time horizon of interest (to distinguish from a system of *differential-algebraic equations* — see later).

It is important to realize that system (ODE) is coupled — each function can in general depend on *all* the dependent variables. The consequence of this coupling is that the equations cannot be solved independently and thus must be solved *simultaneously*.

(ODE) is also sometimes referred to as a *nonlinear state-space description* of a dynamic system.

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

The *general solution* $\underline{x}(t)$ of the first order system (ODE):

- involves, in general, n arbitrary constants.
- these constants can be fixed by imposing *n side conditions* => a *particular* solution satisfying these side conditions.
- if these side conditions fix the values for all the elements of $\underline{x}$ at the same initial point in time, the combination of the system (ODE) and the side conditions constitutes an *initial value problem* (IVP):

$$\left.\begin{aligned} &\dot{\underline{x}} = \underline{f}(\underline{x},\underline{t}) \\ &\underline{x}(t_0) = \underline{x}_0 \\ &t \in [t_0, t_f] \end{aligned}\right\} \qquad \text{(IVP)}$$

where the notation $\underline{x}(a)$ denotes the vector of variables at time a, t_0 is used to denote the initial time, t_f the final time. Note the distinction between *system* and *problem*.

Dynamic simulation of chemical processes gives rise to initial value problems.

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

In system (ODE), if $\underline{f}$ is independent of t, the problem (and the system) are said to be *autonomous* and *non-autonomous* otherwise. A non-autonomous system can be converted to an autonomous system by defining a new variable:

$$x_{n+1} = t \qquad \Rightarrow \qquad \dot{x}_{n+1} = 1$$

and the autonomous system:

$$\dot{\underline{X}} = \underline{F}(\underline{X})$$

where,

$$\underline{X} = (x_1, x_2, \ldots, x_n \vdots x_{n+1})^T$$
$$\underline{F} = (f_1, f_2, \ldots, f_n \vdots 1)^T$$

so, there is no loss of generality in assuming a system is autonomous.

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

Higher order systems (e.g., those involving the second and higher order derivatives of the dependent variables with respect to the independent variable) can always be transformed into a (larger) equivalent first order system.

Example: consider the second order system:

$$\ddot{x}_1 = f_1(x_1, x_2)$$
$$\dot{x}_2 = f_2(x_1, x_2)$$

define a new variable $x_3 = \dot{x}_1$:

$$\Rightarrow \quad \dot{x}_3 = \ddot{x}_1$$

and this leads to the first order autonomous system:

$$\begin{aligned} &\dot{x}_1 = x_3 \\ \Rightarrow \quad &\dot{x}_2 = f_2(x_1, x_2) \\ &\dot{x}_3 = f_1(x_1, x_2) \end{aligned}$$

So, there is again no loss of generality in assuming a system is first-order.

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

The first-order system (ODE) is said to be *linear* (or *linear time varying*) if it takes the form:

$$\underline{\dot{x}} = \underline{\underline{A}}(t)\underline{x} + \underline{g}(t)$$

Further if the $n \times n$ coefficient matrix $\underline{\underline{A}}$ is independent of t , the system is said to be *linear with constant coefficients* (or *linear time invariant*):

$$\underline{\dot{x}} = \underline{\underline{A}}(t)\underline{x} + \underline{g}(t) \qquad \text{(LTI)}$$

A closed form general solution can be obtained for the autonomous linear constant coefficient system:[1]

$$\underline{\dot{x}} = \underline{\underline{A}}\underline{x}$$

expressed as:

$$\underline{x}(t) = \sum_{i=1}^{n} c_i \exp(\lambda_i t)\underline{s}_i$$

where λ_i is an eigenvalue of $\underline{\underline{A}}$, $\underline{s}_i$ is the corresponding eigenvector, and c_i are the arbitrary constants that characterize the general solution (see above). The c_i can be calculated for a particular solution given values for $\underline{x}$ at some initial time (an initial value problem).

[1] the result is stated assuming $\underline{\underline{A}}$ has n distinct eigenvalues - the Jordan form defines the solution when this does not hold

ORDINARY DIFFERENTIAL EQUATIONS DEFINITIONS

System (LTI) is sometimes written:

$$\dot{\underline{x}} = \underline{\underline{A}}\,\underline{x} + \underline{\underline{B}}\,\underline{u}(t)$$

where $\underline{u}(t)$ are the known inputs to the systems. This form, which is popular with control engineers, is called a *linear time invariant* system.

Linear constant coefficient systems are the only general class of systems for which analytical solutions can always be found. Process simulation problems give rise to nonlinear systems, so we must resort to numerical solution of (IVP) (*numerical integration*).

Numerical methods generate *approximations* to particular solutions satisfying given initial conditions. Obviously, it is desirable if we guarantee that the results of a numerical method are within a specified numerical tolerance of the "true" particular solution => guarantees an accuracy of the dynamic simulation results (e.g., that they are a solution of the model equations — this does *not* guarantee that the model itself is an adequate representation of the true system behaviour).

DIFFERENTIAL-ALGEBRAIC EQUATIONS DEFINITIONS

A generalization of the explicit linear time invariant ODE (LTI) can be expressed as:

$$\underline{\underline{A}}\dot{\underline{x}} + \underline{\underline{B}}\underline{x} = \underline{g}(t) \qquad \text{(LTIDAE)}$$

If the matrix $\underline{\underline{A}}$ is nonsingular, then this system reduces trivially to an ODE:

$$\dot{\underline{x}} = -\underline{\underline{A}}^{-1}\underline{\underline{B}}\underline{x} + \underline{\underline{A}}^{-1}\underline{g}(t) = \underline{\underline{C}}\underline{x} + \underline{h}(t)$$

on the other hand, is $\underline{\underline{A}}$ singular (i.e., $\underline{\underline{A}}^{-1}$ does not exist) then (LTIDAE) is known as a *linear time invariant differential-algebraic equation* (or DAE).

Example: if $\underline{\underline{A}}$ has a row of all zeroes (making it singular), then that row of (LTIDAE) will amount to a completely algebraic relationship between the state variables. This is the origin of the term differential-algebraic equation.

See (Campbell, 1980; Campbell, 1982; Brenan *et al*., 1996) for detailed discussions of the theory of DAEs.

DIFFERENTIAL-ALGEBRAIC EQUATIONS DEFINITIONS

Necessary and sufficient conditions for solvability of (LTIDAE) are that the family of matrices $\lambda\underline{\underline{A}} + \underline{\underline{B}}$ defined by the scalar variable λ is nonsingular for some value of $\lambda \in \mathbf{R}$ (the family $\lambda\underline{\underline{A}} + \underline{\underline{B}}$ is known as a *matrix pencil*). Such a matrix pencil is said to be *regular*.

Example: if $\underline{\underline{B}}$ is nonsingular, then the matrix pencil is regular (i.e., at least for $\lambda = 0$, $\lambda\underline{\underline{A}} + \underline{\underline{B}}$ is nonsingular).

If the matrix pencil $\lambda\underline{\underline{A}} + \underline{\underline{B}}$ is regular then there exist nonsingular matrices $\underline{\underline{P}}$ and $\underline{\underline{Q}}$ such that:

$$\underline{\underline{P}}\,\underline{\underline{A}}\,\underline{\underline{Q}} = \begin{bmatrix} \underline{\underline{I}} & 0 \\ 0 & \underline{\underline{N}} \end{bmatrix} \text{ and } \underline{\underline{P}}\,\underline{\underline{B}}\,\underline{\underline{Q}} = \begin{bmatrix} \underline{\underline{C}} & 0 \\ 0 & \underline{\underline{I}} \end{bmatrix}$$

where $\underline{\underline{N}}$ is a nilpotent matrix to degree ν (i.e., $\underline{\underline{N}}^{\nu-1} \neq 0$ and $\underline{\underline{N}}^{\nu} = 0$).

Note that ν is equivalent to the *differential index* (see definition below) of (LTIDAE).

This property can be used to construct an analytic solution for (LTIDAE).

DIFFERENTIAL-ALGEBRAIC EQUATIONS DEFINITIONS

Applying the co-ordinate transformations defined by $\underline{\underline{P}}$ and $\underline{\underline{Q}}$ to (LTIDAE) (i.e., multiply through by $\underline{\underline{P}}$ and define $\underline{x} = \underline{\underline{Q}}\,\underline{y}$) combined with the above identities yields:

$$\underline{\underline{A}}\dot{\underline{x}} + \underline{\underline{B}}\,\underline{x} = \underline{g}(t) \qquad \text{(LTIDAE)}$$

$$\Rightarrow \quad \underline{\underline{P}}\,\underline{\underline{A}}\,\underline{\underline{Q}}\,\dot{\underline{y}} + \underline{\underline{P}}\,\underline{\underline{B}}\,\underline{\underline{Q}}\,\underline{y} = \underline{\underline{P}}\,\underline{g}(t)$$

$$\Rightarrow \quad \begin{aligned} \dot{\underline{y}}_1 + \underline{\underline{C}}\,\underline{y}_1 &= \underline{h}_1(t) \\ \underline{\underline{N}}\,\dot{\underline{y}}_2 + \underline{y}_2 &= \underline{h}_2(t) \end{aligned}$$

The first block row of this transformed and uncoupled system is just a linear time invariant ODE for which the analytic solution is already known. The analytic solution of the second block row can be constructed as follows:

$$\left(\underline{\underline{N}}\frac{d}{dt} + \underline{\underline{I}}\right)\underline{y}_2(t) = \underline{h}_2(t)$$

and using a famous identity yields:

$$\underline{y}_2(t) = \sum_{i=0}^{\infty}(-1)^i\left(\underline{\underline{N}}\frac{d}{dt}\right)^i \underline{h}_2(t) = \sum_{i=0}^{\nu-1}(-1)^i\,\underline{\underline{N}}^i\,\frac{d^{(i)}\underline{h}_2}{dt^{(i)}}(t)$$

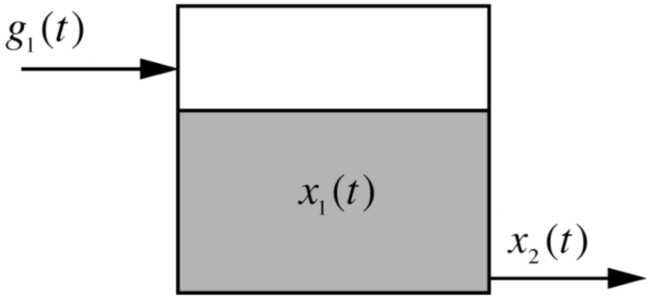
the infinite series being truncated by the nilpotency of $\underline{\underline{N}}$.

DIFFERENTIAL-ALGEBRAIC EQUATIONS DEFINITIONS

Note that solvability in general will also require sufficient differentiability of the right hand sides (or *forcing functions*). This result also implies that the differential index must be ≥ 2 for time derivatives of the forcing functions to appear in the solution.

DIFFERENTIAL-ALGEBRAIC EQUATIONS EXAMPLES

Example 1:



$$\dot{x}_1 + x_2 = g_1(t)$$
$$x_2 = g_2(t)$$

Hence:

$$\underline{\underline{A}} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \underline{\underline{B}} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\underline{\underline{P}} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \quad \underline{\underline{Q}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \underline{\underline{P}}\,\underline{\underline{A}}\,\underline{\underline{Q}} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\Rightarrow \underline{\underline{P}}\,\underline{\underline{B}}\,\underline{\underline{Q}} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$=> \underline{\underline{N}} = [0] \qquad \underline{\underline{C}} = [0]$$

DIFFERENTIAL-ALGEBRAIC EQUATIONS EXAMPLES

Hence the system is nilpotent to degree 1 (index 1). Only the zero matrix is nilpotent to degree 1, so index 1 system does not involve time derivatives of the inputs.

Analytic solution to Example 1:

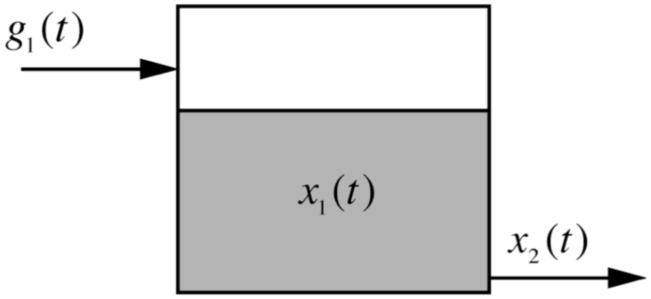
$$\Rightarrow \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} g_1 - g_2 \\ g_2 \end{bmatrix}$$

$$\Rightarrow \begin{aligned} \dot{x}_1 &= g_1(t) - g_2(t) \\ x_2 &= g_2(t) \end{aligned}$$

Which is evident from inspection of the original system.

DIFFERENTIAL-ALGEBRAIC EQUATIONS
EXAMPLES

Example 2:



$$\dot{x}_1 + x_2 = g_1(t)$$
$$x_1 = g_2(t)$$

Hence:

$$\underline{\underline{A}} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \underline{\underline{B}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\underline{\underline{P}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \underline{\underline{Q}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\Rightarrow \underline{\underline{P}}\,\underline{\underline{A}}\,\underline{\underline{Q}} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\Rightarrow \underline{\underline{P}}\,\underline{\underline{B}}\,\underline{\underline{Q}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \underline{\underline{N}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

DIFFERENTIAL-ALGEBRAIC EQUATIONS EXAMPLES

$$\underline{\underline{N}}^2 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Hence the system is nilpotent to degree 2 (index 2).

Analytic solution to Example 2:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y_2 \\ y_1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} g_1' \\ g_2' \end{bmatrix}$$

$$\Rightarrow \begin{array}{l} x_1 = g_2(t) \\ x_2 = g_1(t) - g_2'(t) \end{array}$$

Which is again evident from inspection of the original system.

DIFFERENTIAL-ALGEBRAIC EQUATIONS DEFINITIONS

The analytical solution above reveals several interesting properties of DAEs:

- the solution can involve the first and higher order time derivatives of the forcing functions.
- not all initial conditions will satisfy the solution (i.e., $\underline{y}_2(t_0)$ cannot be chosen arbitrarily).
- the solution must satisfy explicit or implicit algebraic constraints.

These properties will also be revealed by our later, more physically based, analysis.

As in the ODE case, linear time varying and nonlinear DAEs are elusive to general analytical results, so we must resort to numerical solutions.

DYNAMIC SIMULATION

A dynamic model (or system of equations) encapsulates a representation of the time dependent behaviour of a system in the real world.

Dynamic simulation is one[2] activity (or calculation) which utilizes dynamic models. Specifically, to formulate a dynamic simulation we must:

- specify the model representing the system
- the initial state of the system
- the time variation of the inputs to the system

⇒ mathematically an initial value problem that can be solved numerically to determine the time variation of the system from the initial state to some final state.

. . . this clarifies the distinction between a representation of the system (the *model*) and the calculations it is used for (the *simulation*). Obviously the same model may be used for many different simulations (different scenarios).

[2] but by no means the only calculation.

DYNAMIC SIMULATION

More formally, we can view a dynamic simulation as an *experiment* with a model.

An experiment is composed of three elements:

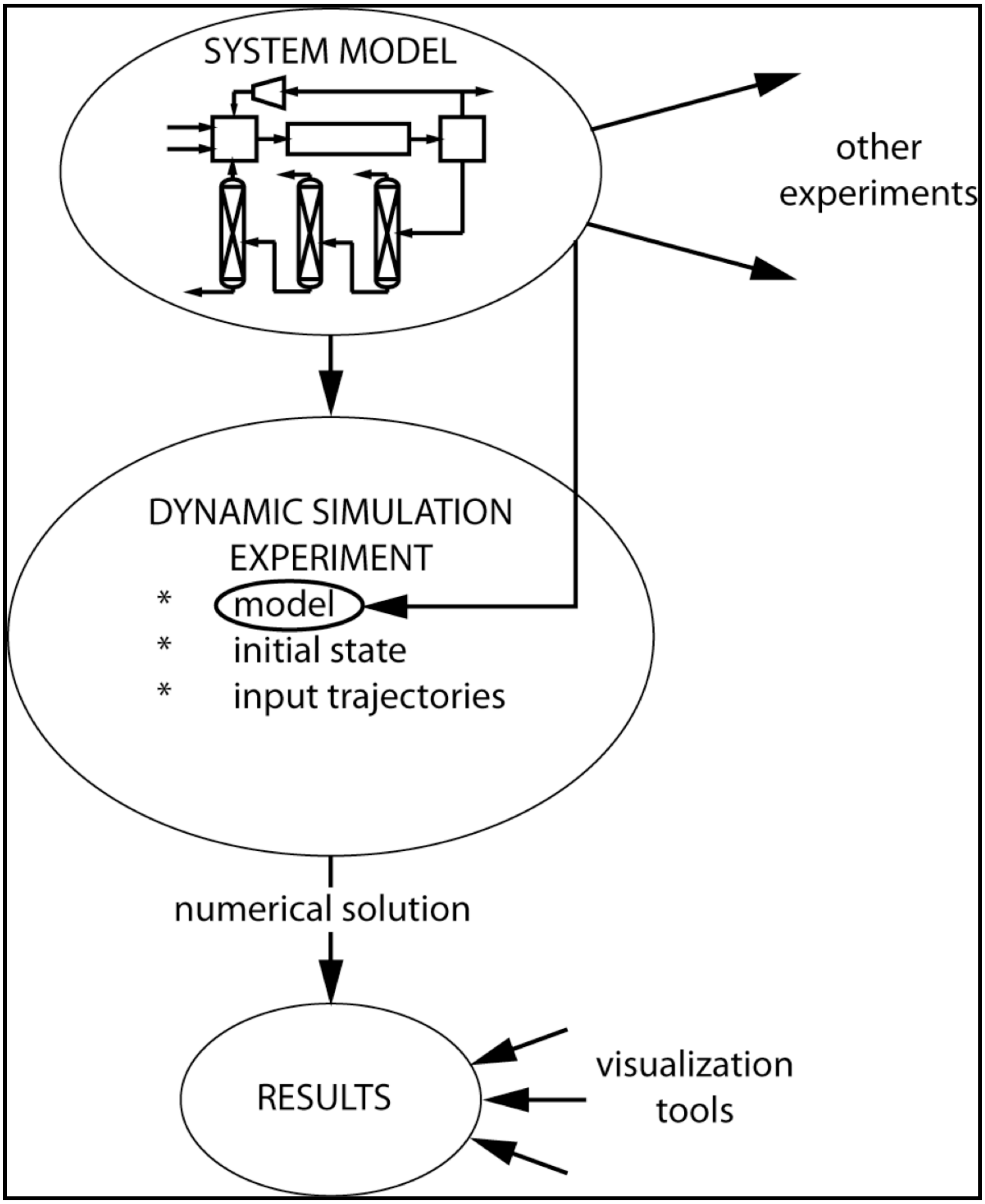
- the object (or artifact, system) under investigation - in this case we use a model to represent the behaviour rather than experimenting with a physical system.
- the experimental frame - e.g., the circumstances under which the system is to be investigated.
- the data generated by execution of the experiment.

Obviously, the experimental frame is unique to each experiment and a dynamic simulation experiment will require specific categories of information in order to be fully defined.

This second section of the notes is concerned with specification and solution of dynamic simulation experiments:

- software tools that support specification and solution of dynamic simulation experiments.
- specifying the experimental frame: initial conditions and input trajectories.
- checking if the experiment is well posed.
- numerical solution of the experiment.

DYNAMIC SIMULATION AS AN EXPERIMENT



. . . the most useful model is one that can be disengaged from the details of individual experiments (it can be *reused*).

SETTING UP THE DYNAMIC SIMULATION EXPERIMENT - EXAMPLE

Let us consider the model of a non isothermal CSTR from section I and identify the information required to define an experimental frame:

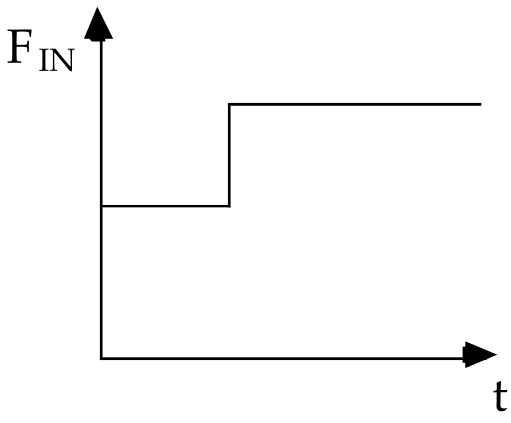
(a) assign values to all the time invariant (constant) parameters:

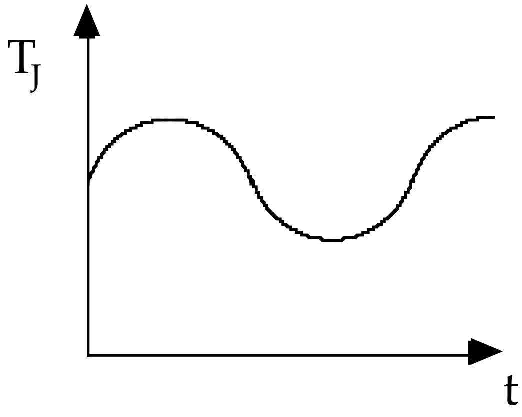
$$A, A_J, U_J, g, K, K_R, \Delta E_R, R, P_0$$

(b) select a subset of the variables to be designated as inputs (disturbances, controls), e.g.:

$$F_{IN}(t), C_{A,IN}(t), C_{B,IN}(t), T_{IN}(t), T_J(t)$$

and specify their time variation explicitly, e.g.:





Denote a variable that is a known function of time by $F_{IN}(t)$ - this function is not necessarily continuous or differentiable (e.g. step change above).

SETTING UP THE DYNAMIC SIMULATION EXPERIMENT - EXAMPLE

Notes:

- the selection of a set of input variables is analogous to the selection of the degrees of freedom in a steady-state simulation.
- once we have selected the input variables, we should have a fully determined set of differential equations (number of unknown variables = number of equations) that determine the time variation of these *unknowns* uniquely.
- from a practical point view, in dynamic simulation we are very limited in our choice of input variables — see discussion of high index problems later (numerical methods for solution of dynamic 'design' problems are only just emerging).

This raises the question of how to select a set of input variables that will lead to a problem we can solve with current technology?

SETTING UP THE DYNAMIC SIMULATION EXPERIMENT - EXAMPLE

(c) specify the initial state of the system (initial condition at $t = t_0$). This defines where we start from! Note:

i) correct numerical solution of the initial value problem requires consistent values for all the unknown variables at $t = t_0$

ii) in general, we are not free to specify initial values for *all* the unknowns independently — the values must satisfy the model equations. In this case, for example, only the values:

$$M(t_0), C_A(t_0), C_B(t_0), T(t_0)$$

are sufficient to define the initial state of the reactor - given these, all the other initial values can be determined by solving the model equations.

Hence we must address the questions:

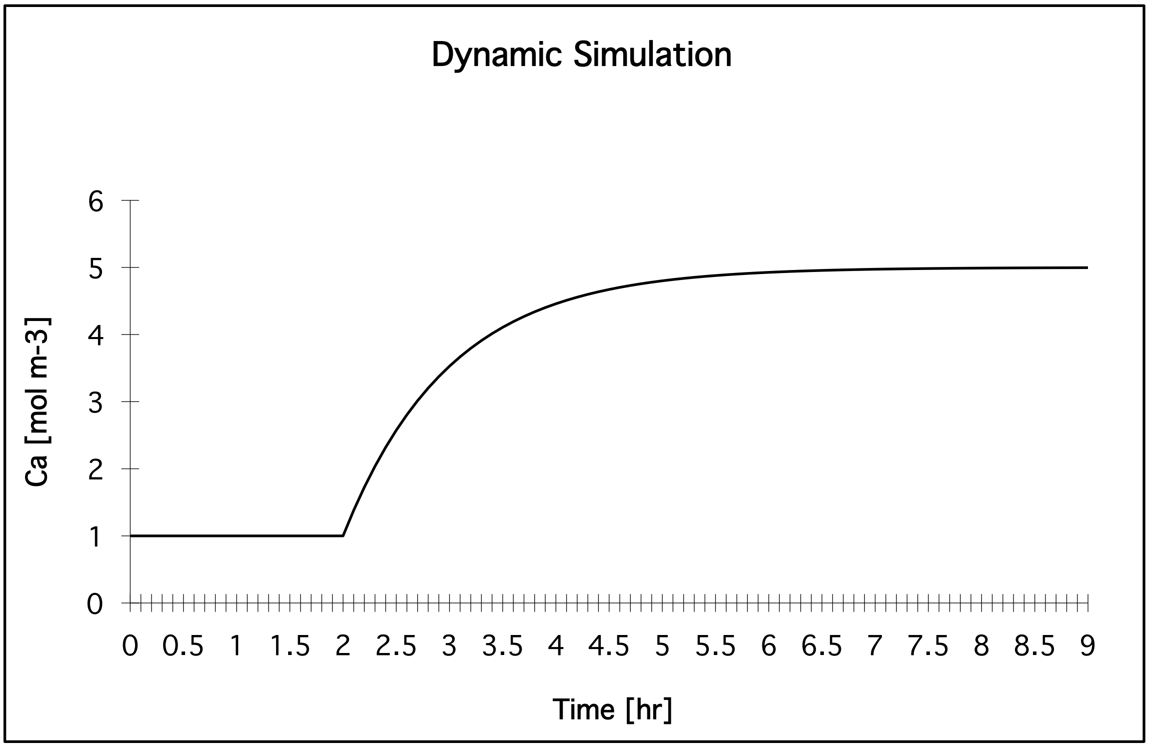
- for models of the general functional form derived in section I, what is the number of side conditions at $t = t_0$ that is sufficient to define uniquely the initial state of the system?
- how are these side conditions expressed?
- given the side conditions, how can initial values for all the unknowns be determined?

(d) the duration of the simulation - e.g. t_f.

SETTING UP THE DYNAMIC SIMULATION EXPERIMENT - EXAMPLE

The initial value problem is now fully defined — submit it to the computer for numerical solution.

This process will determine the variation of all the unknowns with respect to time in the interval $[t_0, t_f]$ e.g.:



Note that we are calculating *functions* of time, rather than point (steady-state) values. These are commonly referred to as the 'trajectories' of the variables. Again, these functions need not be continuous or differentiable. Change any of the above information and you have a new scenario — a different dynamic simulation experiment, with different solution trajectories.

DYNAMIC SIMULATION TOOLS

A *dynamic simulation tool* is some form of computer software that supports the engineer in building dynamic models and solving the resulting simulation problems (e.g., something more than coding everything from scratch in FORTRAN).

There are really three features of a dynamic simulation tool that are of interest:

(a) how is the model developed and coded?

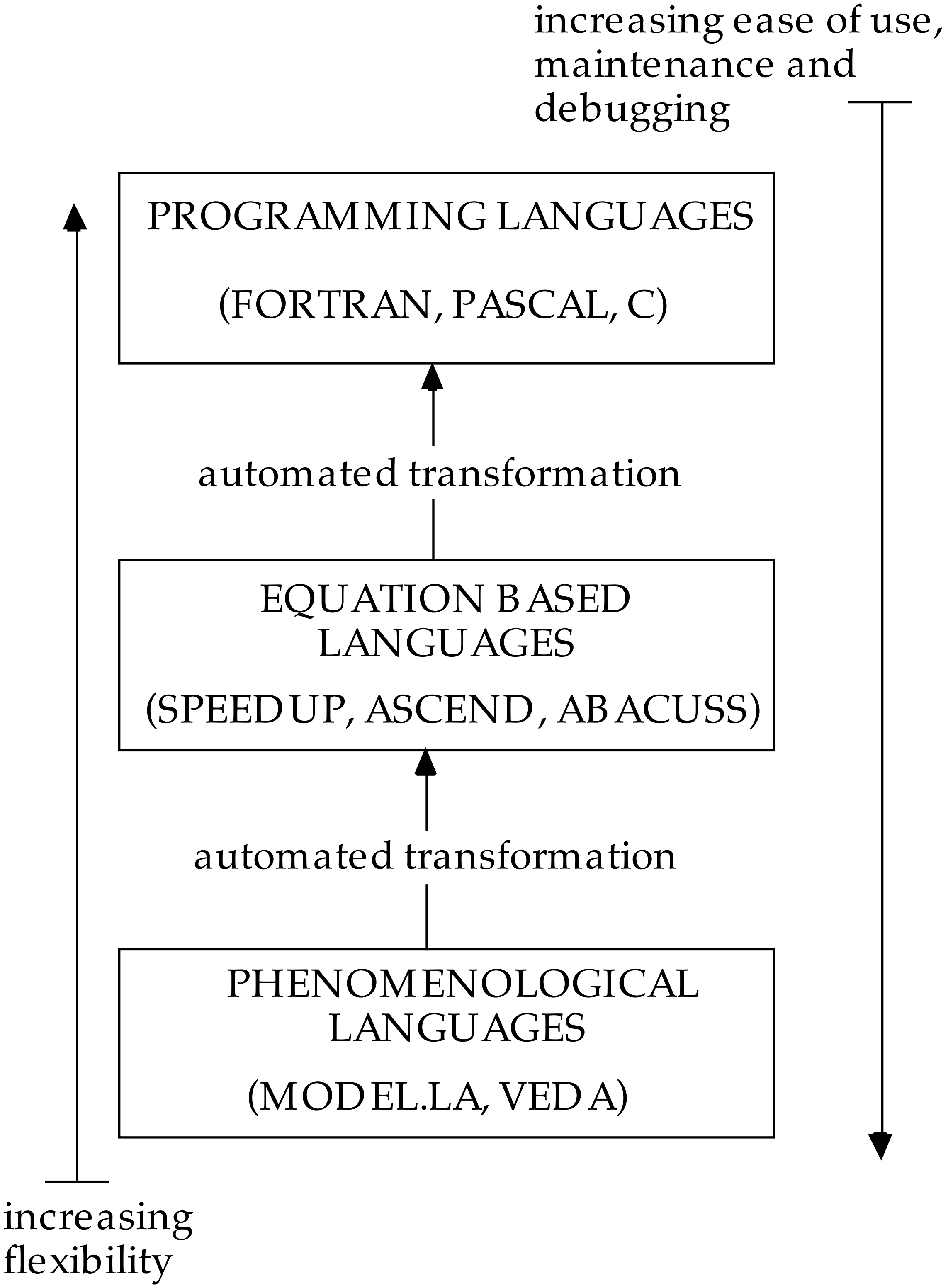
(b) how are individual simulation experiments specified?

(c) how is numerical solution of the simulation experiment conducted?

. . . most dynamic simulation tools offer a mixture of different features in each category.

Note: in the early days *analogue* computers were widely used for dynamic simulation of chemical processes — I don't think anyone is doing this any more! All these remarks are concerning dynamic simulation on *digital* computers.

CODIFYING THE MODEL



increasing ease of use, maintenance and debugging

PROGRAMMING LANGUAGES

(FORTRAN, PASCAL, C)

automated transformation

EQUATION BASED LANGUAGES

(SPEEDUP, ASCEND, ABACUSS)

automated transformation

PHENOMENOLOGICAL LANGUAGES

(MODEL.LA, VEDA)

increasing flexibility

PROGRAMMING LANGUAGES

The old fashioned approach! Basically, the user codes a subroutine in some general-purpose programming language (usually FORTRAN) that returns some specific information concerning the process model.

All numerical methods for solving ordinary differential equations require either:

- *function evaluations* — consider system (ODE): a function evaluation will evaluate the vector of functions $\underline{f}$ given values for $\underline{x}$ and t (e.g., evaluate $\underline{\dot{x}}$ given $\underline{x}$ and t)
- *residual evaluations* — a more general approach: the model equations are rearranged to the general form:

  $$\underline{g}(\underline{\dot{x}},\underline{x},t)=\underline{0}$$

  . . . and the vector of functions $\underline{g}$ (residuals) is evaluated given values for $\underline{\dot{x}}$, $\underline{x}$ and t. Note that the residuals will only equal the zero vector if the set of values $\{\underline{\dot{x}},\ \underline{x},\ t\}$ are a solution of the differential equations.

Consider the model of a buffer tank from section I — overleaf, I have coded a FORTRAN subroutine to return the residuals given values for the unknowns. This subroutine would (for example) be called repeatedly during an iterative process to find variable values that satisfy the differential equations.

RESIDUAL SUBROUTINE — EXAMPLE

```
      SUBROUTINE BUFFER(X,XDOT,U,T,RES)
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION X(5), XDOT(5), U(2), RES(5)
C
C      Definition of unknown variables:
C
C    X(1) mass of fluid in vessel       [kg]
C    X(2) volume of fluid in vessel          [m3]
C    X(3) volumetric flow out of vessel [m3 s-1]
C    X(4) pressure at bottom of vessel  [N m-2]
C    X(5) liquid level in vessel        [m]
C    XDOT(i) time derivative of X(i)
C
C      Definition of input variables:
C
C    U(1) volumetric flow into vessel   [m3 s-1]
C    U(2) atmospheric pressure          [N m-2]
C
C      Definition of time invariant parameters:
C
C    AREA cross sectional area of vessel     [m2]
C    DENS fluid density                 [kg m-3]
C    COEF loss coefficient
C    G    gravitational acceleration    [m s-2]
C
      AREA = 1.767
      DENS = 1000
      COEF = 0.01
      G = 9.81
C
C       Evaluate residuals given X, XDOT and U
C
      RES(1) = XDOT(1) - (DENS*U(1) - DENS*X(3))
      RES(2) = DENS*X(2) - X(1)
      RES(3) = X(5)*AREA - X(2)
      RES(4) = X(4) - (U(2) + DENS*G*X(5))
      RES(5) = X(3) - (COEF*SQRT(X(4) - U(2)))
      RETURN
      END
```

PROGRAMMING LANGUAGES

In addition, other subroutines can be written that return:

- *Jacobian evaluations* — many numerical methods work better if it is possible to evaluate the matrix of partial derivatives of the functions with respect to the dependent variables analytically. The user can derive expressions for the partial derivatives and then write an appropriate subroutine (extremely tedious and prone to error).

  In fact, there are now *automatic differentiation* codes being quite widely used (see Griewank *et al*., 1991). These codes can take a FORTRAN residual evaluator as input, and automatically generate a FORTRAN routine coding the Jacobian evaluation.

- *incidence information* — for medium to large systems of equations (>20) only a few of the variables will appear in each equation — *sparse equation system.* Therefore, most of the Jacobian entries (partial derivatives) will be zero at all times. Incidence information encapsulates this — an explicit statement of which variables appear in each equation. This can be used to great advantage in numerical Jacobian evaluations and in solving systems of linear equations (*sparse linear algebra*). Again, this is tedious and error prone.

. . . as you can imagine, these become monumental tasks to do by hand when dealing with models composed of 1,000's of equations!

PROGRAMMING LANGUAGES

Advantages:

- anyone can do it — you don't need to buy a dynamic simulator!
- the ultimate in flexibility — can express the most general functional forms.

Disadvantages:

- tedious, time consuming, error prone, not a natural way to represent equations, etc. . . Probably the biggest problem is human error - one Jacobian entry that is calculated incorrectly can cause a numerical method to fail to converge.
- difficult to separate model from the mechanics of its solution — model locked into a single application (calculation).
- mathematical information concerning the model is lost — e.g., the only access we have to the model is the residuals given values for the variables.
- inflexible — if you want to change the model (or even the set of input variables) must code a completely new model.
- no way to check automatically if the model has been coded correctly
- requires expertise in modelling, computer science and mathematics — expert users!

EQUATION BASED LANGUAGES

The simulation software supports an equation based programming language with which to code process models.

These languages are:

- high level and specialized to the task of specifying equation based models.
- **declarative** rather than **procedural** — the purpose is to specify the functional form of the model (e.g., a system of equations) and is *not* a series of statements to be executed in a sequence.

. . . currently, this is the state of the art.

This high level representation can be automatically translated into FORTRAN subroutines to interface to standard numerical solvers.

Procedural: $x := x$^2 + 1 ;

Compute $x^2 + 1$ based on the current value of memory location x and assign the result to memory location x.

Declarative: $x = x$^2 + 1 ;

Find solution of quadratic equation $x^2 + 1 - x = 0$.

EQUATION BASED LANGUAGES — EXAMPLE

```
MODEL Buffer_Tank

###########################################################
#
#  ABACUSS model of a buffer tank - Paul I. Barton 7/27/94
#
#  NOTE: anything following a # symbol is a comment
#
###########################################################

PARAMETER
  Cross_Sectional_Area                    AS REAL
  Grav_Acceleration                       AS REAL
  Loss_Coefficient                        AS REAL
  Density                                 AS REAL

VARIABLE
  Mass_Holdup                             AS Mass
  Liquid_Volume                           AS Volume
  Flow_In, Flow_Out                       AS Volumetric_Flow
  Atm_Press, Press                        AS Pressure
  Liquid_Level                            AS Length

STREAM
  Inlet  : Flow_In, Atm_Press             AS MainStream
  Outlet : Flow_Out, Press                AS MainStream

EQUATION

 # Mass balance - $ is the time differential operator
 $Mass_Holdup = Density*Flow_In - Density*Flow_Out ;

 # relate liquid volume and mass
 Density*Liquid_Volume = Mass_Holdup ;

 # relate liquid level and volume
 Liquid_Level*Cross_Sectional_Area = Liquid_Volume ;

 # hydrostatic pressure
 Press = Atm_Press + Density*Grav_Acceleration*Liquid_Level ;

 # flow/pressure relationship
 Flow_Out = Loss_Coefficient*(Press - Atm_Press)^0.5 ;

END # model buffer_tank
```

EQUATION BASED LANGUAGES

Advantages:

- declarative — model is decoupled from mechanics of solution. No assumption concerning which variables are inputs is made during model coding.
- incidence information and Jacobian matrix can be generated automatically — using symbolic differentiation or automatic differentiation technology (automatic, fast, and error free!)
- all mathematical information is retained — computer stores complete functional form of model.
- tools that check if a system of equations is mathematically well posed can be applied automatically (see appendix).
- can embed automatically more sophisticated solution strategies: e.g., combined discrete/continuous simulation, dummy derivative method for high index systems. FORTRAN based approach will just solve an IVP with smooth inputs unless user adds code for discontinuity handling herself.

Disadvantages:

- although high flexibility, advances are still needed to express more general functional forms.
- user must derive equations => expert users! - but, user only has to focus on the physics and chemistry.
- model documentation only as comments.

PHENOMENOLOGICAL LANGUAGES

This is really only in the realm of (not very mature) academic research. . .

Problem: the following paradox:

(a) deriving models from first principles in terms of equations is a difficult and skilled task requiring experts - error prone!

(b) it is difficult (or even impossible) to standardize dynamic models[3] — each study of a new system will require some model development activity.[4]

Idea: can we develop dynamic models in terms of higher level concepts than equations?

- develop a phenomena based modelling methodology
- codify this as a software tool
- and then generate the equations automatically from this higher level description

[3] in contrast to steady-state simulation where by and large (reactors being an exception) standardization has been achieved.

[4] this is, in my opinion, a major reason why the equation based langauges have become so popular

PHENOMENOLOGICAL LANGUAGES

The advantages claimed of such an approach are:

- the engineer can work directly with the principles of physics, chemistry and biology rather than equations — this is perceived as "easier" (but a complete set of concepts is required and the user must learn the phenomena based modelling methodology).
- guarantees can be made that the model is correct for the assumptions made (?)
- assumptions, domains of validity, etc., can be generated and documented automatically — qualitative information forms part of the model.

Several ideas have emerged in the recent literature, for example:

(a) specify the phenomena of interest => balance equations required; specify the assumption => terms in these balance equations (Stephanopoulos *et al*., 1990a; 1990b)

(b) specify a set of thermodynamic phases exchanging mass, energy and/or momentum. Connections between phases are characterized by the transport phenomena leading to transfer of mass, energy and/or momentum (Perkins *et al*., 1994; Marquardt 1994)

. . . research continues.

DYNAMIC SIMULATION TOOLS

The software tools currently available roughly fall into the following categories:

(a) subroutine libraries coded in a general-purpose programming language (typically FORTRAN):

- models coded as subroutines - some library models
- series of subroutines to conduct numerical integration, coordinate models, report results, etc.
- at *least* use a standard numerical integration code (Perkins, 1986) — e.g., Harwell Subroutine Library, IMSL, ODEPACK, DASSL, DSL48S.

Examples: DYFLO (Franks, 1972), DAEPACK (Tolsma and Barton, 2000).

(b) continuous system simulation languages (CSSLs) — typically derived from the CSSL'67 (Strauss, 1967) standard:

- equation based simulation language
- extensively used by control engineers for many years.
- only solve models in state space form (i.e. ODEs but *not* DAEs)
- not really suitable for large-scale simulations.
- no chemical engineering specific concepts (e.g., multi-component streams, physical properties, unit operation libraries).

Examples: ACSL, SIMUSOLV, SIMULINK (Matlab), etc.

DYNAMIC SIMULATION TOOLS

(c) operator training tools — typically vendor will develop plant model, simulation and interface.

(d) dynamic flowsheeting packages[5] — e.g., dynamic simulation tools specifically designed for chemical process simulation:

- some have equation based simulation languages
- facilities to solve large-scale differential-algebraic models.
- support flowsheet paradigm (e.g., concept of streams, unit operations, etc.)
- physical properties, unit operation, model libraries, etc.

Examples: ACM, gPROMS, ABACUSS II, DIVA, HYSYS

Most of the simulation tools above use simultaneous numerical solution methods — e.g., the differential equations are assembled together and solved simultaneously with a single algorithm.

However, in the development of dynamic flowsheeting packages two approaches have emerged: *modular* and *direct*. These are introduced in the following pages.

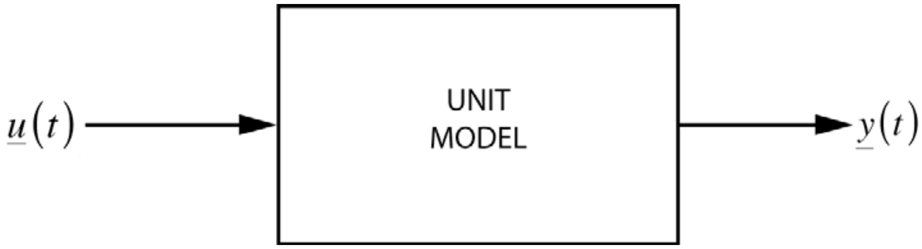
[5] see Marquardt (1991) for a comprehensive list of software tools and their current availability.

MODULAR INTEGRATORS

Effectively, this is an attempt to extend the sequential modular approach to steady-state process simulation to dynamic simulation.

Each unit operation model (black box model):

- is coded with its own numerical integration routine (any suitable method)
- can solve for the time variation of its outputs given the time variation of its inputs:



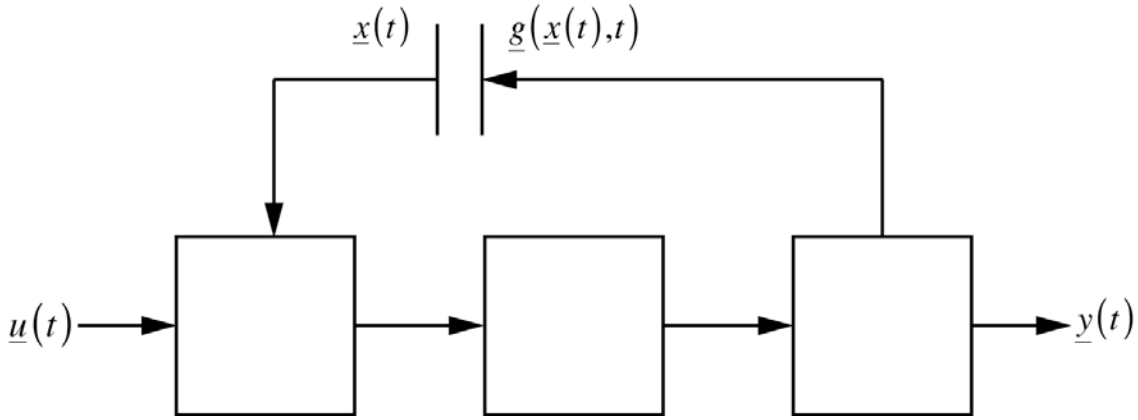
So, given the input trajectories each unit model can be integrated independently over a time horizon TH (the numerical algorithms embedded may take several steps in this time horizon).

Advantage: can tailor algorithm to a particular unit operation - for example, rapid transients may be localized to a single unit operation.

Problem: must coordinate solution of individual models if there is feedback of material *or* information in the flowsheet - e.g. inputs of some unit operations not known in advance. Need to develop a *coordinator algorithm*.

MODULAR INTEGRATORS

Solution: tearing concept - guess the time variation of the recycle streams and iterate until the *function* guessed matches the function calculated.



Problem: find $\underline{x}(t), t \in [0,TH]$ such that:

$$\underline{x}(t) = \underline{g}(\underline{x}(t),t) \qquad \forall t \in [0,TH]$$

Note that in dynamic simulation we are solving for a *function* rather than a single value. Severe problems with:

- integration error control
- stability
- converging recycles (choice of norm to compare functions)
- extrapolation of variable trajectories
- choice of time horizon
- treatment of discontinuities
- computational expense of co-ordination

Examples: GEPURS, FLOWPACK II, PROSIM

DIRECT INTEGRATORS

Examples: SPEEDUP, DIVA, DYNSIM, ASCEND, gPROMS, ABACUSS, POLYRED:

- all equations assembled and integrated simultaneously with a single algorithm
- guarantees for stability, and integration error can be controlled automatically — theoretical guarantees on the accuracy of solution ($\therefore$ errors = modelling errors)
- efficient general-purpose codes for large-scale problems exist
- can use equation based modelling language
- in practice, more efficient for accurate simulation of processing systems!

MATHEMATICAL FORMULATION OF DYNAMIC SIMULATION PROBLEMS

A simulation tool will be coded to support a specific mathematical formulation for dynamic simulation experiments. In principle, the most general formulation is desirable. However, from a practical point of view we must work within the constraints of current technology.

Assuming we only have time as an independent variable, a dynamic model together with a choice of input variables will typically give rise to one of the following (similar) mathematical problems:

(A) systems of explicit first order ordinary differential equations (ODEs):

$$\dot{\underline{x}} = \underline{f}(\underline{x}, t) \tag{1}$$

which is the mathematical formulation supported by the CSSL-type simulation languages.

More specifically, it is possible to introduce auxiliary algebraic equations of the form:

$$\begin{aligned} y_1 &= g_1(\underline{x}) \\ y_2 &= g_2(\underline{x}, y_1) \\ &\vdots \\ y_m &= g_m(\underline{x}, y_1 \cdots y_{m-1}) \end{aligned} \tag{2}$$

MATHEMATICAL FORMULATION OF DYNAMIC SIMULATION PROBLEMS

In words: algebraic equations are acceptable provided a suitable calculation sequence can be found so that each algebraic variable can be determined explicitly given $\underline{x}$ and any previously calculated y_i - i.e. the algebraic variables can be eliminated by direct substitution of the algebraic equations into the ordinary differential equations.

Note that most of the models derived in section I do not conform to this formulation without some form of manipulation (in general, differentiation with respect to time).

(B) systems of differential-algebraic equations (DAEs - coupled systems of ordinary differential and algebraic equations):

$$\underline{F}\left(\dot{\underline{z}},\underline{z},t\right)=\underline{0} \qquad (3)$$

where $\underline{z}\in\mathbf{R}^n$, $\underline{F}:\mathbf{R}^n\times\mathbf{R}^n\times\mathbf{R}\mapsto\mathbf{R}^n$ – this general formulation is known as the *fully implicit* form.

Note: what distinguishes (3) from a system of implicit ODEs (IODE) is the fact that the Jacobian of $\underline{F}$ with respect to $\dot{\underline{z}}$ is singular everywhere. In other words, equations (3) cannot be solved for $\dot{\underline{z}}$ given $\underline{z}$ and t.

MATHEMATICAL FORMULATION OF DYNAMIC SIMULATION PROBLEMS

Chemical engineering models frequently give rise to the *semi-explicit* form of differential-algebraic equations, which provides useful insights into the structure of DAEs:

$$\begin{aligned} \underline{f}\left(\underline{\dot{x}}, \underline{x}, \underline{y}, t\right) &= \underline{0} \\ \underline{g}(\underline{x}, \underline{y}, t) &= \underline{0} \end{aligned} \tag{4}$$

where we can categorize:

$\underline{x} \in \mathbf{R}^n$	(differential variables)
$\underline{y} \in \mathbf{R}^m$	(algebraic variables)
$\underline{\dot{x}} \in \mathbf{R}^n$	(time derivatives of the differential variables)
$\underline{f} : \mathbf{R}^n \times \mathbf{R}^n \times \mathbf{R}^m \times \mathbf{R} \mapsto \mathbf{R}^n$	(differential equations)
$\underline{g} : \mathbf{R}^n \times \mathbf{R}^m \times \mathbf{R} \mapsto \mathbf{R}^m$	(algebraic equations)

...so we have $(n+m)$ unknowns and $(n+m)$ equations.

Note that (4) is just a specialization of (3): $\underline{z}$ would be just a $n+m$ vector composed of the components of $\underline{x}$ and $\underline{y}$.

The Jacobian of equations (4) with respect to the time derivatives of the unknowns is clearly rank deficient everywhere ($\underline{\dot{y}}$ do not appear anywhere).

MATHEMATICAL FORMULATION OF DYNAMIC SIMULATION PROBLEMS

In the literature some authors advocate that two other (more restricted) forms are appropriate for chemical engineering applications, the *linear semi-explicit* form:

$$\begin{aligned} \dot{\underline{x}} &= f\left(\underline{x},\underline{y},t\right) \\ \underline{g}\left(\underline{x},\underline{y},t\right) &= \underline{0} \end{aligned} \tag{5}$$

or the *linear implicit* form:

$$\underline{\underline{B}}(\underline{x},t)\dot{\underline{x}} = \underline{f}(\underline{x},t) \tag{6}$$

These arguments seem rather pointless because reliable general-purpose codes for the direct solution of the general formulation (3) are widely available (*if* index = 1). Further, formulation (5) is clearly unsuitable for some of the models derived in section I of these notes.

However, theoretical treatment of both (5) and (6) is easier.

WHY USE DAEs?

(a) it is the most natural form in which to derive the model — e.g. modelling of physico-chemical phenomena in chemical processes gives rise to DAEs (see section I).

Advantage: user can just submit model directly to the computer without further manipulation.

(b) derivation of ODEs requires (extremely) tedious differentiation and algebraic manipulation of the equations. To illustrate this, consider the semi-explicit form (4):

$$\underline{f}(\dot{\underline{x}},\underline{x},\underline{y},t)=\underline{0} \qquad (7)$$

$$\underline{g}(\underline{x},\underline{y},t)=\underline{0} \qquad (8)$$

Take the total differential and divide by *dt* to yield the time derivatives of these equations:

$$\dot{\underline{f}}=\frac{\partial \underline{f}}{\partial \dot{\underline{x}}}\ddot{\underline{x}}+\frac{\partial \underline{f}}{\partial \underline{x}}\dot{\underline{x}}+\frac{\partial \underline{f}}{\partial \underline{y}}\dot{\underline{y}}+\frac{\partial \underline{f}}{\partial t}=\underline{0} \qquad (9)$$

$$\dot{\underline{g}}=\frac{\partial \underline{g}}{\partial \underline{x}}\dot{\underline{x}}+\frac{\partial \underline{g}}{\partial \underline{y}}\dot{\underline{y}}+\frac{\partial \underline{g}}{\partial t}=\underline{0} \qquad (10)$$

and provided that the matrix: $\begin{bmatrix} \frac{\partial \underline{f}}{\partial \dot{\underline{x}}} & 0 \\ \frac{\partial \underline{g}}{\partial \underline{x}} & \frac{\partial \underline{g}}{\partial \underline{y}} \end{bmatrix}$

WHY USE DAEs?

is nonsingular, then (7) and (10) can be solved simultaneously for $\dot{\underline{x}}$ and $\dot{\underline{y}}$ given $\underline{x}$, $\underline{y}$ and t. Hence the system (7) and (10) is a system of implicit ODEs (i.e. the relevant Jacobian is nonsingular).

For example, consider just the energy balance and enthalpy relations in our CSTR model:

$$\frac{dH}{dt} = F_{IN}\rho_{IN}h_{IN} - F_{OUT}\rho_{OUT}h_{OUT} + \dot{Q}$$
$$H = Mh_{OUT}$$
$$h_{OUT} = h(T, C_A, C_B)$$

...and derive ODEs by differentiation of the algebraic equations:

$$\frac{dH}{dt} = h_{OUT}\frac{dM}{dt} + M\frac{dh_{OUT}}{dt}$$
$$\frac{dh_{OUT}}{dt} = \left(\frac{\partial h}{\partial T}\right)_{C_A, C_B}\frac{dT}{dt} + \left(\frac{\partial h}{\partial C_A}\right)_{T, C_B}\frac{dC_A}{dt} + \left(\frac{\partial h}{\partial C_B}\right)_{T, C_A}\frac{dC_B}{dt}$$

- these are not yet in state space form — in general, algebraic manipulation may not be able to yield the explicit ODE form.
- total differentials of physical property models are particularly tedious to derive.

In general, derivation of ODEs from DAEs requires one (or more) differentiations of the equations (or subsets) with respect to time.

WHY USE DAEs?

(c) the solution set of the ODEs derived by differentiation is larger than the solution set of the original DAEs — the solution to the derivative of an equation is the original equation plus an arbitrary constant.

Example: the numerical integration of the ODEs above will treat the enthalpy holdup and temperature as independent variables — however, we know they are not independent but related by the physical property model.

Typically, we can ensure the initial conditions (e.g. $H(t_0)$ and $T(t_0)$) are consistent — but we cannot avoid numerical errors (i.e. under discretization) building up during integration that will lead to $H(t)$ and $T(t)$ at some later time that are not consistent with the physical property model. This may, or may not, be disastrous (see below).

Direct integration of the DAEs ensures that the algebraic equations are satisfied at each time step — hence the algebraic variables will remain consistent with the differential variables.

WHY USE DAEs?

(d) the numerical integration of a minimal set of independent variables (e.g. the differential variables in a DAE system) is more consistent with the results of classical thermodynamics - e.g. only a certain minimal set of independent quantities is required to fully define the 'state' of a simple system (Modell and Reid, 1983).

DEGREE OF FREEDOM ANALYSIS

Process models are typically *under determined* systems of equations - there are more variables describing the behaviour of the system than equations relating them. Degree of freedom analysis is concerned with identifying a subset of variables to assign values to (or *specify*), so that the equations become *fully determined* in the remaining *unknown* (or dependent) variables - and thus can be solved to determine a set of unique values for these unknowns.

A necessary condition that a system of NEQ equations is fully determined in the NUK unknown variables is that:

$$NEQ \geq NUK$$

So, clearly, we must *at least* specify values for a sufficient number of variables so that the number of unknowns becomes equal to the number of equations.

If the situation arises where so many specifications have been made that there are *more* equations than unknowns (i.e. $NEQ > NUK$), the extra equations are either:

- **redundant** - i.e. they add no extra information, so they do not constrain the solution further (example: adding an overall mass balance to a flowsheet model where we already have mass balances for each unit)
- **inconsistent** - i.e. there is no solution that will satisfy all the equations.

DEGREE OF FREEDOM ANALYSIS

Bearing this in mind, we will only accept a model for solution if :

$$NEQ = NUK$$

...and degree of freedom analysis is concerned with selecting a subset of variables to specify so that this condition is satisfied.

When setting up a dynamic simulation experiment we have to worry about degree of freedom analysis twice:

- first derive a fully determined system of DAEs from the under determined model by selecting a set of input variables (in fact, we are assigning known functions of time rather than values to the input variables)
- next (given a fully determined system of DAEs) establish the degrees of freedom available to specify the initial state of the system - e.g. the number of side conditions required at $t = t_0$.

...details of each follow.

INPUT SPECIFICATIONS - GENERAL FORMULATION

Typically, chemical engineering models give rise to under determined systems of DAEs — we have degrees of freedom to satisfy before a fully determined system that can be submitted to the computer.

Process model:

$$\underline{f}\left(\dot{\underline{x}},\underline{x},\underline{y},\underline{u},t\right)=\underline{0}$$

where:

$\underline{x}\in\mathbf{R}^{n}$ (differential variables)

$\underline{y}\in\mathbf{R}^{m}$ (algebraic variables)

$\underline{u}\in\mathbf{R}^{l}$ (input variables)

$\underline{f}:\mathbf{R}^{n}\times\mathbf{R}^{n}\times\mathbf{R}^{m}\times\mathbf{R}^{l}\times\mathbf{R}\mapsto\mathbf{R}^{n+m}$

This is just a restatement of the general formulation (3) above — here, the set of variables has merely been partitioned into *differential*, *algebraic*, and *input* variables according to the definitions:

- input - assigned a known function of time
- differential - whose time derivative appears
- algebraic - whose time derivative do *not* appear

Satisfying the degrees of freedom is merely selecting which subset of l variables will belong to the vector $\underline{u}$, and assigning functions of time to these variables.

INPUT SPECIFICATIONS - GENERAL FORMULATION

This leads to the fully determined DAE system:

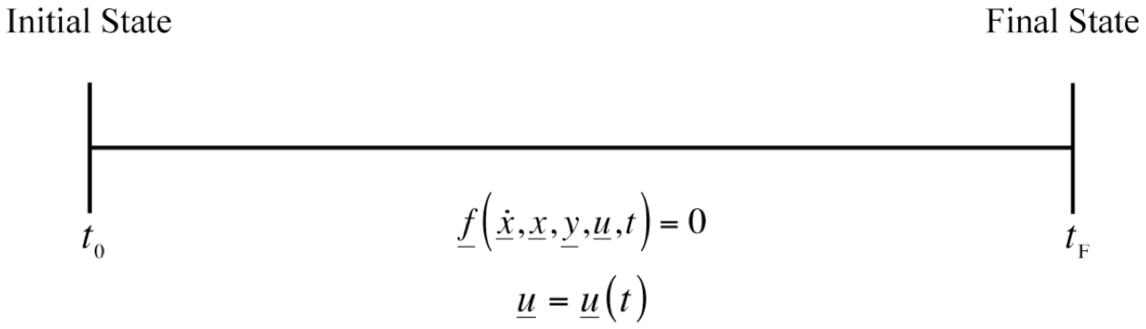
$$\underline{f}(\underline{\dot{x}}, \underline{x}, \underline{y}, \underline{u}, t) = 0$$

$$\underline{u} = \underline{u}(t)$$

Alternatively, we could add extra equations to the model, reducing the number of input specifications required by one for each non redundant equation added.

SOLVING DYNAMIC SIMULATIONS

Solve initial value problem in differential-algebraic equations in the interval $[t_0, t_f]$:

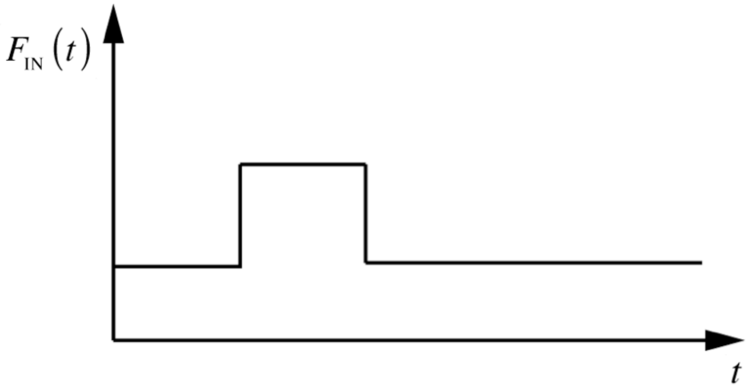

Initial State
Final State
t_0
$\underline{f}\left(\dot{\underline{x}}, \underline{x}, \underline{y}, \underline{u}, t\right) = 0$
$\underline{u} = \underline{u}(t)$
t_F

This requires solution of three sub problems:

1. Defining and solving for the initial state of the system at t_0.

2. Numerical integration of the DAEs from t_0 to t_f.

3. Discontinuity handling — for example, correct processing of discontinuities in the functions assigned to the input variables (also their time derivatives):



Details of each follow. . .

DEFINING THE INITIAL STATE OF THE SYSTEM

Mathematically: defining the side conditions for the initial value problem, and utilizing these to determine initial values for the unknowns.

For explicit ODE systems the side conditions are usually specified by providing initial values for $\underline{x}$:

$$\underline{x}(t_0) = \underline{x}_0$$

$$\Rightarrow \dot{\underline{x}}(t_0) = \underline{f}(\underline{x}(t_0), t_0)$$

In this case, calculating the initial state is trivial and requires only a function evaluation - i.e. $\underline{x}(t_0)$ independent.

However, for most dynamic simulation experiments a more general approach is both desirable and possible.

CONSISTENT INITIALIZATION

Consistent initialization: find a consistent set of initial values for the variables $\underline{x}(t)$ and $\dot{\underline{x}}(t)$ at $t = t_0$:

Denoted by: $\left\{\underline{x}(t_0), \dot{\underline{x}}(t_0)\right\}$

A set of consistent initial values defines one possible initial state for the system.

Idea: in general, undertake a consistent initialization calculation in which the model equations at $t = t_0$, e.g.:

$$\dot{\underline{x}}(t_0) = \underline{f}(\underline{x}(t_0), t_0)$$

are solved for the initial state of the system.

The initial values $\left\{\underline{x}(t_0), \dot{\underline{x}}(t_0)\right\}$ will then be consistent with the model equations.

CONSISTENT INITIALIZATION

Example: many dynamic simulations involve starting from the system from steady-state and then introducing a disturbance (e.g. step change, ramp, etc.).

In the ODE example, an initial state of *steady-state* is equivalent to defining:

$$\underline{\dot{x}}(t_0) = \underline{0}$$

...substitute this specification into the ODEs (system at $t = t_0$) to yield:

$$\underline{0} = \underline{f}(\underline{x}(t_0), t_0)$$

and solve the resulting system of nonlinear algebraic equations to find consistent initial values for $\underline{x}(t_0)$.

This assumes that the model has a steady-state solution - not always the case, e.g. a batch reactor.

Note that this is entirely equivalent to a steady-state simulation. In fact, steady-state simulation is really only a subset of dynamic simulation: it is merely one possible consistent initialization calculation.

DEGREES OF FREEDOM FOR CONSISTENT INITIALIZATION OF ODEs

For ODEs, degrees of freedom for the initialization calculation are clear.

Given a fully determined ODE system:

n equations $\quad\quad \underline{\dot{x}}(t_0) = \underline{f}(\underline{x}(t_0), t_0)$

$2n$ variables $\quad\quad \left\{\underline{x}(t_0), \underline{\dot{x}}(t_0)\right\}$

=> n degrees of freedom

Examples of possible specification sets for consistent initialization:

a) $\left\{\underline{x}(t_0)\right\}$

b) $\left\{\underline{\dot{x}}(t_0)\right\}$

c) or some mixture of values for the variables and their time derivatives

DEGREES OF FREEDOM FOR CONSISTENT INITIALIZATION OF DAEs

In direct analogy, to define the initial state of a fully determined system of DAEs we need to find a set of consistent initial values for the variables $\underline{x}(t)$, $\underline{\dot{x}}(t)$ and $\underline{y}(t)$ at $t = t_0$, e.g.:

$$\left\{\underline{x}(t_0), \underline{\dot{x}}(t_0), \underline{y}(t_0)\right\}$$

Clearly, a *necessar*y condition for a set of consistent initial values is that the model equations are satisfied at $t = t_0$, e.g.:

$$\underline{f}(\underline{\dot{x}}(t_0), \underline{x}(t_0), \underline{y}(t_0), \underline{u}, t_0) = \underline{0}$$
$$\underline{u} = \underline{u}(t_0)$$

Assuming that this necessary condition is also sufficient (*not* always the case - see discussion of high index problems), degree of freedom analysis gives:

$n + m$ equations $\underline{f}(\underline{\dot{x}}(t_0), \underline{x}(t_0), \underline{y}(t_0), \underline{u}(t_0), t_0) = \underline{0}$

$2n + m$ variables $\quad \left\{\underline{x}(0), \underline{\dot{x}}(0), \underline{y}(0)\right\}$

=> n degrees of freedom

Note the degrees of freedom is equal to the number of differential variables n.

DEGREES OF FREEDOM FOR CONSISTENT INITIALIZATION OF DAEs

So, we can choose a n membered subset of variables out of $\left\{ \underline{x}(t_0), \dot{\underline{x}}(t_0), \underline{y}(t_0) \right\}$ to assign initial values to. This is the formulation supported by SpeedUp.

These values (substituted into the model equations) will fully define the initial state of the system.

Notes:

a) consistent initialization of DAEs requires the solution of large sparse systems of nonlinear algebraic equations from poor initial guesses - in principle, we can use any technique already developed for steady-state simulation.

b) as with steady-state simulation, not all subsets of variables and not all choices of values lead to a well-posed problem (functionally nonsingular equations).

c) numerical integration of DAEs should always be started from consistent initial values - otherwise the results can be garbage and/or the integration can fail. The numerical integration algorithms most commonly used can generate a solution from arbitrary (possibly inconsistent) initial values - in fact, standard DAE solver routines do not usually have facilities for the consistent initialization calculations described above! (see Kroner, *et al.*, 1992)

DEGREES OF FREEDOM FOR CONSISTENT INITIALIZATION OF DAEs

d) we will also discuss the situation when the model equations are not necessary and sufficient to define consistent initial values - in general, we have *at most n* degrees of freedom to specify the initial state of the system.

CONSISTENT INITIALIZATION - EXAMPLES

Consider possible initial states for our well stirred reactor system:

a) operate the reactor as a CSTR and start the simulation form steady-state at $t = 0$.

   Define steady-state by setting the time derivatives of all the differential variables to zero (n specifications):

   $$\dot{M}(0) = 0, \dot{N}_A(0) = 0, \dot{N}_B(0) = 0, \dot{H}(0) = 0$$

   ...and solve the model equations for $\mathbf{x}(0)$ and $\mathbf{y}(0)$. In this case, the equations need to be solved simultaneously.

b) operate the reactor as a batch reactor - define the initial charge to the vessel - e.g. at $t = 0$:

   $$M(0) = 1000, N_A(0) = 100, N_B(0) = 0, H(0) = 1E7$$

   so, given $\mathbf{x}(0)$ solve equations $\dot{\mathbf{x}}(0)$ and $\mathbf{y}(0)$. Note that the equations are much easier to solve for this choice of initial condition!

CONSISTENT INITIALIZATION - EXAMPLES

c) for a batch reactor, we typically do not know the enthalpy of a charge - after all this is a value related to some arbitrary zero energy reference state.

For example, a more realistic operating policy would specify the charge as 1m^3 of solvent with a specified concentration of A at 300K:

$$V(0) = 1, C_A(0) = 1, C_B(0) = 0, T(0) = 300$$

and we solve the equations for the remaining variables - $\mathbf{x}(0)$, $\dot{\mathbf{x}}(0)$ and remaining $\mathbf{y}(0)$.

Note that it is often most convenient to specify the initial condition in terms of values for a subset of the algebraic variables - hence the immense flexibility of this approach.

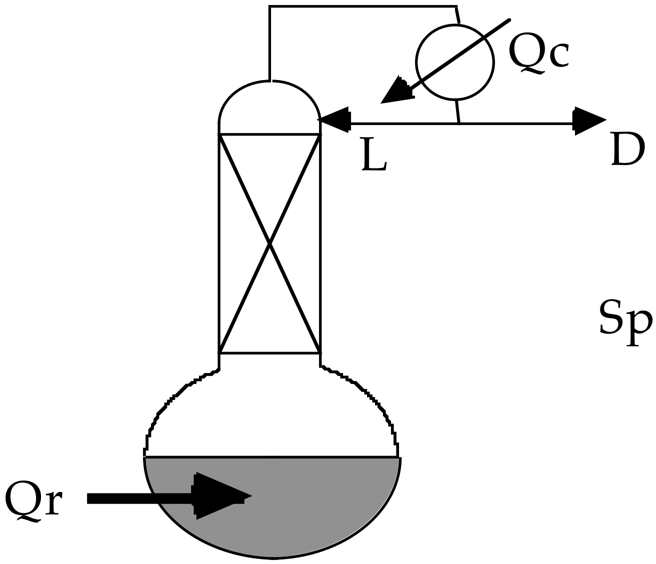
Exercise: why is the specification of:

$$\{V(0), N_A(0), C_A(0), T(0)\}$$

invalid?

CONSISTENT INITIALIZATION - GENERAL FORMULATION

Example: consider a batch distillation column:



Species: A, B, C

Typical operating policy:

a) charge the reboiler with mixture of A, B and C

b) apply steam to reboiler and bring to steady-state at total reflux $(D(t)=0)$

c) draw cut rich in A overhead — $D(t)>0$

d) draw cut rich in B overhead — $D(t)>0$

Simulation experiment: define initial state as steady-state at total reflux and simulate the sequence of cuts.

Note: the model required to simulate start-up is much more complex - neglect this period.

CONSISTENT INITIALIZATION - GENERAL FORMULATION

How to define steady-state at total reflux?

a) if we specify steady-state: $\dot{\mathbf{x}}(0) = 0 \Rightarrow$ *badly posed problem* - no way to determine the initial extent of material in the reboiler.

b) define the initial condition as:

$$N_{T,i} = N_{R,i}(0) + \sum_{j=1}^{NTray} N_{j,i}(0) \qquad \forall i = 1 \ldots NC$$

$$\dot{N}_{i,j}(0) = 0 \qquad \forall j = 1 \ldots NTray \quad \forall i = 1 \ldots NC$$

where

$N_{T,i}$ = amount of species i in charge (known)

$N_{R,i}$ = holdup of species i in reboiler

$N_{j,i}$ = holdup of species i on stage j

i.e. all trays at steady-state, holdup in reboiler determined by an overall mass balance on column.

Note that this time we have defined an initial condition in terms of a set of equations rather than an assignment of values.

CONSISTENT INITIALIZATION - GENERAL FORMULATION

The general formulation for consistent initialization requires the solution of the following system of nonlinear algebraic equations for the unknowns $\{\dot{\mathbf{x}}(t_0), \mathbf{x}(t_0), \mathbf{y}(t_0)\}$:

$$\mathbf{f}(\dot{\mathbf{x}}(t_0), \mathbf{x}(t_0), \mathbf{y}(t_0), \mathbf{u}(t_0), t_0) = \mathbf{0}$$
$$\mathbf{c}(\dot{\mathbf{x}}(t_0), \mathbf{x}(t_0), \mathbf{y}(t_0), \mathbf{u}(t_0), t_0) = \mathbf{0}$$

where:

$$\mathbf{c} : \mathbf{R}^n \times \mathbf{R}^n \times \mathbf{R}^m \times \mathbf{R}^l \times \mathbf{R} \mapsto \mathbf{R}^n$$

and the functions **c** may be value assignments or general functional relationships.

Again, this system of equations must be functionally nonsingular.

Remarks:

a) we are again faced with the problem of solving a large set of nonlinear algebraic equations from poor initial guesses.

b) two competing technologies:

   i) sequential modular
   ii) equation oriented - assemble all equations and solve simultaneously with general-purpose solver.

CONSISTENT INITIALIZATION - GENERAL FORMULATION

c) the sequential modular approach is superior for finding steady-state - but what about other initial states? - little flexibility to define the initial state of a system.

d) however, direct (equation-oriented) methods superior for numerical integration of DAEs.

e) unresolved research question, two approaches:

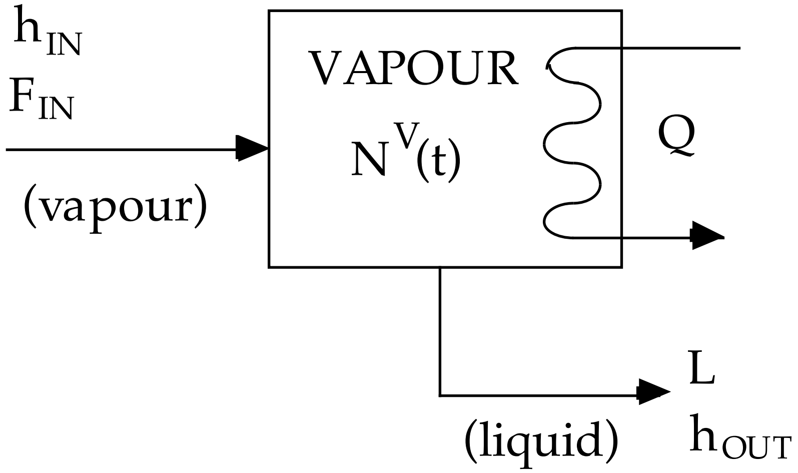
   i) improve robustness of equation oriented technology - ongoing problem: solve large systems of nonlinear equations from poor initial guesses.

   ii) interface sequential modular initial condition to direct integrator - but problem with flexibility remains.

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

Some DAE systems pose fundamental mathematical difficulties.

Example: total condenser for a pure component vapour (e.g. a steam jacket):



Assumptions:

(i) all liquid formed is immediately removed from the system (e.g., $N^L = 0$)
(ii) no sub cooling occurs
(iii) system reaches vapour-liquid equilibrium instantaneously
(iv) ideal physical properties

NOMENCLATURE

F_{IN}	total molar flow rate to vessel (vapour)	[mol s^{-1}]
h_{IN}	molar enthalpy of feed stream	[J mol^{-1}]
h_{OUT}	molar enthalpy of liquid stream	[J mol^{-1}]
L	total molar flow rate of liquid from vessel	[mol s^{-1}]
N^V	number of moles of vapour in vessel	[mol]
P	pressure of vessel	[Nm^{-2}]
Q	cooling load	[J s^{-1}]
R	gas constant	[J mol^{-1} K^{-1}]
T	temperature of vessel contents	[K]
U	internal energy of gas in vessel	[J]
V	volume of vessel (constant)	[m^{-3}]

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

Material Balance:

$$\frac{dN^V}{dt} = F_{IN} - L$$

Energy Balance:

$$\frac{dU}{dt} = F_{IN} h_{IN} - L h_{OUT} - Q$$

Physical Properties:

$P = P^{SAT}(T)$ (vapour-liquid equilibrium)

$PV = N^V RT$ (equation of state)

$$h_{OUT} = h^L(T)$$

$$N^V h^V(T) = U + PV$$

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

Simulation experiment:

(a) input set $R, V, F_{in}(t), h_{in}(t), Q(t)$
(b) initial condition at $t_0 = 0$

solve model for: $N^V(t), T(t), L(t), P(t), h_{OUT}(t), U(t)$

Degrees of freedom to specify initial condition:

6 equations
8 unknowns $\{N^V(0), \dot{N}^V(0), U(0), \dot{U}(0), T(0), L(0), P(0), h_{OUT}(0)\}$

⇒ 2 degrees of freedom for consistent initialization (?)

Try to specify the initial conditions $N^V(0), U(0)$:

1. given $N^V(0)$ solve:

$$P(0)V = N^V(0)RT(0)$$
$$P(0) = P^{SAT}(T(0))$$

for $P(0), T(0)$.

2. $h_{OUT}(0) = h^L(T(0))$

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

3. this leaves 3 equations:

$$\dot{N}^V(0) = F_{IN} - L(0)$$
$$\dot{U}(0) = F_{IN}h_{IN} - L(0)h_{OUT}(0) - Q$$
$$N^V(0)h^V(T(0)) = U(0) + P(0)V$$

in the unknowns $L(0), \dot{N}^V(0), \dot{U}(0)$.

BUT: equations are structurally singular - none of the unknowns appears in the third equation, e.g. the incidence matrix is singular for all values of the unknowns:

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \end{array} \left[\begin{array}{ccc} \dot{N}^V(0) & \dot{U}(0) & L(0) \\ \times & 0 & \times \\ 0 & \times & \times \\ 0 & 0 & 0 \end{array} \right]$$

⇒ badly posed problem.

This implies that the differential variables N^V and U are not independent. This is a:

"high index" problem

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

In this case, the problem is a consequence of our modelling assumptions: from classical thermodynamics (Duhem's theorem) we know that $NC + 2$ (3 in this case) independently variable properties fully define that state of a system at equilibrium (we have assumed the system is always at equilibrium). We have specified:

(a) the volume of the system
(b) the vapour fraction of the system (=1).

Therefore, we can only specify one out of $N^V(0)$ and $U(0)$ to fully define the state of the system.

Remarks:

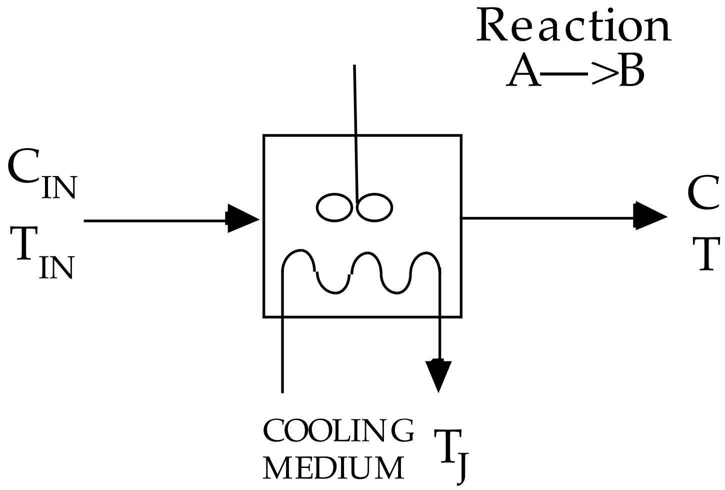
(a) if we can only specify one out of $N^V(0)$ and $U(0)$, where do we get an (additional) specification or equation to satisfy the degrees of freedom for consistent initialization?

(b) in this case, the high index problem is a consequence of our modelling assumptions (e.g. $N^L = 0$) - a fundamental property of the model. We could relax this assumption, and add a relationship relating the liquid holdup to the liquid flow rate.

However, we frequently want to make assumptions like this due to a lack of data, to reduce stiffness of the model, for simplicity, etc.

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

Example: dynamic “design” problem. Consider an exothermic fixed volume CSTR:



Mass Balance:

$$\frac{dC}{dt} = \frac{1}{\tau}(C_{IN} - C) - r$$

Energy Balance:

$$\frac{dT}{dt} = \frac{1}{\tau}(T_{IN} - T) + \left[\frac{-\Delta H_R}{\rho C_p}\right] r - \left[\frac{UA}{\rho C_p V}\right](T - T_J)$$

Rate equation:

$$r = ke^{-\Delta E / RT} C$$

NOMENCLATURE

A	heat transfer area of heating coil	[m^2]
C	concentration of A in reactor	[mol m^{-3}]
C_{IN}	feed concentration of A	[mol m^{-3}]
C_P	heat capacity	[J mol^{-1} K^{-1}]
K	pre-exponential factor	[s^{-1}]
T	temperature in reactor	[K]
T_{IN}	feed temperature	[K]
T_J	jacket temperature	[K]
U	heat transfer coefficient	[J m^{-2} K^{-1} s^{-1}]
V	reactor volume	[m^2]
ΔH_R	heat of reaction	[J mol^{-1}]
ΔE	activation energy	[J mol^{-1}]
ρ	density of reactor contents	[mol m^{-3}]
τ	residence time	[s]

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

1. Simulation Problem

Given: $A, C_p, \Delta E, \Delta H_R, k, U, V, \rho, R, \tau(t),$
$C_{IN}(t), T_{In}(t), T_J(t)$

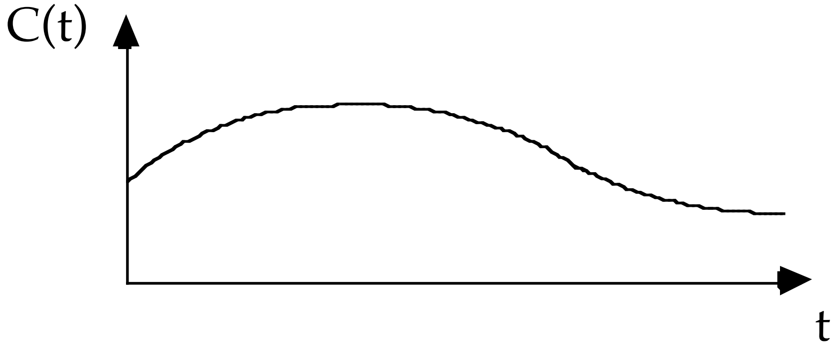
Determine: $C(t), T(t), r(t)$.

Straightforward: specify initial conditions $C(0), T(0)$ and solve equations.

⇒ no high index problem

We can assume from this that any high index problems are *not* a fundamental property of this model.

2. Dynamic Design Problem (or trajectory tracing problem): determine variation in coolant temperature $T_J(t)$ necessary to achieve a specified outlet concentration profile, e.g.:



CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

Simulation experiment: dynamic design problem:

Given: $A, C_p, \Delta E, \Delta H_R, k, U, V, \rho, R,$
$\tau(t), C_{IN}(t), T_{In}(t), C(t)$

Determine: $T(t), T_J(t), r(t)$

Degrees of freedom for consistent initialization:

3 equations
4 unknowns $\{T(0), T_J(0), r(0), \dot{T}(0)\}$

⇒ 1 degree of freedom for consistent initialization
($\dot{C}(0)$ derived by differentiating $C(t)$)

Try: $C(0)$ or $T(0)$:

1. obviously we cannot specify an arbitrary initial value for $C(0)$ — already specified.
2. can we specify an arbitrary initial value for $T(0)$?

Substitute rate equation into mass balance:

$$\dot{C}(0) = \frac{1}{\tau}(C_{IN} - C(0)) - ke^{-\Delta E / RT(0)} C(0)$$

BUT: $C(0), \dot{C}(0)$ fully specified

⇒ $T(0)$ determined by this equations, so it cannot be specified.

CONSISTENT INITIALIZATION AND HIGH INDEX DAE SYSTEMS

In this case, the high index problem arises through a particular choice of input variables and is not a fundamental property of the model.

In general, dynamic design problems will lead to high index problems because you are effectively trying to specify the time variation of a differential variable — hence it is no longer independent.

HIGH INDEX PROBLEMS - REMARKS

(a) despite the difficulties with consistent initialization, both problems have proper mathematical solutions.

(b) common features of high index problems:

(i) the differential variables are not independent and therefore cannot be given arbitrary initial values.

(ii) in many problems, a quick way to spot a high index problem is the fact that the algebraic equations cannot be solved for the algebraic variables given the differential variables - e.g. one or more algebraic variables only appear in the differential equations (example: the liquid flow rate from the total condenser).

(iii) it follows from (ii), that high index DAEs cannot be reduced to ODEs by using the algebraic equations to eliminate the algebraic variables ⇒ differentiation is required.

(c) two important issues with high index problems:

(i) where to get the additional equations to satisfy the degrees of freedom for consistent initialization?

(ii) controlling the error of integration in numerical algorithms — the differential variables are not really independent, even though the discretization treats them as such.

CONSISTENT INITIALIZATION OF HIGH INDEX SYSTEMS

We have seen that problems with consistent initial-ization are associated with “high index” problems.

Definition: the *index* of a DAE system is defined as the smallest non-negative integer *I* such that the DAE system and its first *I* derivatives with respect to time define $\dot{\mathbf{x}}$ and $\dot{\mathbf{y}}$ as locally unique functions of $\mathbf{x},\mathbf{y},\mathbf{u}$ (and its time derivatives) and t (Brenan *et al.*, 1989).

Although the above definition is more precise, it is, as is the case with many mathematical definitions, not very helpful from a practical point of view.

A more informal statement is that the index of a DAE system is the maximum number of differentiations of the original DAEs (or subsets of them) that must be undertaken to derive an ODE system (implicit or explicit).

⇒ all ODE systems are by definition of index 0.

This definition is sometimes called the *differential* index. Some authors use a *perturbation* index with a different definition, which is equal to or one greater that the differential index. In general, both indices are local quantities.

CONSISTENT INITIALIZATION OF HIGH INDEX SYSTEMS

Example 1:

$$\begin{aligned} \dot{x}_1 &= 2x_1 + x_2 + y && (1) \\ \dot{x}_2 &= x_1 - 2x_2 - 3y && (2) \\ 0 &= x_1 + x_2 - y && (3) \end{aligned}$$

Consistent initialization at $t = 0$: specify $x_1(0), x_2(0)$:

1. $y(0) = x_1(0) + x_2(0)$
2. $\dot{x}_1(0) = 2x_1(0) + x_2(0) + y(0)$
3. $\dot{x}_2(0) = x_1(0) - 2x_2(0) - 3y(0)$

Everything fine.

What is the index? Differentiate the algebraic equation (3):

$$\begin{aligned} &\Rightarrow 0 = \dot{x}_1 + \dot{x}_2 - \dot{y} \\ &\Rightarrow \dot{y} = 3x_1 - x_2 - 2y && (3') \end{aligned}$$

Equations (1),(2),(3′) form a system of three ODEs in x_1, x_2, y:

$$\Rightarrow \text{Index} = 1$$

CONSISTENT INITIALIZATION OF HIGH INDEX SYSTEMS

Example 2:

$$\dot{x}_1 = 2x_1 + x_2 + y \tag{1}$$
$$\dot{x}_2 = x_1 - 2x_2 - 3y \tag{2}$$
$$0 = x_1 + x_2 \tag{3}$$

Note: the algebraic variable y does not appear in the algebraic equation so we can expect trouble.

Consistent initialization: clearly $x_1(0)$ and $x_2(0)$ not independent.

What is the index? Differentiate the algebraic equation (3):

$$\Rightarrow 0 = \dot{x}_1 + \dot{x}_2$$
$$\Rightarrow 0 = 3x_1 - x_2 - 2y \tag{3′}$$

Equations (1), (2), (3′) are still DAEs! Differentiate (3′) - to derive second derivative of (3):

$$\Rightarrow 0 = 3\dot{x}_1 - \dot{x}_2 - 2\dot{y}$$
$$\Rightarrow \dot{y} = \frac{5}{2}x_1 + \frac{5}{2}x_2 + 3y \tag{3″}$$

Equations (1), (2), (3″) are an ODE system in x_1, x_2, y. We have differentiated (3) twice to derive the ODEs:

$$\Rightarrow \text{Index} = 2$$

CONSISTENT INITIALIZATION OF HIGH INDEX SYSTEMS

Each differentiation of (3) (i.e., a subset of equations) reduces the index by one — (1), (2), (3′) is an index-1 system.

In our index 2 example, if we cannot specify $x_1(0)$ and $x_2(0)$ independently, can we try another specification set of two variables?

Example: $x_1(0) = 1, y(0) = 1$:

1. $x_2(0) = -1$
2. $\dot{x}_1(0) = 2$
3. $\dot{x}_2(0) = 0$

which at first glance seems fine. However, from differentiation of the algebraic equation (3), we also know that:

$$0 = \dot{x}_1(t) + \dot{x}_2(t) \tag{3'}$$

...and this relationship must be satisfied at $t = 0$.

Clearly the values calculated above are inconsistent with this equation.

So, what are the necessary and sufficient conditions for a set of consistent initial values?

NECESSARY AND SUFFICIENT CONDITIONS FOR CONSISTENT INITIALIZATION

Necessary and sufficient conditions for a consistent set of initial values $\{\mathbf{x}(t_0),\dot{\mathbf{x}}(t_0),\mathbf{y}(t_0)\}$ are that they satisfy:

(a) the model equations at $t = t_0$ - i.e.:

$$\mathbf{f}(\dot{\mathbf{x}}(t_0), \mathbf{x}(t_0), \mathbf{y}(t_0), \mathbf{u}(t_0),) = \mathbf{0}$$

(b) the first and higher order time derivatives of the model equations at $t = t_0$

Note:

(i) we originally have $2n + m$ degrees of freedom.

(ii) the model equations take up $n + m$ degrees of freedom.

(iii) whether or not the derivatives of the equations take up further degrees of freedom is closely related to the index of the equations:

(a) if *index* ≥ 2, extra equations derived by differentiation are *always* required.

(b) for *most* index 1 systems, the original equations are sufficient.

For high index systems it is therefore necessary to identify which subset of the derivatives take up further degrees of freedom.

NECESSARY AND SUFFICIENT CONDITIONS FOR CONSISTENT INITIALIZATION

Obviously, for each extra equation required we have to make one less specification on the initial state of the system. Revisiting our examples:

Example 1:

$$\begin{aligned}
&\dot{x}_1 = 2x_1 + x_2 + y \\
&x_2 = x_1 - 2x_2 - 3y \\
&0 = x_1 + x_2 - y
\end{aligned}$$

differentiate:

$$\begin{aligned}
&\ddot{x}_1 = 2\dot{x}_1 + \dot{x}_2 + \dot{y} \\
&\ddot{x}_2 = \dot{x}_1 - 2\dot{x}_2 - 3\dot{y} \\
&0 = \dot{x}_1 + \dot{x}_2 - \dot{y}
\end{aligned}$$

we have introduced 3 new equations and three new variables $(\ddot{x}_1, \ddot{x}_2, \dot{y})$ — these equations do not constrain our original variables further.

⇒ we have 2 degrees of freedom to specify the initial condition.

NECESSARY AND SUFFICIENT CONDITIONS FOR CONSISTENT INITIALIZATION

Example 2:

$$\begin{aligned}
&\dot{x}_1 = 2x_1 + x_2 + y \\
&\dot{x}_2 = x_1 - 2x_2 - 3y \\
&0 = x_1 + x_2
\end{aligned}$$

differentiate:

$$\begin{aligned}
&\ddot{x}_1 = 2\dot{x}_1 + \dot{x}_2 + \dot{y} \\
&\ddot{x}_2 = \dot{x}_1 - 2\dot{x}_2 - 3\dot{y} \\
&0 = \dot{x}_1 + \dot{x}_2
\end{aligned}$$

Again, we have introduced three new equations and three new variables ($\ddot{x}_1, \ddot{x}_2, \dot{y}$). **BUT** the third equation does not involve these variables, it only involves our original variables and therefore must constrain them further. Further differentiation yields no further constraints on our original variables.

Therefore, the necessary and sufficient conditions for consistent initial values in this example are:

$$\begin{aligned}
&\dot{x}_1(0) = 2x_1(0) + x_2(0) + y(0) \\
&\dot{x}_2(0) = x_1(0) - 2x_2(0) + 3y(0) \\
&0 = x_1(0) + x_2(0) \\
&0 = \dot{x}_1(0) + \dot{x}_2(0)
\end{aligned}$$

This yields 4 equations in the 5 unknowns, so we have one degree of freedom to specify the initial condition.

NECESSARY AND SUFFICIENT CONDITIONS FOR CONSISTENT INITIALIZATION

Exercise: perform this analysis for the CSTR dynamic design problem — in fact, in this case we have zero degrees of freedom to specify the initial condition!

CONSISTENT INITIALIZATION - REMARKS

(a) in general, the degrees of freedom for consistent initialization is equal to or less than the number of differential variables.

(b) for large systems of DAEs it is difficult to determine by inspection which (if any) of the equations must be differentiated with respect to time to derive the necessary and sufficient conditions.

An algorithm that uses the structure of the equations (the incidence matrix) to identify the minimal set of equations that must be differentiated has been developed (Pantelides, 1988a). For most practical chemical engineering problems, the necessary and sufficient conditions can be derived in this manner from structural information alone.

However, there are situations in which structural information is insufficient. For example:

$$\dot{x}_1 = x_1 - x_2 + y \tag{1}$$
$$\dot{x}_2 = x_1 + x_2 - y \tag{2}$$
$$x_1 + x_2 = 1 \tag{3}$$

the structural algorithm will differentiate (3) to yield:

$$\dot{x}_1 = x_1 - x_2 + y \tag{1}$$
$$\dot{x}_2 = x_1 + x_2 - y \tag{2}$$
$$\dot{x}_1 + \dot{x}_2 = 0 \tag{3'}$$

and terminate.

CONSISTENT INITIALIZATION - REMARKS

The algorithm terminates because these equations are structurally nonsingular with respect to $\{\dot{x}_1, \dot{x}_2, y\}$. However, they are functionally singular with respect to this set — a further differentiation is actually required to derive the necessary and sufficient conditions.

(c) A sufficient condition for the original equations to be necessary and sufficient to define consistent initial values is:

$$\mathrm{Rank}\begin{bmatrix}\mathbf{f}_{\dot{\mathbf{x}}} & \mathbf{f}_{\mathbf{y}}\end{bmatrix} = n + m$$

i.e. the Jacobian of the equations with respect to $\dot{\mathbf{x}}$ and $\mathbf{y}$ is nonsingular.

Practical Consequence: if we can solve the original equations for $\dot{\mathbf{x}}(0)$ and $\mathbf{y}(0)$ given $\mathbf{x}(0)$, then we do not need to differentiate the original equations.

(d) most dynamic simulators[1] are therefore limited to those DAEs that satisfy the condition above. ABACUSS (Feehery and Barton, 1996) uses a combination of the structural algorithm and automated symbolic differentiation of the equations to find consistent initial conditions for a much broader class of DAE systems.

[1] i.e. any simulator apart from ABACUSS

CONSISTENT INITIALIZATION - REMARKS

(e) the condition above:

$$\text{Rank}\begin{bmatrix}\mathbf{f}_{\dot{\mathbf{x}}} & \mathbf{f}_{\mathbf{y}}\end{bmatrix} = n + m$$

is also a sufficient (but not necessary) condition for the DAEs to be index 1. Not all index 1 DAEs satisfy this condition: for example:

$$\dot{x} + 2\dot{x}_2 = 1 \qquad (1)$$
$$x_1 + x_2 = 1 \qquad (2)$$

Clearly $x_1(0)$ and $x_2(0)$ are not independent — one must derive an extra equation to define a consistent initial condition. However, the index is still 1 by the rigorous definition at the beginning of this section: differentiate the algebraic equations:

$$\Rightarrow \dot{x}_1 + \dot{x}_2 = 0 \qquad (2')$$

and (1) and (2′) yield the ODEs:

$$\dot{x}_1 = -1$$
$$\dot{x}_2 = 1$$

One differentiation ⇒ Index = 1.

CONSISTENT INITIALIZATION - REMARKS

Special index 1 problems of the above form arise most frequently when an index 2 problem has been transformed into an index 1 problem by eliminating an algebraic variable that only appears in the differential equation.

Observe that this transformation does not eliminate the problem of finding a consistent initial condition!

(f) there are claims in the literature concerning a necessary and sufficient condition for an index 1 system (Lefkopoulos and Stadtherr, 1993). These are incorrect: the condition in question is also only sufficient.

(g) in many cases, high index problems arise due to errors in model formulation. Typically, the model formulation implies that a greater number of quantities are independent than is actually the case. An index 1 model for the system can be derived through identification of the true number of independent quantities (Ponton and Gawthrop, 1991).

Clearly it is extremely important for a simulator to automatically identify these errors, and inform the user of alternatives.

CONSISTENT INITIALIZATION - REMARKS

(h) it is sometimes possible to “trick” a simulator into finding consistent initial values for a high index problem by specifying steady-state as the initial condition. In these cases, the specification of steady-state is redundant with the additional equations derived through differentiation, rather than inconsistent. Therefore, solving the equations with $\dot{\mathbf{x}}(0) = 0$ will give the same answer.

There are several examples in the literature of attempts to solve high index DAEs in this manner — it works provided no discontinuities in the inputs are introduced (consistent initialization is required at discontinuities as well - see later).

However, there are also severe problems with the numerical integration of DAEs with index ≥ 2 (see below) — the results of attempting to do this are invariably failure or garbage!

NUMERICAL INTEGRATION OF DAEs

Dynamic simulation of chemical processes requires the numerical solution of large DAE systems that are:

(a) **large** — typically 100s - 10,000s of equations (or even 100,000s!)

(b) **sparse** — only a few (~5) variables appear in each equation - special advantage can be taken in Jacobian evaluations and linear algebra.

(c) **stiff** — a range of phenomena taking place on a wide range of time scales — so the equations have widely different time constants.

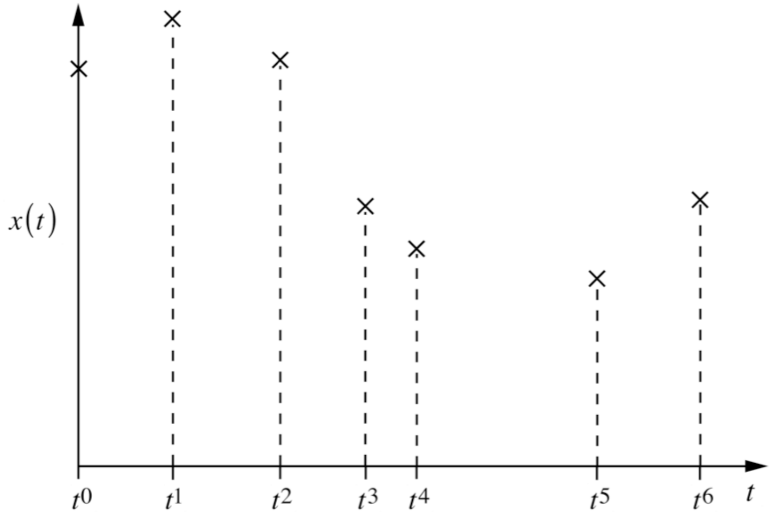
Numerical methods for integration of differential equations are usually categorized as *explicit* or *implicit*.

Implicit algorithms are used in most cases . . . details to follow.

See the recommended text books for a comprehensive treatment of this subject: different integration algorithms; properties of algorithms: convergence, consistency, stability (of the solution process, not the system of equations); integration error control; etc.

NUMERICAL INTEGRATION

Numerical integration methods approximate the solution of differential equations by obtaining the values of the unknowns at a set of distinct points in time (a process known as *discretization*):



Distinct time points: $t^{(i)}$ $\quad \forall i = 1 \ldots N$

Variables at $t^{(i)}$: $\underline{x}^{(i)}$

Derivatives at $t^{(i)}$: $\underline{\dot{x}}^{(i)}$

The values obtained must satisfy the differential equations at these points in time:

$$\underline{\dot{x}}^{(i)} = \underline{f}(\underline{x}^{(i)}, t^{(i)})$$

This is a set of n algebraic equations in the $2n$ unknowns $\left\{\underline{\dot{x}}^{(i)}, \underline{x}^{(i)}\right\}$.

=> we need another n equations to solve for the unknowns. Different numerical methods suggest different forms of these n extra equations.

EXPLICIT METHODS

There are two main objections to explicit methods for solving DAEs:

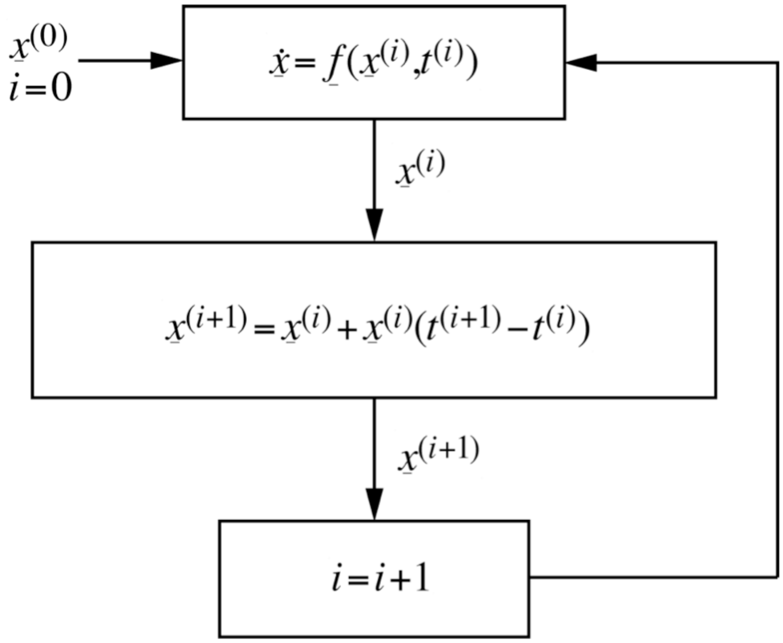
(a) it is not any cheaper (in computational effort) to solve DAEs with explicit methods (vs. implicit methods).

With ODEs, one advantage of an explicit method is that it is unnecessary to solve a set of nonlinear algebraic equations at each time step.

Example: Explicit Euler Method for ODEs uses the approximation:

$$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \dot{\underline{x}}^{(i)}(t^{(i+1)} - t^{(i)})$$

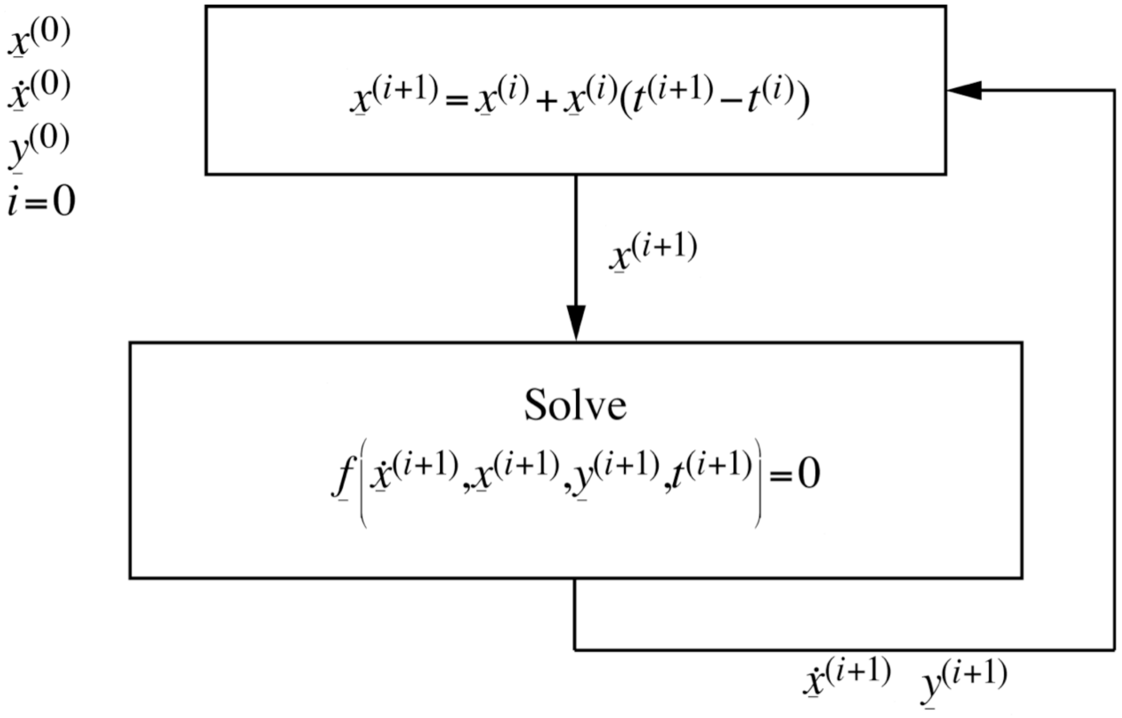
This leads to the algorithm:


$\underline{x}^{(0)}$
$i=0$
$\dot{\underline{x}} = \underline{f}(\underline{x}^{(i)},t^{(i)})$
$\underline{x}^{(i)}$
$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \underline{x}^{(i)}(t^{(i+1)} - t^{(i)})$
$\underline{x}^{(i+1)}$
$i=i+1$

EXPLICIT METHODS

⇒ for Explicit Euler each time step only requires a function evaluation of the model to determine $\underline{\dot{x}}^{(i)}$

For a system of DAEs the algorithm would be:


$\underline{x}^{(0)}$
$\underline{\dot{x}}^{(0)}$
$\underline{y}^{(0)}$
$i=0$
$\underline{x}^{(i+1)}=\underline{x}^{(i)}+\underline{x}^{(i)}(t^{(i+1)}-t^{(i)})$
$\underline{x}^{(i+1)}$
Solve
$\underline{f}\left(\underline{\dot{x}}^{(i+1)},\underline{x}^{(i+1)},\underline{y}^{(i+1)},t^{(i+1)}\right)=0$
$\underline{\dot{x}}^{(i+1)}$ $\underline{y}^{(i+1)}$

Note: must be able to solve $\underline{f}(\underline{\dot{x}},\underline{x},\underline{y},\underline{t})=0$ given $\underline{x}(t)$; explicit methods limited to index one problems that satisfy this condition.

∴ explicit methods for the general DAE formulation[1] require iterative solution of nonlinear equations.

[1] and indeed ODEs in implicit form as well (remember that an ODE in implicit form is the typical result of trying to derive an ODE from a DAE).

EXPLICIT METHODS

(b) process models give rise to *stiff* sets of differential equations.

For linear time invariant ODEs, i.e.:

$$\underline{\dot{x}} = \underline{\underline{A}}\,\underline{x}$$

stiffness is characterized by the Stiffness Ratio:

$$\mathrm{SR} \equiv \frac{\max\limits_{i} \left| \mathrm{Re}\,\lambda_i \right|}{\min\limits_{i} \left| \mathrm{Re}\,\lambda_i \right|} \geq 1$$

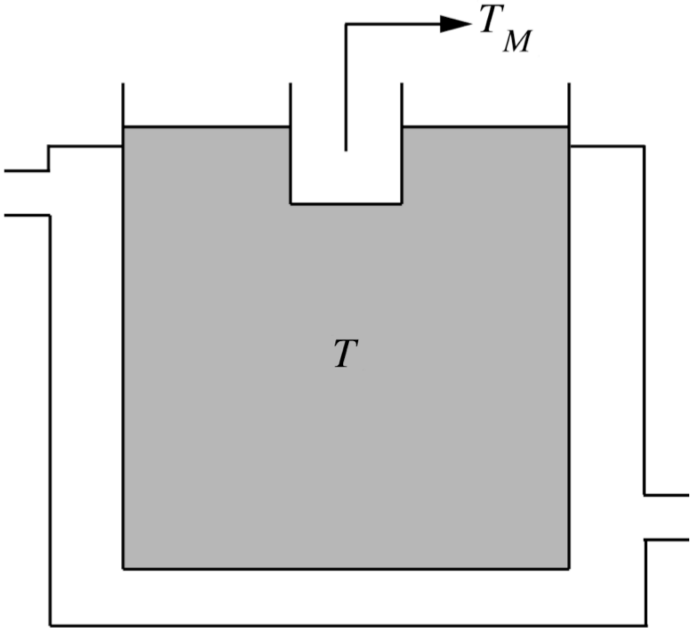
where λ_i = ith eigenvalue of $\underline{\underline{A}}$.

i.e. the stiffness ratio is the ratio of the eigenvalues of ☒ with the largest and smallest real components.

Process models tend to have high stiffness ratios (>1000) — i.e., there are phenomena taking place with widely different time constants.

EXPLICIT METHODS

Example: jacketed bath with thermowell:



Normalized model:

$$\begin{aligned}\frac{dT}{d\theta} &= 1-T \\ \frac{dT_M}{d\theta} &= \frac{1}{\tau}(T-T_M)\end{aligned} \Rightarrow \begin{bmatrix}\dot{T} \\ \dot{T}_M\end{bmatrix} = \begin{bmatrix}-1 & 0 \\ 1/\tau & -1/\tau\end{bmatrix}\begin{bmatrix}T \\ T_M\end{bmatrix} + \begin{bmatrix}1 \\ 0\end{bmatrix}$$

The time constant τ of the thermowell is related to ratio of the thermal inertias of the bath and the thermowell — i.e., the smaller the thermowell, the smaller the time constant $\tau (\tau << 1)$:

$$\mathrm{SR} = \frac{-1/\tau}{-1} = \frac{1}{\tau}$$

...as the thermowell becomes smaller relative to the bath, the stiffness ratio increases.

EXPLICIT METHODS

The maximum time step an explicit method can take is usually limited by stability criteria — i.e., the step size must always be smaller than some threshold for the solution process to remain stable.

Note:

- the time domain of interest is determined by the slowest decaying component — that with the largest time constant (smallest eigenvalue).

- but, the maximum step size is limited by the fastest decaying component (largest eigenvalue) to maintain stability.

As the stiffness ratio increases, the maximum step size decreases, but the time domain of interest remains fixed → an excessive number of time steps required.

Because stability criteria severely limit the step size in stiff problems, explicit methods are not applied very often to process simulation problems.

EXPLICIT METHODS

Note: several textbooks that discuss dynamic simulation of chemical processes advocate the use of explicit methods. These books were either written before reliable methods for solving stiff systems of ODEs had been developed, or recommend that the engineer determine the stiffness of the problem and select an appropriate numerical method. In the latter case, it seems safer to always use an implicit method because most problems are stiff, and non-stiff problems are trivial to solve with today's computational resources.

IMPLICIT METHODS

Implicit methods: always required to solve a set of nonlinear equations at each time step.

Advantage: theoretical guarantees for stability regardless of step size — step size limited by the need to control truncation error rather than stability.

=> we can take much larger time steps when the problem is stiff and the fast components of the solution have decayed away.

Example: Implicit Euler method for ODEs:

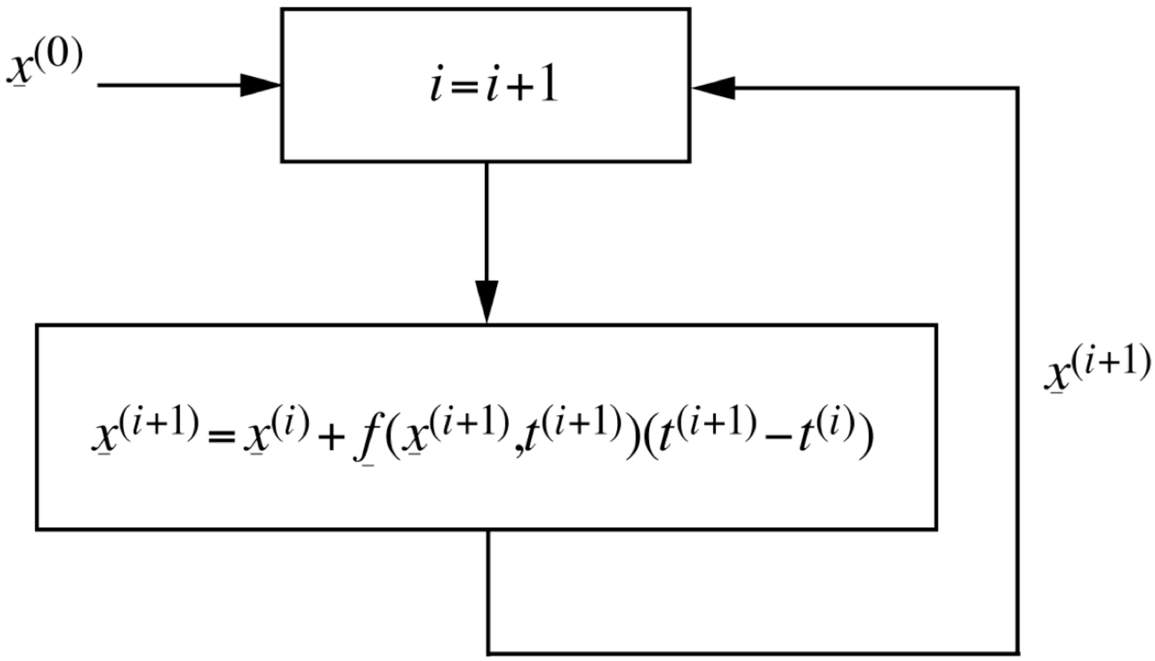
$$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \underline{\dot{x}}^{(i+1)}(t^{(i+1)} - t^{(i)})$$

...and substitute:

$$\underline{\dot{x}}^{(i+1)} = \underline{f}(\underline{x}^{(i+1)}, t^{(i+1)})$$

into this to yield Implicit Euler algorithm.

IMPLICIT METHODS


$\underline{x}^{(0)}$
$i = i + 1$
$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \underline{f}(\underline{x}^{(i+1)}, t^{(i+1)})(t^{(i+1)} - t^{(i)})$
$\underline{x}^{(i+1)}$

Predictor-Corrector methods:

- the numerical methods used to solve the set of nonlinear equations at each step require a good initial guess.
- use an explicit method to provide this guess (predictor step)
- then solve nonlinear equations (corrector step)

Example:

Predictor:	Explicit Euler step
Corrector:	Implicit Euler step using results of predictor step as initial guess.

IMPLICIT METHODS FOR DAEs

Most codes for solving DAEs use linear multi-step methods - in particular, the Backward Differentiation Formula (BDF) family of methods (also known as Gear's method) (Gear, 1971).

These methods have particularly good stability properties and hence are well suited to stiff problems.

Idea: fit *q*th order polynomial to *q+1* values of $\underline{x}$:

$$\underline{x}(t) = \underline{c}_0 + \underline{c}_1 t + \underline{c}_2 t^2 + \ldots + \underline{c}_q t^q$$

$$\dot{\underline{x}}^{(i)} = \frac{d\underline{x}(t^{(i)})}{dt} = \underline{c}_1 + 2\underline{c}_2 t^{(i)} + \ldots + q\underline{c}_q {t^{(i)}}^{q-1}$$

Given *q+1* values for $t^{(i)}$ and $\underline{x}^{(i)}$

$$\left\{t^{(i)}, \underline{x}^{(i)}\right\}, \left\{t^{(i-1)}, \underline{x}^{(i-1)}\right\}, \ldots, \left\{t^{(i-q)}, \underline{x}^{(i-q)}\right\}$$

we can derive the coefficients for these polynomials in terms of $\underline{x}^{(i-j)} \quad \forall j = 0 \ldots q$.

IMPLICIT METHODS FOR DAEs

Therefore, by differentiating the polynomial, the time derivatives can be expressed as a linear combination of these values:

$$\underline{\dot{x}}^{(i)} = \frac{1}{h^{(i)}} \sum_{j=0}^{q} \alpha_j \underline{x}^{(i-j)}$$

where:

$h^{(i)}$ = current step size
α_j = scalar multipliers; a function of the order of integration q and the current and previous step sizes.

So, we have established a relationship between $\underline{\dot{x}}^{(i)}$ and $\underline{x}^{(i)}$ and q previous values of the unknowns (presumed known).

This relationship can be solved simultaneously with the differential equations to determine $\underline{x}^{(i)}$ and $\underline{\dot{x}}^{(i)}$.

Note that the order of the integration method is given by the order of the polynomial q - the number of previous values used to derive the approximation to $\underline{\dot{x}}^{(i)}$.

Exercise: show that the first order Gear method is equivalent to the Implicit Euler method.

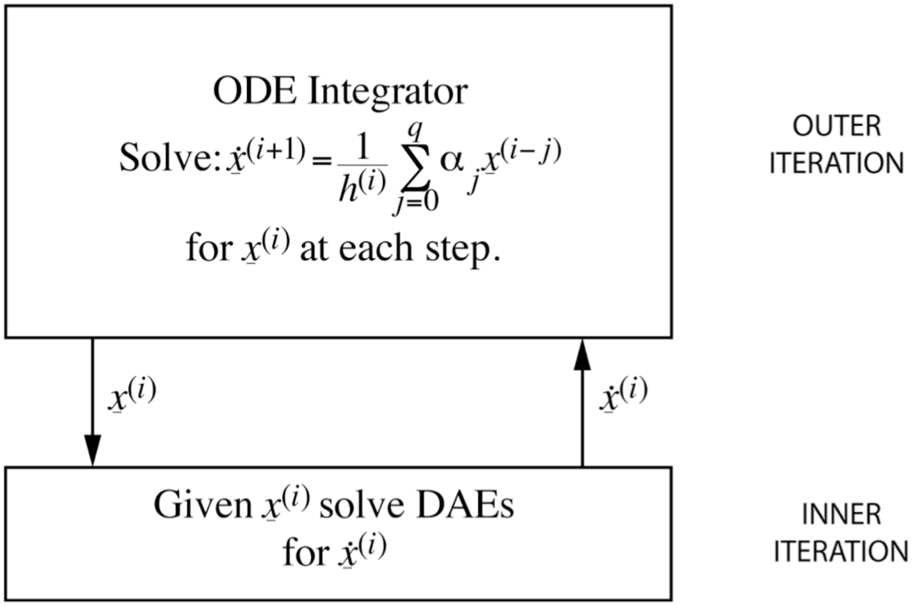
IMPLICIT METHODS FOR DAEs

Gear's method is a predictor corrector method:

a) predictor: extrapolate polynomial derived at previous time step.
b) corrector: solve polynomial and model equations iteratively.

Two approaches to solving DAEs with implicit methods have been proposed:

(A) reduce the DAE system to an explicit ODE system and solve using a standard BDF code for ODEs:



ODE Integrator

Solve: $\underline{\dot{x}}^{(i+1)} = \frac{1}{h^{(i)}} \sum_{j=0}^{q} \alpha_j \underline{x}^{(i-j)}$

for $\underline{x}^{(i)}$ at each step.

$\underline{x}^{(i)}$

$\underline{\dot{x}}^{(i)}$

Given $\underline{x}^{(i)}$ solve DAEs for $\underline{\dot{x}}^{(i)}$

OUTER ITERATION

INNER ITERATION

IMPLICIT METHODS FOR DAEs

Objections:

a) **VERY** expensive - at each step of the ODE method:

- each polynomial approximation must be solved iteratively.
- each function evaluation of this iteration requires the nested iterative solution of the DAEs

b) each nested solution to the DAEs is inexact and subject to numerical noise.

c) again, can only be applied to those index one DAEs for which we can solve $\underline{f}(\underline{\dot{x}},\underline{x},\underline{y},t)=0$ given $\underline{x}(t)$.

IMPLICIT METHODS FOR DAEs

(B) Solve the DAE system directly - eliminate $\underline{\dot{x}}(t)$ from $\underline{f}(\underline{\dot{x}},\underline{x},\underline{y},t)=0$ using the polynomial approximation to yield:

$$\underline{f}(\frac{\alpha_0}{h^{(i)}}\underline{x}^{(i)}+\underline{d}^{(i)},\underline{x}^{(i)},\underline{y}^{(i)},\underline{t}^{(i)})=0$$

...and solve this set of nonlinear equations simultaneously for $\underline{x}^{(i)}$ and $\underline{y}^{(i)}$ at each time step.

The $\underline{d}^{(i)}$ are constants defined by:

$$\underline{d}^{(i)}=\frac{1}{h^{(i)}}\sum_{j=1}^{q}\alpha_j\underline{x}^{(i-j)}$$

This method is much more efficient because the nested iteration is avoided - note the number of equations solved is the same as the inner iteration above, and in the explicit method.

There are several reliable and efficient implementations of the direct approach - e.g., DASSL (Petzold, 1983), DASOLV (Jarvis and Pantelides, 1992), and DSL48S (Barton *et al*., 1997).

A version of DASOLV is used by SpeedUp. Within ABACUSS, the user may either use DSL48S or DASOLV.

VARIABLE ORDER, VARIABLE STEP SIZE METHODS

Typically, rather than requiring the user to specify the step size and integration order, a good BDF code will automatically adjust:

- the integration step size
- and the order of integration

so that estimates of the truncation error satisfy some user specified tolerance.

Warning: controlling the error of integration (i.e. choosing step size, order) is a complex task - standard codes will invariably do a better job than you can.
Fix the step size at your peril!

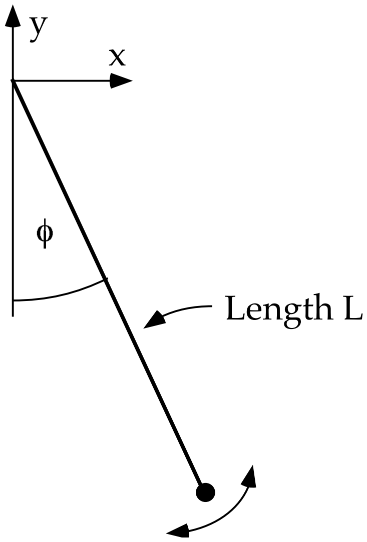
Clearly Gear's method must always start with a first order step (Implicit Euler) and then automatically adjust the order and step size as the number of previous steps increases and "confidence" concerning the behaviour of the solution is built up.

Note: some simulators prompt the user for a step size - always check what this means (for example, in SpeedUp this is just a reporting interval for the results and has no effect whatsoever on the actual steps taken by the numerical methods).

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

Severe difficulties are encountered when the standard codes mentioned above are applied for numerical integration of high index systems.

Example: undamped rigid pendulum modelled in Cartesian co-ordinates.



Problem:

$$m\ddot{x} + \frac{\lambda x}{L} = 0$$

$$m\ddot{y} + \frac{\lambda y}{L} = -mg$$

$$x^2 + y^2 = L^2$$

$$y(0) = 0$$

$$\dot{y}(0) = 0$$

Differential index = 3.

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

The pendulum problem is the standard academic example. In this case, the index problem arises due to the assumptions used in model development (use of Cartesian co-ordinates when rigidity of the pendulum implies the x and y positions are not independent).

For comparison purposes, if the model is reformulated in polar co-ordinates, we get:

$$\ddot{\phi} + \frac{g}{L}\sin(\phi) = 0$$

where ϕ is the angle of rotation.

The index of this model is 0 (an ODE). Hence, the numerical solution to this model is a reference by which numerical solutions to the index 3 model can be judged.

Note that in the following, only the solution found by the method of dummy derivatives is virtually indistinguishable from that of the index 0 model.

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

CASE 1: SOLVE FROM CONSISTENT INITIAL CONDITIONS WITH STANDARD CODE

All case studies are solved from the consistent initial condition defined by the solution to the equations:

$$
\begin{aligned}
m\ddot{x}(0)+\frac{\lambda(0)x(0)}{L} &= 0\\
m\ddot{y}(0)+\frac{\lambda(0)y(0)}{L} &= -mg\\
x(0)^2+y(0)^2 &= L^2\\
2x(0)\dot{x}(0)+2y(0)\dot{y}(0) &= 0\\
2x(0)\ddot{x}(0)+2y(0)\ddot{y}(0)+2\dot{x}(0)^2+2\dot{y}(0)^2 &= 0\\
y(0) &= 0\\
\dot{y}(0) &= 0
\end{aligned}
$$

For this problem, standard code will take a few steps, then cut the step size to zero and terminate abnormally.

Since a high index problem is not the only cause of this type of behaviour, the user is left clueless as to what to do next.

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

CASE 2: SOLVE USING METHOD SUGGESTED BY GEAR AND PETZOLD (1984)

Gear and Petzold (1984) suggested deriving the underlying index 1 system from a high index system using differentiation, and then submitting this index 1 system to a standard solver.

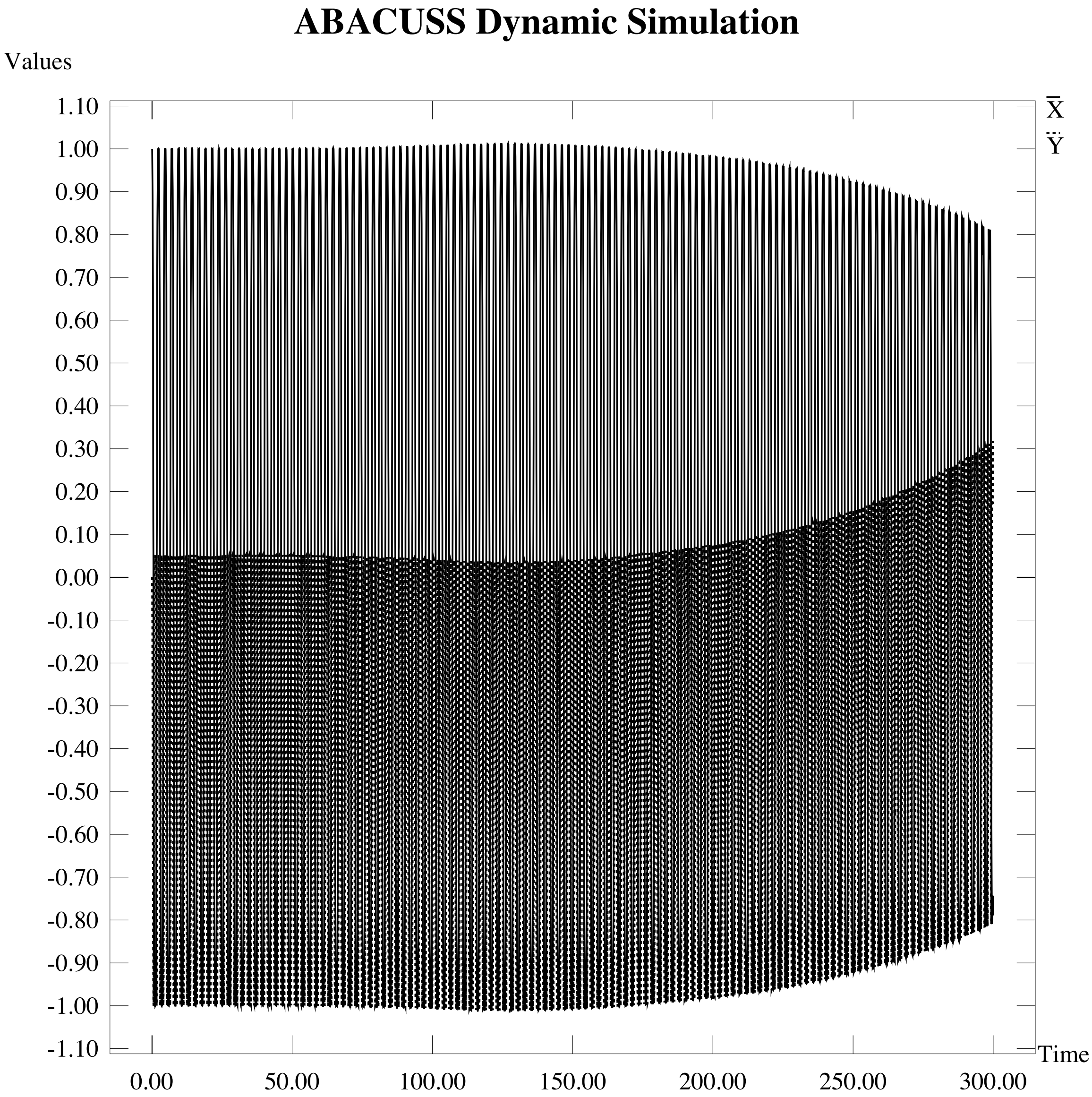
For the pendulum example, this process yields the index 1 system:

$$
\begin{aligned}
m\ddot{x} + \frac{\lambda x}{L} &= 0 \\
m\ddot{y} + \frac{\lambda y}{L} &= -mg \\
2x\ddot{x} + 2y\ddot{y} + 2\dot{x}^2 + 2\dot{y}^2 &= 0
\end{aligned}
$$

PROBLEM: solution set of this system is bigger than original index 1 system. The solution satisfies the second derivative of the path constraint $x^2 + y^2 = L^2$ rather than the path constraint itself.

In practice, even given consistent initial conditions, rounding and truncation error lead to a disastrous deviation from the path constraint (see overleaf).

METHOD OF GEAR AND PETZOLD (1984) RESULTS



The incredible shrinking pendulum!!

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

CASE 3: THE METHOD OF DUMMY DERIVATIVES

(Mattsson and Soderlind, 1993; Feehery and Barton, 1996)

PROPOSITION: for any high index system there exists a family of index 1 systems that have an *equivalent* solution set to the original high index system.

IDEA: if we can find one of these equivalent index 1 systems, we can then submit this index 1 system to a standard code and the results from solution of this system can be treated as the solution of the high index system.

Mattsson and Soderlind (1993) proposed an algorithm that can find a member of this family of equivalent index 1 systems for a broad class of nonlinear DAEs of arbitrary index.[1]

Overleaf we show how their algorithm works when applied to the pendulum example.

[1] specifically, those for which structural criteria can correctly determine the differential index of the system

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

1. apply Pantelides' (1988) algorithm to derive the system of equations that must be satisfied by consistent initial conditions:

$$m\ddot{x} + \frac{\lambda x}{L} = 0$$

$$m\ddot{y} + \frac{\lambda y}{L} = -mg$$

$$x^2 + y^2 = L^2$$

$$2x\dot{x} + 2y\dot{y} = 0$$

$$2x\ddot{x} + 2y\ddot{y} + 2\dot{x}^2 + 2\dot{y}^2 = 0$$

2. this is an overdetermined dynamic system (more equations than unknowns). Pick a subset of the time derivatives to substitute for dummy algebraic variables.

$$mx'' + \frac{\lambda x}{L} = 0$$

$$m\ddot{y} + \frac{\lambda y}{L} = -mg$$

$$x^2 + y^2 = L^2$$

$$2xx' + 2y\dot{y} = 0 \qquad \text{(EQUIV)}$$

$$2xx'' + 2y\ddot{y} + 2x'^2 + 2\dot{y}^2 = 0$$

...$\dot{x}$ and $\ddot{x}$ replaced by new algebraic variables x' and x''. Note true dynamic degrees of freedom of system revealed.

NUMERICAL INTEGRATION OF HIGH INDEX DAEs

3. system (EQUIV) is a fully determined index 1 system (when $x \neq 0$). Solve with standard code.

In this example there are two members of the family of equivalent index 1 systems: instead, $\dot{y}$ and $\ddot{y}$ could have been substituted (this system is index 1 except when $y \neq 0$).

PROBLEM: members of the family of index 1 models can have their index change locally (to index 2+) at certain points in state space. If a numerical method steps in the neighbourhood of such points, severe difficulties, typically failure, are encountered (Feehery and Barton, 1996).

Example: the model (EQUIV) above becomes high index locally when $x = 0$ (similarly, the other member of the family becomes high index locally when $y = 0$).

SOLUTION: since the points in state space where the index rises locally do not coincide, switch between members of the family to avoid approaching such points for the current model.

This is known as *dummy pivoting*.

Feehery and Barton (1996) have recently developed an efficient and automated approach for dummy pivoting.

THE METHOD OF DUMMY DERIVATIVES IN ABACUSS

ABACUSS is now capable of automatically solving a broad class of nonlinear DAEs of arbitrary index[2] using the method of dummy derivatives (Feehery and Barton, 1996).

Give a model coded in ABACUSS, the following steps are done automatically:

- identification of the extra equations that constrain the initial conditions using Pantelides' (1988) algorithm.

- automated derivation of these extra equations using symbolic or automatic differentiation

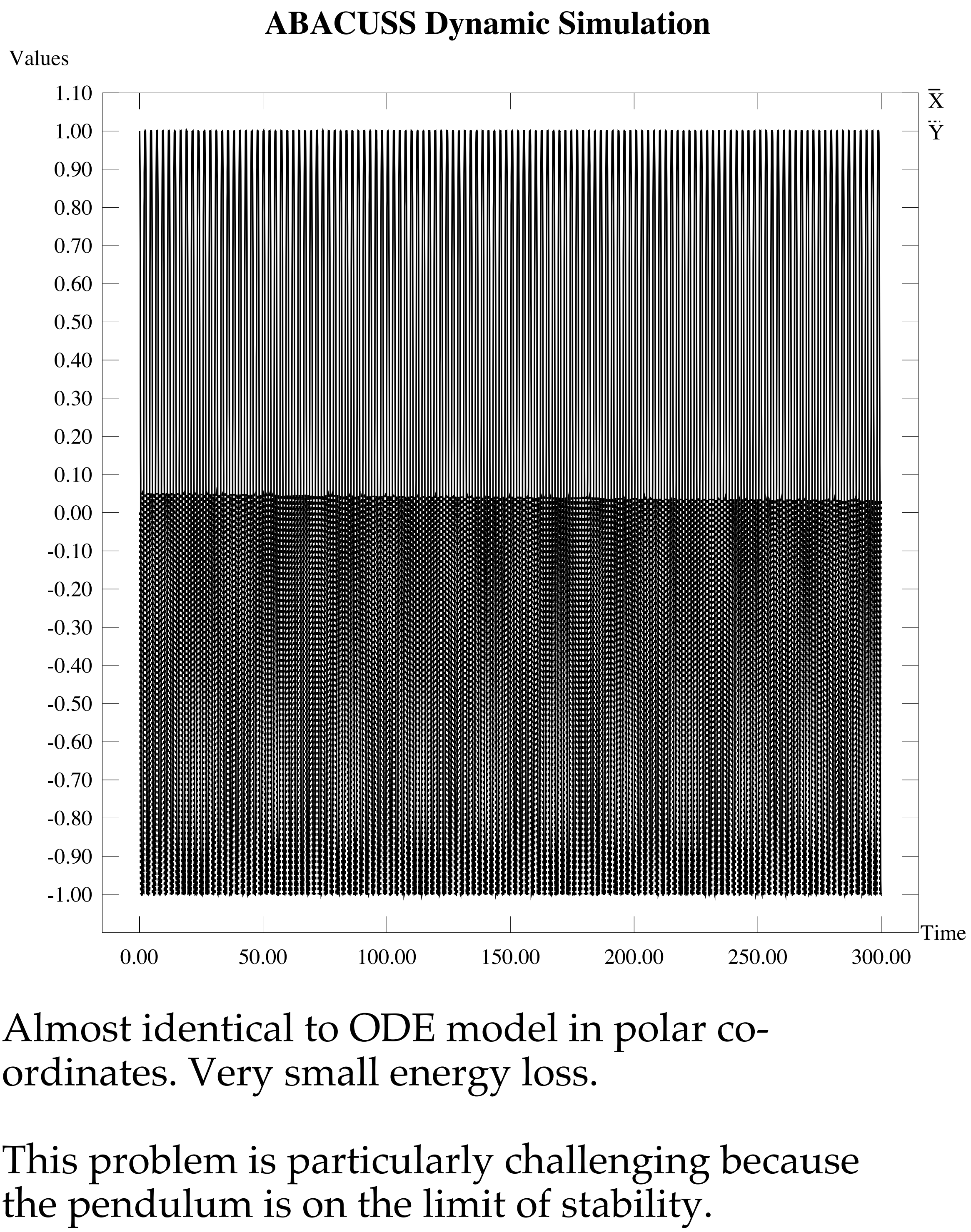
- generation of statistics on the (structural) differential index, degrees of freedom for consistent initialization, etc.

- given the degrees of freedom, solution of the consistent initialization (standard NLE solver)

- automated numerical integration using the method of dummy derivatives, and the dummy pivoting algorithm of Feehery and Barton(1996).

[2] specifically, those for which structural criteria can correctly determine the differential index of the system

METHOD OF DUMMY DERIVATIVES RESULTS



Almost identical to ODE model in polar co-ordinates. Very small energy loss.

This problem is particularly challenging because the pendulum is on the limit of stability.

NUMERICAL INTEGRATION OF HIGH INDEX DAEs
STANDARD CODES

When the standard numerical codes are applied to DAEs with index ≥ 2, the stability, convergence and error control of the BDF method begin to break down.

Health Warning: although (given a consistent initial condition) standard DAE codes may be able to advance the solution of high index problems - the results will be garbage (error control breaks down).

- in the pendulum example we were fortunate that the code gave up

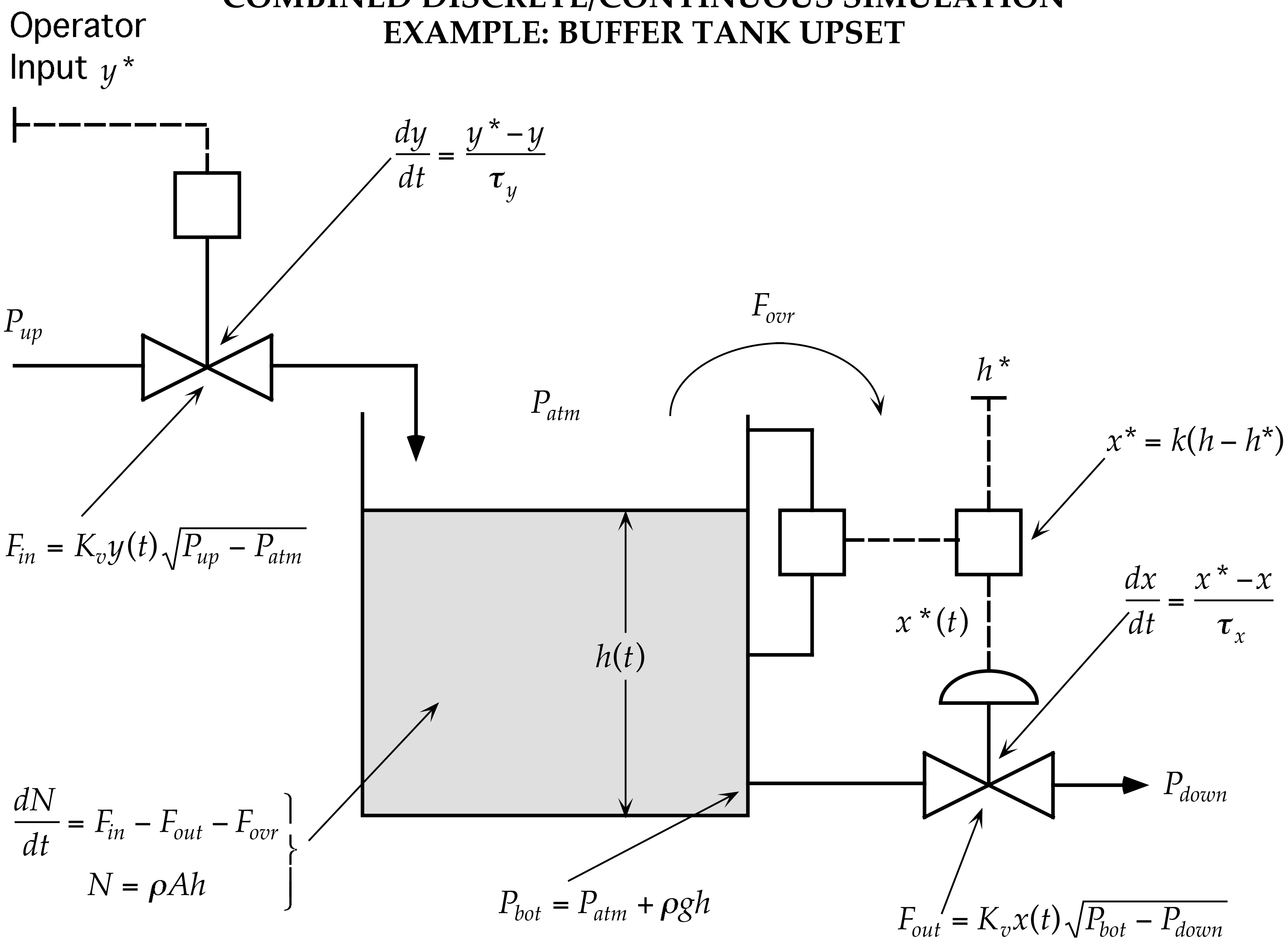
- there are examples in the literature (particularly distillation simulation) where the code has advanced solution of a high index model: the results are terrible - for example, predicting inverse responses where no such physical behaviour exists.

Standard codes can be modified to solve classes of nonlinear index 2 and index 3 systems correctly. However, no such modification exists to solve nonlinear DAEs of arbitrary index.

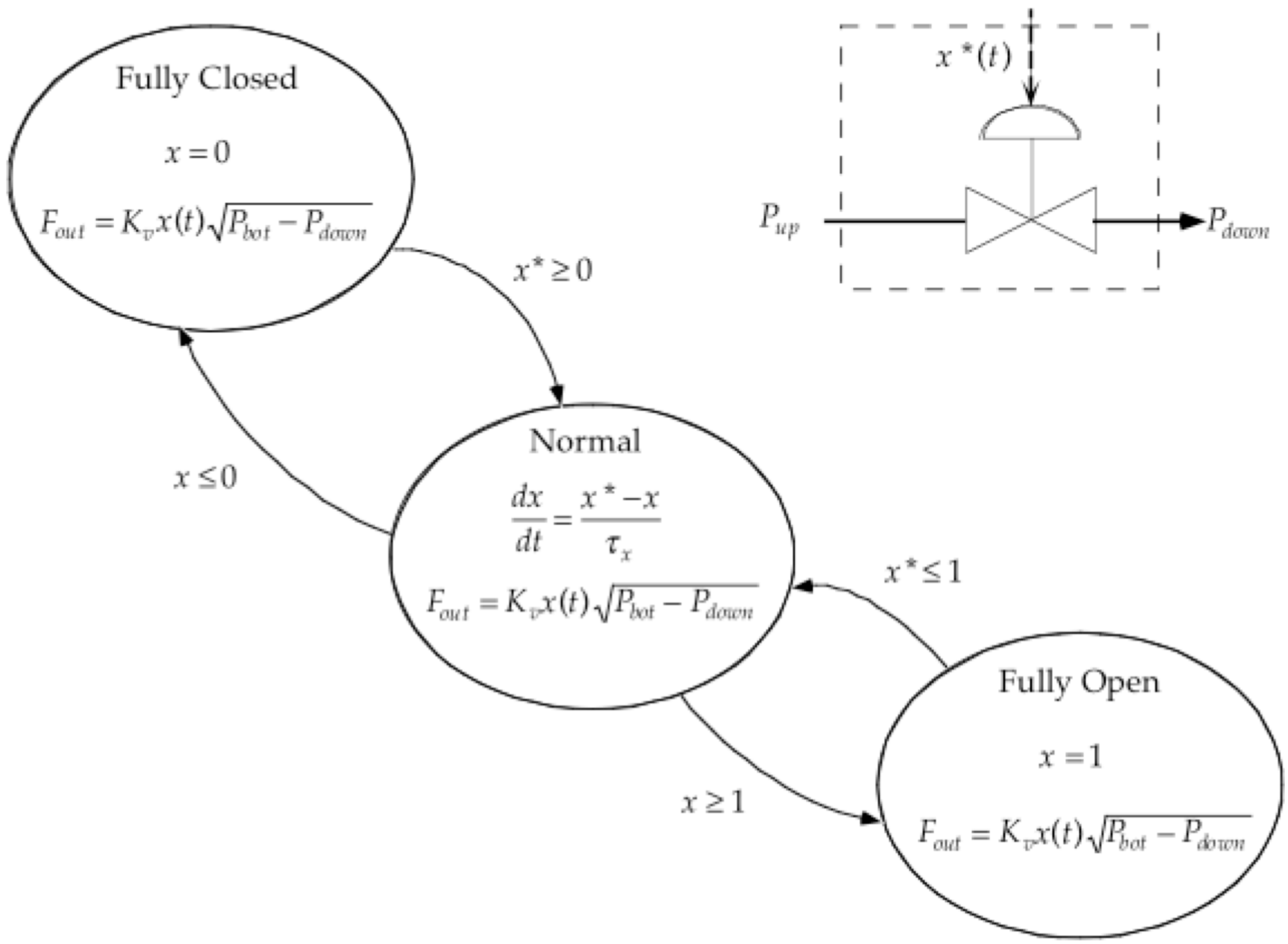
Why care about index 3+ systems?

- dynamic design problems: specify pressure at the bottom of a column and the index is NP+1 where NP is the number of trays
- path constrained dynamic optimizations

COMBINED DISCRETE/CONTINUOUS SIMULATION EXAMPLE: BUFFER TANK UPSET

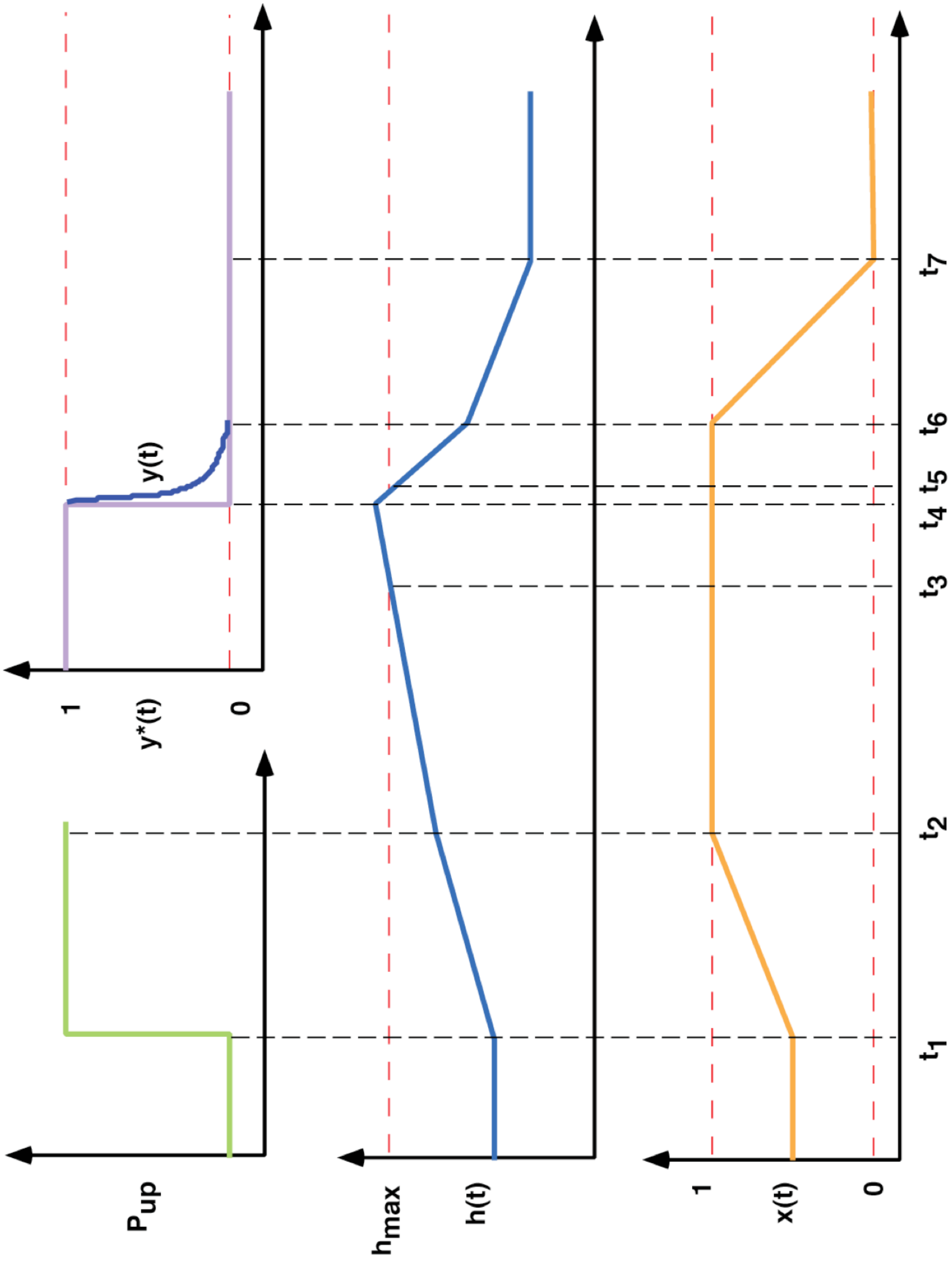

Operator Input y^*
$\frac{dy}{dt} = \frac{y^* - y}{\tau_y}$
P_{up}
$F_{in} = K_v y(t)\sqrt{P_{up} - P_{atm}}$
F_{ovr}
P_{atm}
h^*
$x^* = k(h - h^*)$
$\frac{dx}{dt} = \frac{x^* - x}{\tau_x}$
$x^*(t)$
$h(t)$
P_{down}
$\frac{dN}{dt} = F_{in} - F_{out} - F_{ovr}$
$N = \rho A h$
$P_{bot} = P_{atm} + \rho g h$
$F_{out} = K_v x(t)\sqrt{P_{bot} - P_{down}}$

VALVE AUTOMATON



$x^*(t)$
P_{up}
P_{down}
Fully Closed
$x = 0$
$F_{out} = K_v x(t)\sqrt{P_{bot} - P_{down}}$
$x^* \geq 0$
$x \leq 0$
Normal
$\frac{dx}{dt} = \frac{x^* - x}{\tau_x}$
$F_{out} = K_v x(t)\sqrt{P_{bot} - P_{down}}$
$x^* \leq 1$
$x \geq 1$
Fully Open
$x = 1$
$F_{out} = K_v x(t)\sqrt{P_{bot} - P_{down}}$

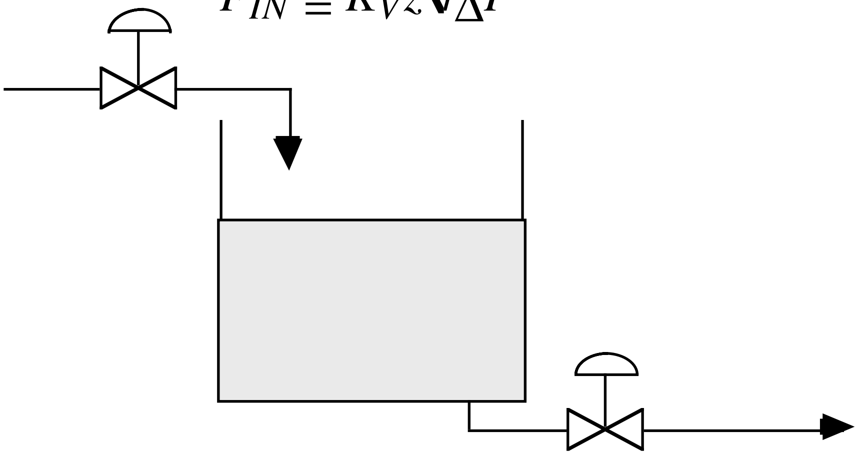
RESPONSE OF TANK TO MAJOR SURGE IN UPSTREAM PRESSURE



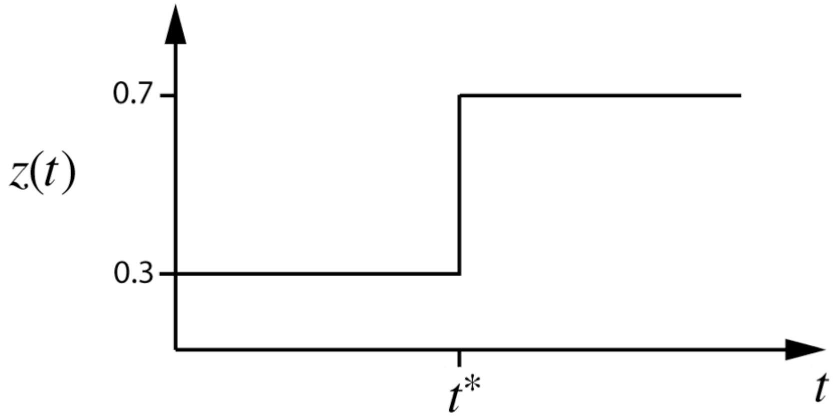
DISCONTINUITY HANDLING

In most dynamic simulation experiments we must be able to handle discontinuities in both the functions assigned to the input variables and discrete changes to the functional form of the equations that describe the behaviour of the system. **Example:** open the feed valve to a tank:

$$F_{IN} = K_V z \sqrt{\Delta P}$$

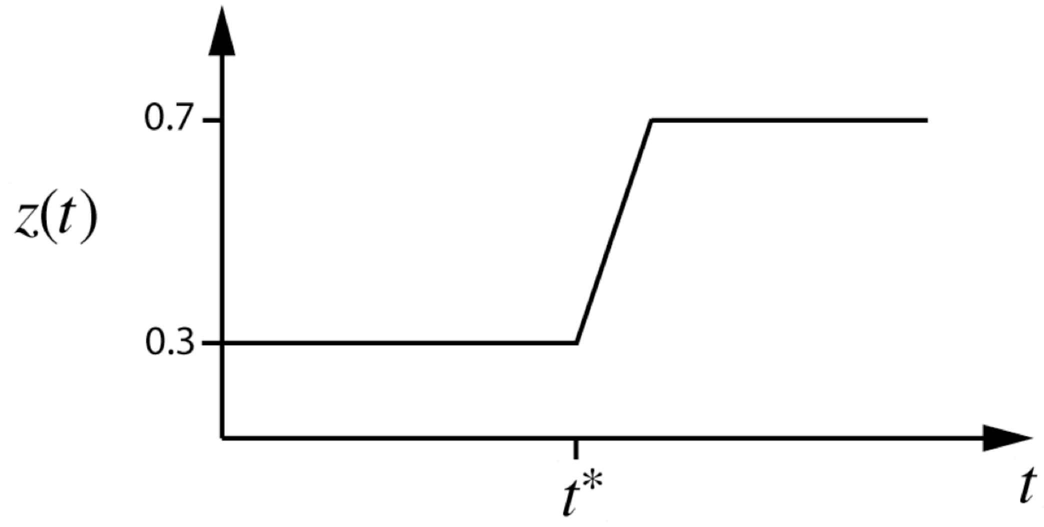


In this case, we select the input to the system (e.g., the variable we can control) as the stem position of the control valve $z(t)$. We can model the valve opening at time $t*$ by defining the time variation of the stem position as a discontinuous function, for example:



DISCONTINUITY HANDLING

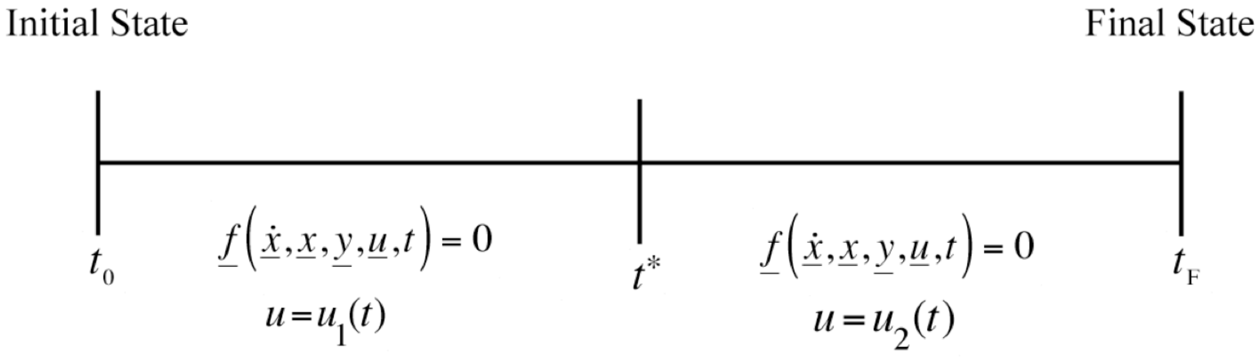
Note: assigning a step change to the stem position will typically be an abstraction. In reality, the valve will require a finite amount of time to open. Obviously, if this level of detail is necessary, we could assign the following function to the stem position:



. . . or even model the valve dynamics and introduce the step change in the pressure signal to the valve. Remember, as the modeller *you* choose the appropriate level of abstraction.

DISCONTINUITY HANDLING

If we know in advance when the valve is to be opened, we can divide the time interval of interest $[t_0, t_f]$ into two sub intervals:


Initial State
Final State
t_0
$\underline{f}\left(\underline{\dot{x}},\underline{x},\underline{y},\underline{u},t\right)=0$
$u=u_1(t)$
t^*
$\underline{f}\left(\underline{\dot{x}},\underline{x},\underline{y},\underline{u},t\right)=0$
$u=u_2(t)$
t_F

. . . note that in this case the process model in each interval is the same, but we have assigned different functions to the input variables in each interval.

What we have done is posed the overall simulation problem as two initial value problems in DAEs:

(a) given consistent initial conditions at $t = t_0$, integrate numerically the first initial value problem to t^*. If the last step goes beyond t^*, simply curtail the step to t^*.

(b) at t^* set up the new initial value problem — change the functions assigned to the inputs.

(c) determine consistent initial conditions for the variables $\underline{\dot{x}}$, $\underline{x}$ and $\underline{y}$ at t^*.

(d) integrate numerically this new initial value problem from t^* to t_f. Note integration is restarted: a BDF method must start again with a first order step.

DISCONTINUITY HANDLING

If there are more discontinuities like this, just divide the time interval into more sub intervals:

=> **dynamic process simulation is a sequence of initial value problems in DAEs interspersed by discontinuities** (Barton and Pantelides, 1994)

Note:

(a) the end of a sub interval is marked by the occurrence of an *event*.

(b) the time variation of the system between events is smooth.

CONSISTENT REINITIALIZATION

At the events that mark the end of the sub intervals it is necessary to determine consistent initial values for the variables at the beginning of the new interval - this is called *consistent reinitialization*.

For a ODE system this reinitialization is trivial:

$$\dot{\underline{x}}(t^*) = f(\underline{x}(t^*), \underline{u}(t^*), t^*)$$

. . . it is assumed that the values of the variables $\underline{x}$ are continuous at the event, so discontinuities can only appear in the values of the time derivatives $\dot{\underline{x}}$ (the function above is evaluated with the new functions assigned to the input variables).

For a DAE system both the time derivatives *and* the values of the algebraic variables can be discontinuous.

Example: consider again the inlet valve of our tank. The flow rate into the tank is related algebraically to the stem position of the valve by the equation:

$$F_{IN} = K_V z \sqrt{\Delta P}$$

. . . even if the pressure drop remains constant at the discontinuity, the value of F_{IN} must be discontinuous to remain consistent with the discontinuous value of the stem position $z(t)$ at t^*.

CONSISTENT REINITIALIZATION

In general, with a DAE system we need to determine consistent initial values for the new initial value problem at each event. We proceed in the following manner:

(a) determine the values of the inputs at $t^* - \underline{u}(t^*)$ – according to the new functions.

(b) the necessary conditions for a consistent initial condition are given by the model equations at t^*:

$$\underline{f}(\dot{\underline{x}}(t^*), \underline{x}(t^*), \underline{y}(t^*), \underline{u}(t^*), t^*) = 0$$

These *may* have to be augmented with additional equations derived by differentiation of the original equations - i.e. when index ≥ 2.

For this discussion, we will assume the index = 1 and the original equations are necessary and sufficient to define consistent initial values.

(c) we have:

$n + m$ equations

and

$2n + m$ unknowns $\left\{\dot{\underline{x}}(t^*), \underline{x}(t^*), \underline{y}(t^*)\right\}$

=> we need n more side conditions to fully define the initial state of the system in the new sub interval.

CONSISTENT REINITIALIZATION

In order to obtain these side conditions, the normal assumption is that the values of the differential variables are continuous across the boundary between two intervals, e.g., we solve:

$$\underline{f}(\underline{\dot{x}}(t^*),\underline{x}(t^*),\underline{y}(t^*),t^*)=0 \qquad \text{(R1)}$$

for $\underline{\dot{x}}(t^*)$ and $\underline{y}(t^*)$ keeping $\underline{x}(t^*)$ constant at the values at the end of the previous interval (leading to a fully determined system of equations).

The physical reasoning behind this is that the differential variables represent conserved quantities (such as mass, energy, or momentum) or are directly related to them:

=> at an event, the values of input, algebraic or time derivative variables *may* be discontinuous, but the values of the differential variables are continuous.

Note: this poses problems for DAE systems for which the condition:

$$\text{Rank}\begin{bmatrix} \underline{f}_{\underline{\dot{x}}} & \underline{f}_{\underline{y}} \end{bmatrix} = n+m$$

is not satisfied (i.e. all index ≥ 2 and some index = 1). Clearly, if this condition is not satisfied the calculation (R1) above is impossible. In these cases, the assumption of continuity for *all* the differential variables at the boundary is either redundant or inconsistent with some of the time derivatives of the model equations. Research continues...

EVENTS

Discontinuities[1] are triggered by the occurrence of *events* during the simulation experiment.

Events belong to one of two categories:

1. *Time Events* — the exact time of occurrence is known in advance. Solution of the sequence of initial value problems can proceed to these events in strict time order:

   (a) advance numerical integration in the current interval.

   (b) curtail the final integration step length to the event time.

   (c) restart the simulation with the new initial value problem.

   Time events can be further categorized as:

   - exogenous — if the time of occurrence is known *a priori*
   - endogenous — if the time of occurrence is calculated from some earlier event in the simulation.

   ...in either case, the time of occurrence of the next time event is known when integration starts in a sub interval.

[1] discontinuities may occur in the value of variables and/or in the value of their time derivatives.

EVENTS

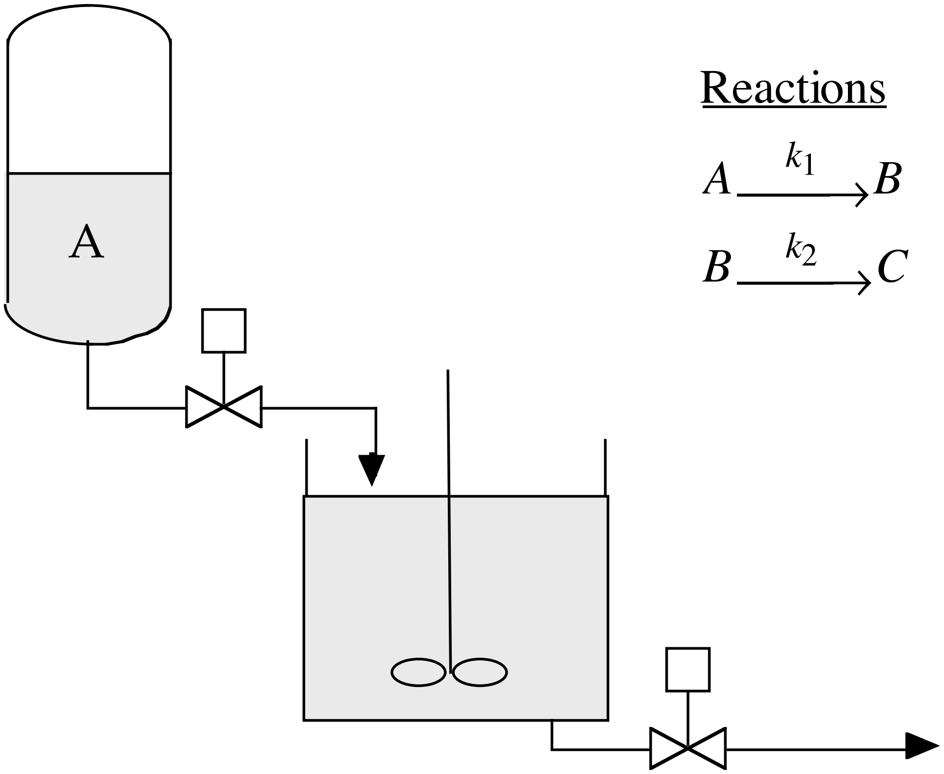
2. *State Events* — the time of occurrence of state events is not known in advance because it is dependent on the system fulfilling certain conditions — *state conditions*.

   Numerical integration of the current IVP must be advanced speculatively until the state condition becomes satisfied.

Discontinuities as a consequence of state events require careful handling . . . details to follow.

STATE EVENTS — EXAMPLE

Example: consider a batch reactor:



Reactions

$$A \xrightarrow{k_1} B$$

$$B \xrightarrow{k_2} C$$

Homogeneous catalytic liquid phase reactions; product B desired, C waste.

=> operating objective: maximize yield of B

From reaction engineering we know that the maximum yield of B occurs when:

$$\frac{dC_B}{dt} = 0$$

i.e. the concentration of B in the reactor reaches a maximum.

STATE EVENTS — EXAMPLE

Provided we can estimate dC_B / dt with some analytical instrument, we could define the operating policy for the reactor in the following manner:

```
# charge the reactor
OPEN INLET VALVE
WAIT 10 MINUTES
CLOSE INLET VALVE

INJECT CATALYST
WAIT UNTIL $dC_B / dt$ ≤ 0

# discharge the reactor
OPEN OUTLET VALVE
WAIT 10 MINUTES
CLOSE OUTLET VALVE
```

In order to simulate this operating policy, we must be able to deal with the state condition defined by:

$$\frac{dC_B}{dt} \leq 0$$

. . . i.e., advance simulation speculatively until this condition is satisfied, then open the outlet valve (a discontinuity).

It is extremely important to locate state events accurately and deal with them in strict time order — the occurrence of a discontinuity can radically change the way in which the subsequent process behaviour evolves with time.

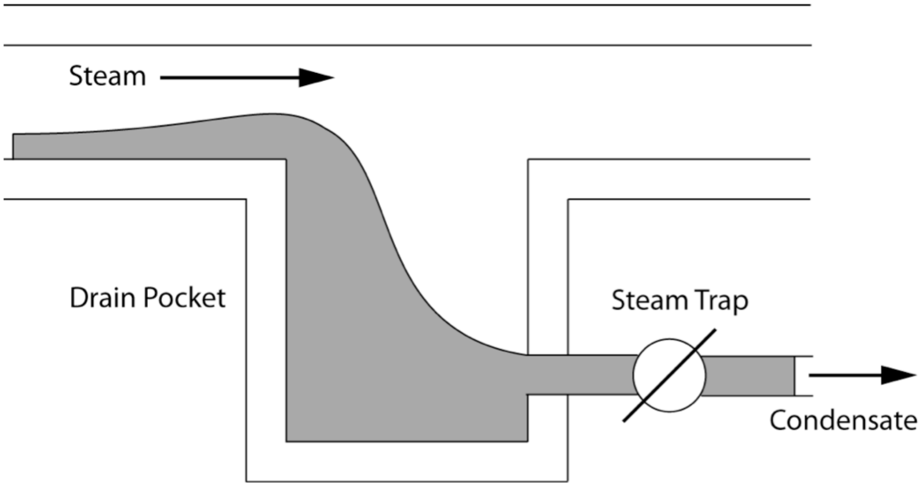
STATE EVENTS

In general, a state event is the time at which a state condition expressed in terms of classical propositional logic becomes true, i.e.:

$$l(\underline{x},\underline{\dot{x}},\underline{y},\underline{\dot{y}},\underline{u},t) \leftrightarrow \begin{cases} TRUE & \forall t \geq t^* \\ FALSE & \forall t < t^* \end{cases}$$

. . . t in some non zero interval around the event time t^*.

Example: steam trap (a common device that only allows a liquid phase to flow):



```
IF $(h \geq h_{MIN}) \wedge (\Delta P \geq 0)$THEN
    $v = K_V \sqrt{\Delta P}$
ELSE
    $v = 0$
END
```

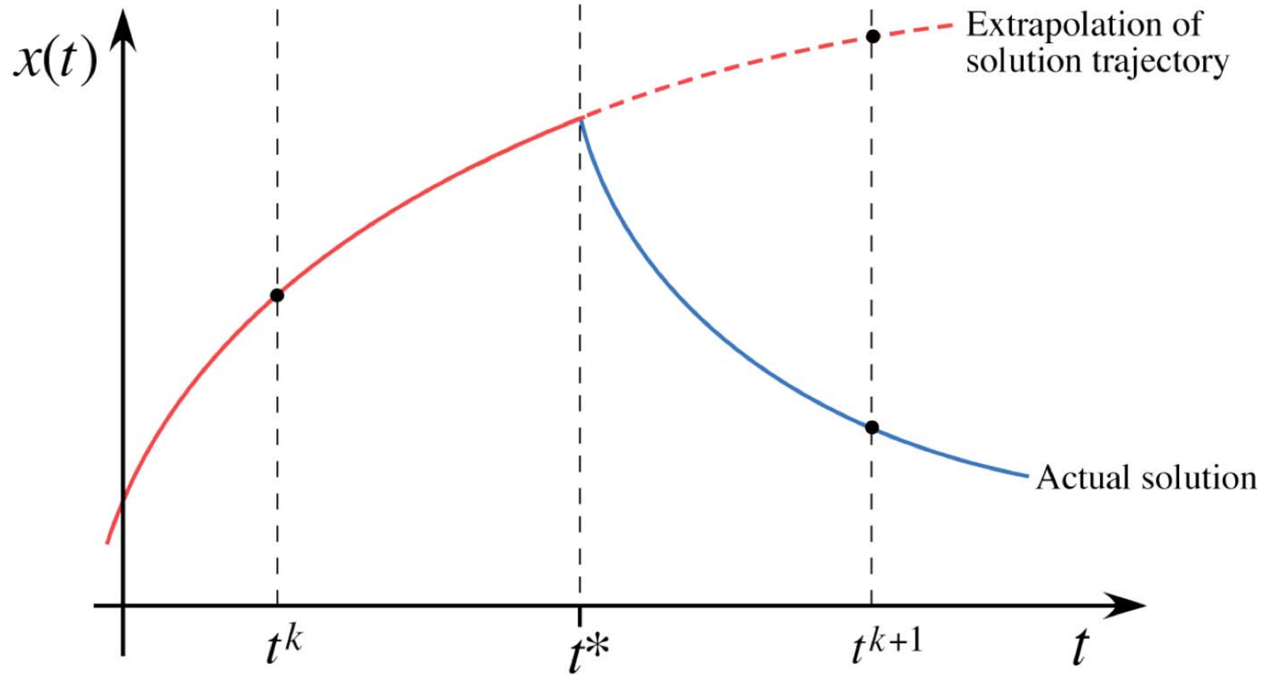
where:

h = liquid level in steam trap
v = velocity of liquid through steam trap
$\wedge$ = logical AND operator. Note: $(h \geq h_{MIN}) \neq (\Delta P \geq 0)$

STATE EVENT LOCATION

Simple minded approach: rely on the automatic error control mechanism of the numerical integration method.

Example: predictor corrector method:



STATE EVENT LOCATION

Algorithm: Ad Hoc:

(a) Take a step from t^k to t^{k+1}. State event occurs at $t*$ in this interval — we do not know $t*$ because we have to integrate to $t*$ to determine when the condition becomes satisfied.

(b) Predictor step gives point on old trajectory. Corrector step gives point on new trajectory.

(c) Error control mechanism uses difference between predictor and corrector steps to estimate the truncation error:

=> if discontinuity in step, error estimate is very large!

Alternatively, the initial guess provided by the predictor may be so poor that the corrector iteration fails to converge.

(d) Error control will mechanism will reject step to t^{k+1}, halve the step length and try the step again.

(e) Eventually. . . after repeated truncation error failures and/or corrector iteration failures, and great reduction in step length . . . a step is taken close enough to $t*$ that the error criterion is satisfied, and integration can then proceed on the new trajectory.

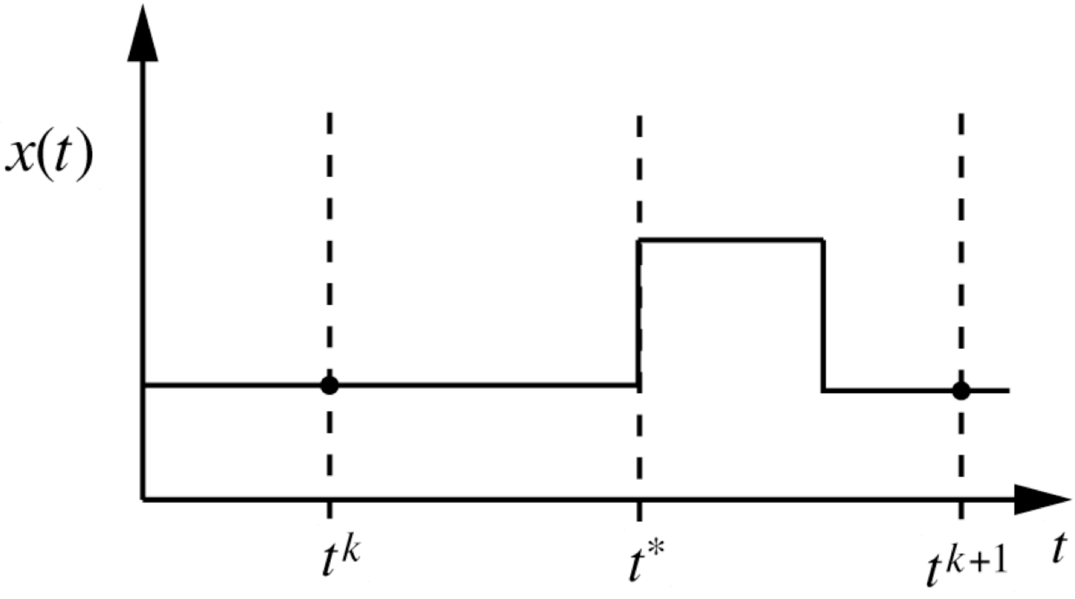
STATE EVENT LOCATION

Problems with Ad Hoc:

i) extremely inefficient — repeated truncation error failures and cutting the step size both drastically slow down the simulation.

ii) DAEs require consistent initialization at a discontinuity - code has to guess that a discontinuity has occurred. However, a rapid transient can also cause repeated truncation error failures and step size reduction - how to distinguish between the two?

iii) as the only way of detecting the state event is the difference between the predictor and corrector steps, it is possible to skip over the discontinuity completely, e.g.,



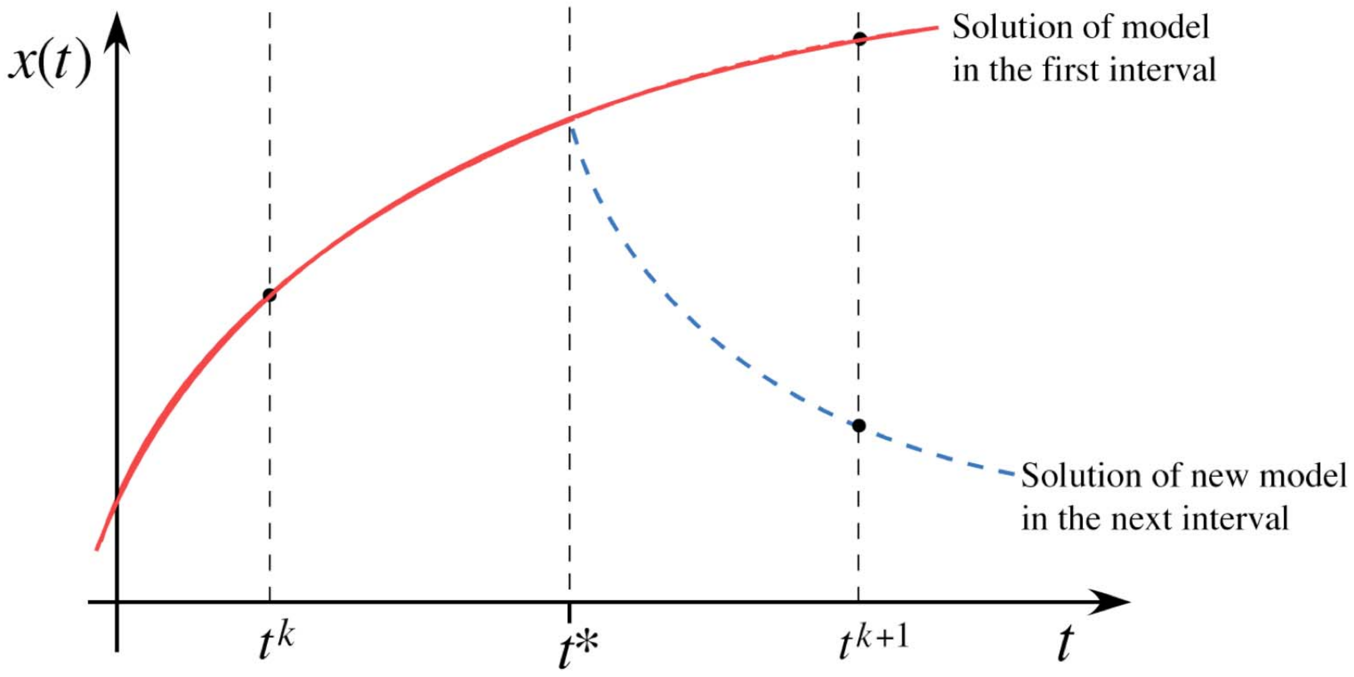
. . . as a new initial value problem must be started at t^* which may have a radically different solution from the predicted trajectory, this is extremely dangerous (particularly in safety related applications).

DISCONTINUITY LOCKING

Numerical methods for ordinary differential equations typically make assumptions concerning the continuity of the solution trajectories:

=> do not expect efficient (or even correct) solutions along discontinuous trajectories!

Solution: discontinuity locking:



DISCONTINUITY LOCKING

Algorithm: Discontinuity Locking:

(a) lock the process model in a sub interval: do not allow equations or input functions to change while numerical integration is being advanced in a sub interval.

(b) advance solution on this continuous trajectory until a converged corrector step beyond the state event is obtained (e.g. we have bounded the time of occurrence)

(c) locate the exact time at which the state event occurs t^* — within the step just taken (i.e., backtrack using interpolation over this step).

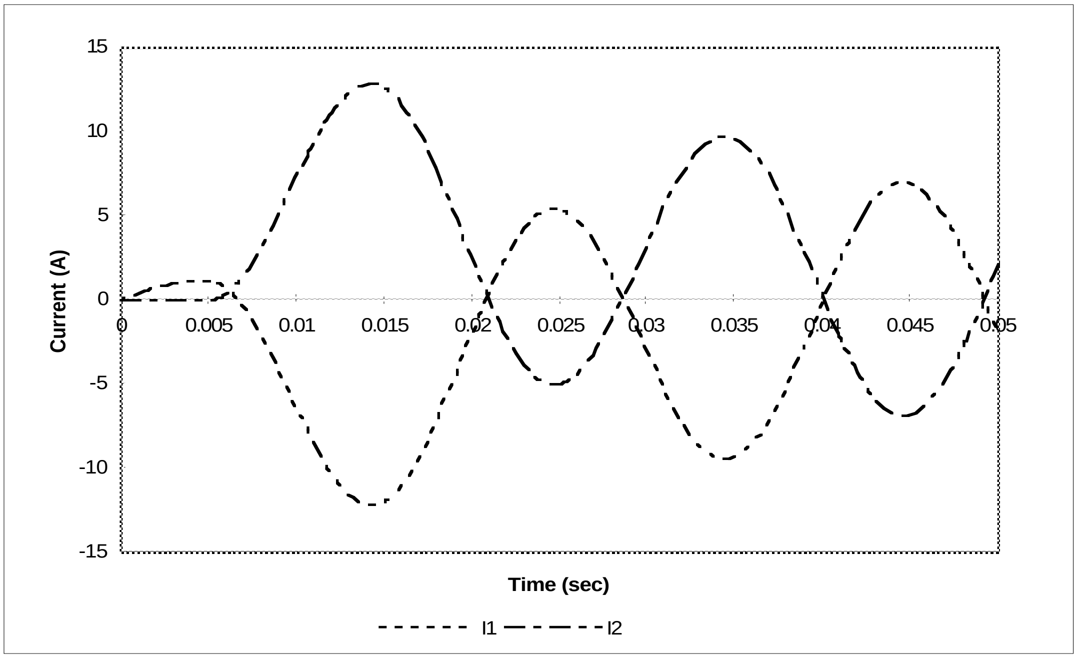
(d) implement the change to the process model at t^*, perform consistent reinitialization at t^*, and restart integration in the new sub interval.

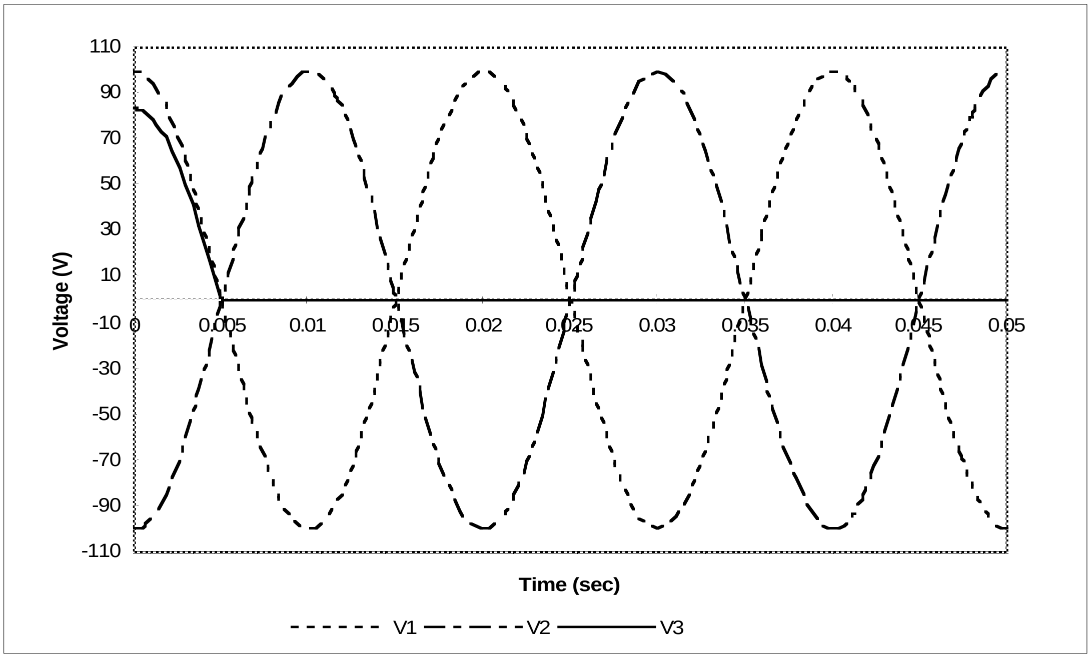
Assumption:

- the differential equations that make up the old model have a mathematically defined solution beyond t^* — even if the solution is not physically meaningful (e.g., negative mole fractions).
- counter examples exist: i.e., a mathematical solution does not exist beyond the event.

Once the time step in which the state event occurs has been located, a variety of methods can be used to locate the exact state event time.

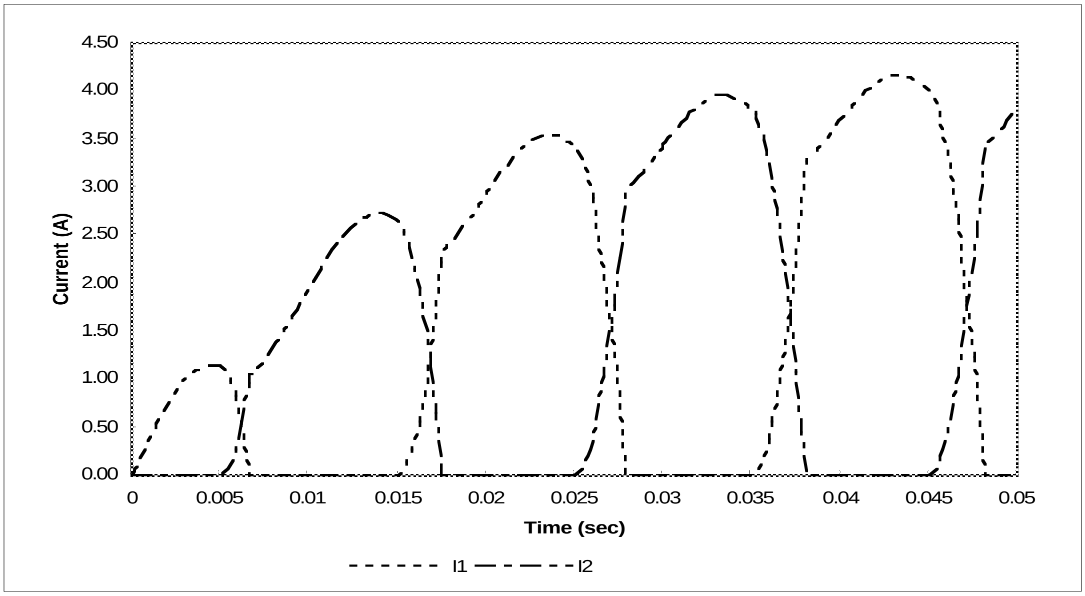
DISCONTINUITY LOCKING - BISECTION ALGORITHM

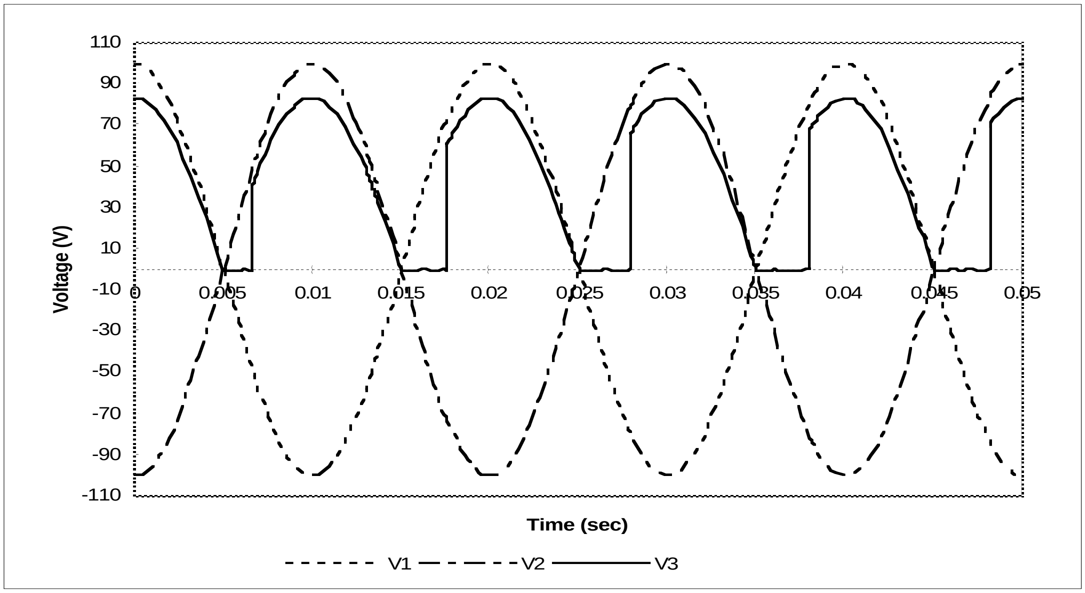




Algorithm implemented in SpeedUp (Pantelides, 1998b), gPROMS.

DISCONTINUITY LOCKING - HIERARCHICAL INTERVAL ARITHMETIC BASED ROOT FINDING





New algorithm implemented in ABACUSS (Park and Barton, 1996).

Which answer is correct? Park and Barton algorithm guarantees correctness.

COMBINED DISCRETE/CONTINUOUS SIMULATION[1]

As the number of discontinuities increases in a dynamic simulation, we are really performing what is called a *combined discrete/continuous simulation* — i.e., we are simulating a system whose dynamic behaviour has both significant continuous and discrete aspects:

- continuous aspects — physico-chemical phenomena modelled by differential equations (material, energy, momentum balances)
- discrete aspects:
  - (a) discontinuities in physical behaviour — e.g., phase changes, flow transitions
  - (b) control actions — e.g., open/close valves, start/stop pumps, open/close control loops.

. . . a dynamic simulation tool must be able to cope with descriptions of both aspects of process behaviour (see ABACUSS workshop).

[1] for a more comprehensive discussion of this subject see Barton and Pantelides (1994)

WHEN IS COMBINED SIMULATION IMPORTANT?

Whenever the system state moves far from steady-state:

- modelling sequential operations — e.g., start-up, shut-down, feedstock changeover. Must model the control actions in the operating procedures, and the physical transitions the system goes through.

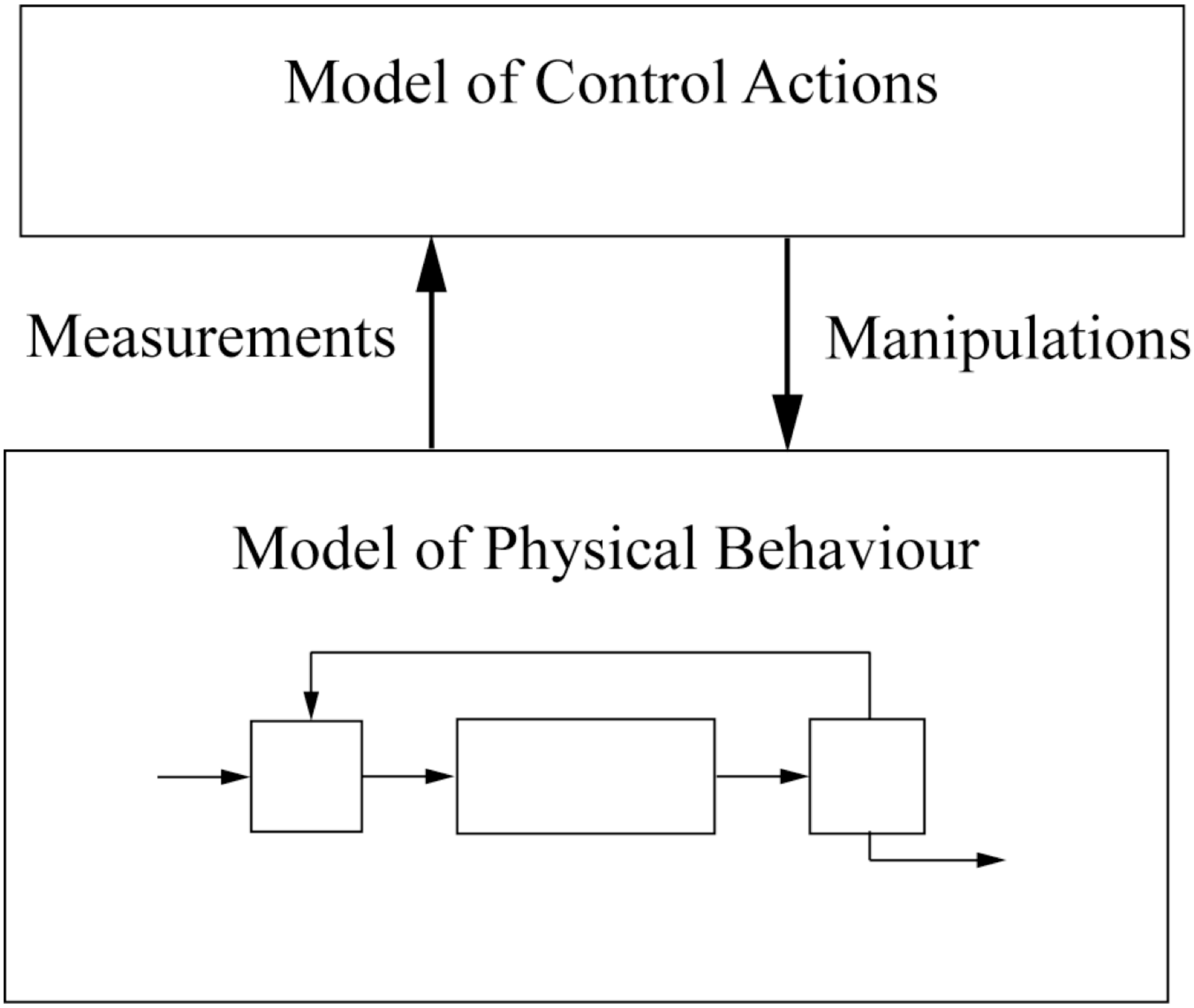
- major process disturbances — model interaction of process, regulatory control system, and automatic protective system. Process model can change (e.g. vessel running dry) and control/safety systems can impose discrete actions.

- periodic processes — e.g., pressure swing absorption.

- batch processes — must model complex operating policies composed of discrete actions.

We can see that almost all interesting process simulations are combined discrete/continuous - certainly it provides a convenient unified strategy for process simulation.

PROCESS MODEL DECOMPOSITION



Model of Control Actions

Measurements

Manipulations

Model of Physical Behaviour

Key concept:

Discrete aspects of process behaviour can be modelled by manipulation of the functional form of the equations describing the process model, e.g. changing the functional form of:

$$\underline{f}(\dot{\underline{x}}, \underline{x}, \underline{y}, \underline{u}, t) = 0$$
$$\underline{u} = \underline{u}(t)$$

PHYSICO-CHEMICAL DISCONTINUITIES

Any description of the physical behaviour should include those discrete changes to the process model that occur as a result of physical phenomena:

(a) so that the model of physical behaviour is compact and complete.

(b) the model will automatically adjust its functional form in response to current conditions without intervention from the user.

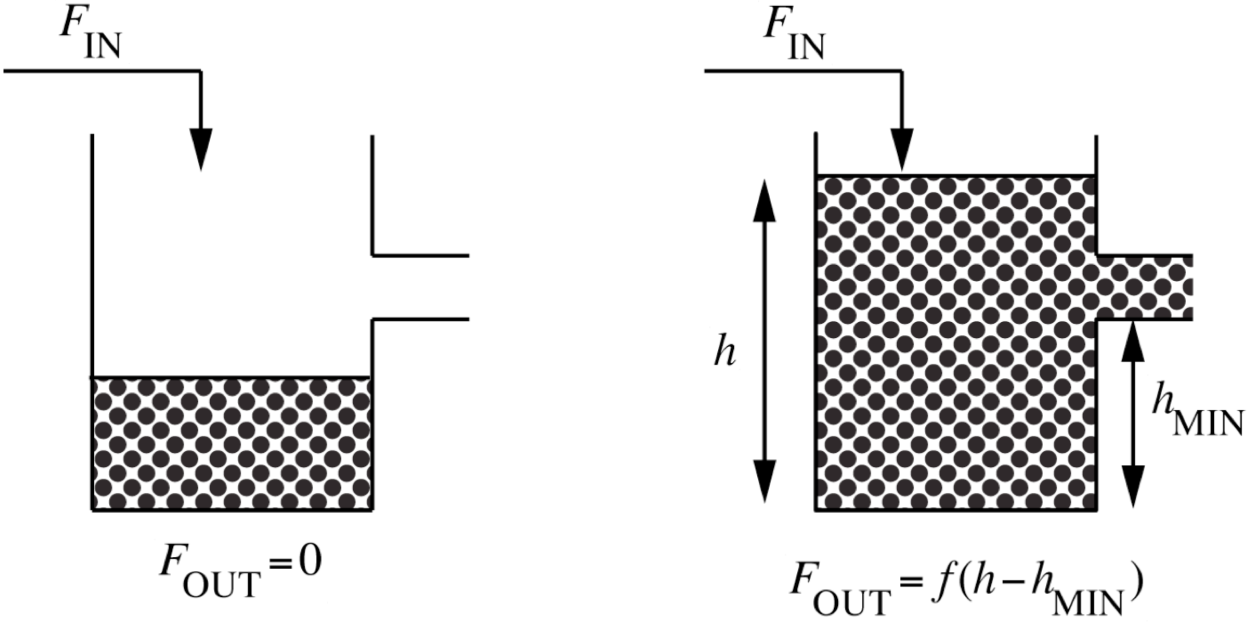
General observation:

> Physico-chemical discontinuities can be modelled changing subsets of the equations that make up the process model in response to changes in process conditions.

. . . therefore physico-chemical discontinuities occur as a consequence of state events.

PHYSICO-CHEMICAL DISCONTINUITIES EXAMPLE

Example: tank with an overflow pipe:



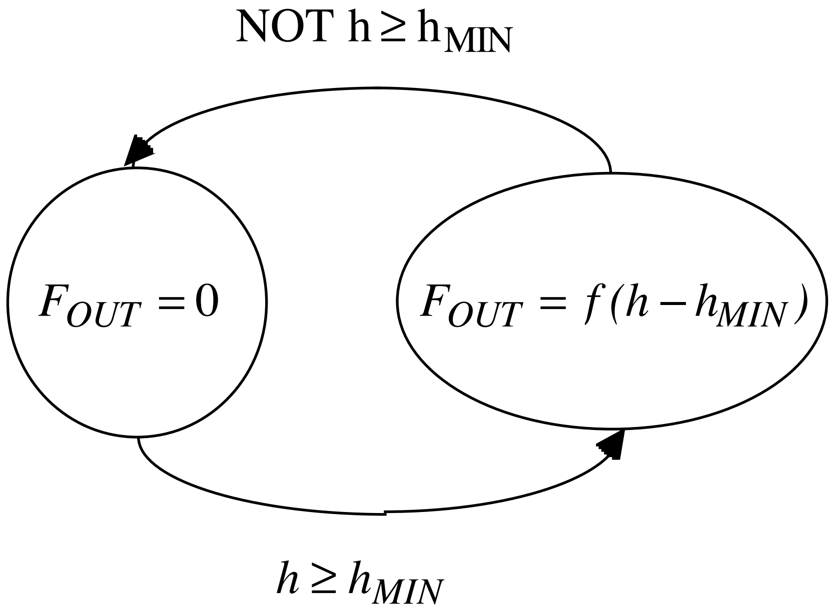
This can be represented in an equation based modelling language by:

```
IF $h \geq h_{MIN}$ THEN
    $F_{OUT} = f(h - h_{MIN})$
ELSE
    $F_{OUT} = 0$
END
```

PHYSICO-CHEMICAL DISCONTINUITIES EXAMPLE

Finite automaton representation of the model state — a digraph:

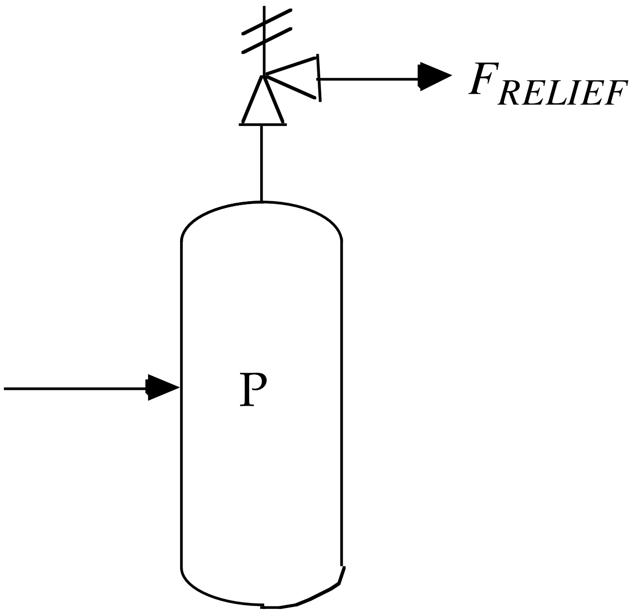
- nodes represent model in a particular state (characterized by a set of equations)
- arcs represent discrete transition from one state to another — transition triggered by the occurrence of a state event.



. . . IF equations represent symmetric reversible discontinuities: the condition for the forward transition is the negation of the condition for the reverse transition.

PHYSICO-CHEMICAL DISCONTINUITIES EXAMPLE

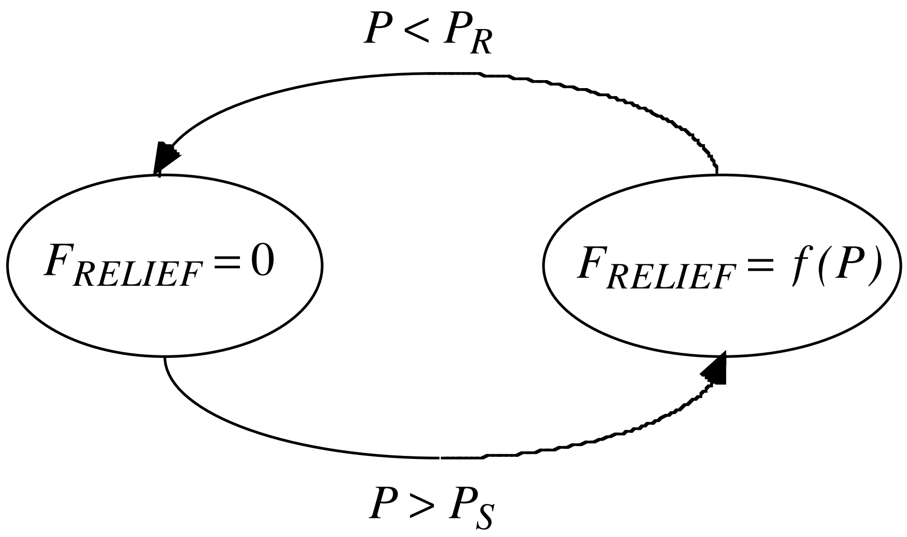
Example: pressure vessel fitted with safety relief valve:



Mechanical safety device: relief valve opens at the set pressure (P_S), but closes at a (lower) reseat pressure (P_R).

PHYSICAL-CHEMICAL DISCONTINUITIES EXAMPLE

Finite automaton:



The behaviour of this system exhibits hysteresis (memory): the equations describing the system are a function of both the current value of the process variables *and* the previous behaviour of the system.

For example, if the pressure is between the reseat and set pressures the system may be in either state — the actual state depends on what has happened in the past. The initial state of the model must be defined as part of the initial conditions of the system.

REFERENCES AND FURTHER READING

Literature Cited

Ahmad B.S. and P.I. Barton, “Solvent Recovery Targeting for Pollution Prevention in Pharmaceutical and Specialty Chemicals Manufacturing”, *AIChE Symposium Series*, **90**(303), 59-73, 1994.

Allgor R.J., M.D. Berrera, P.I. Barton and L.B. Evans, “Optimal Batch Process Development”, *Computers Chem. Engng.*, **20**(6/7), 885-896, 1996.

Aspen Technology, “SPEEDUP User Manual”, volumes I and II, version 5.4, Cambridge, Massachusetts, 1993.

Barton, P.I., “The Modelling and Simulation of Combined Discrete/Continuous Processes”, Ph.D. Thesis, University of London, 1992.

Barton, P.I. and C.C. Pantelides, “Modelling of Combined Discrete/Continuous Processes”, *AIChE Journal*, **40**, 966-979, 1994.

Barton, P.I., R.J. Allgor and W.F. Feehery, “DSL48S A Large-Scale Differential-Algebraic and Parametric Sensitivity Solver”, ABACUSS Project Report no. 97/1, 22nd January 1997.

Campbell S.L., “Singular Systems of Differential Equations”, Pitman, Boston, 1980.

Campbell S.L., “Singular Systems of Differential Equations II”, Pitman, Boston, 1982.

Elmquist, H., “A Structured Model Language for Large Continuous Systems”, Ph.D. Thesis, Lund Institute of Technology, 1978.

Feehery W.F. and P.I. Barton, “A Differentiation-Based Approach to Simulation and Dynamic Optimization with High-Index Differential-Algebraic Equations”, In M. Berz, C. Bischof, G. Corliss and A. Griewank, editors, *Computational Differentiation*, SIAM, 1996.

Gear C. W., “Simultaneous Numerical Solution of Differential-Algebraic Equations”, IEEE Transactions on Circuit Theory, **CT-18**, p. 89, 1971.

Gear C. W. and L.R. Petzold, “ODE Methods for the Solution of Differential/Algebraic Systems”, *SIAM J. Numer. Anal.*, **21**(4), 716-728, 1984.

Griewank A. and G.F. Corliss, editors, “Automatic Differentiation of Algorithms: Theory, Implementation, and Application”, SIAM, Philadelphia, 1991.

Jarvis, R.B. and C.C. Pantelides, “DASOLV a differential-algebraic equation solver”, technical report, Imperial College, London, 1992.

Kroner, A., W. Marquardt and E.D. Gilles, "Computing Consistent Initial Conditions for Differential-Algebraic Equations", *Computers Chem. Engng.,* **16S**, pp. S131-S138, 1992.

Lefkopoulos A. and M.A. Stadtherr, "Index Analysis of Unsteady-State Chemical Process Systems - I: An Algorithm for Problem Formulation", *Computers Chem. Engng.*, **17**, pp. 399-413, 1993.

Marquardt W., "Trends in computer-aided process modeling", In *Proceedings of the 5th International Symposium on Process Systems Engineering*, Kyongju, Korea, 1994.

Mattsson, S.V. and G. Soderlind, "Index reduction in differential-algebraic equations using dummy derivatives", *SIAM J. Sci. Stat. Comput.*, **14**, pp. 677-692, 1993.

Modell, M. and R.C. Reid, "Thermodynamics and its Applications", 2nd Edition, Prentice-Hall, 1983.

Oh M., "Modelling and Simulation of Combined Lumped and Distributed Processes", Ph.D. Thesis, University of London, 1995.

Pantelides, C.C., "The consistent initialization of differential-algebraic systems", *SIAM J. Sci. Stat. Comput.*, **9**, pp. 213-231, 1988a.

Pantelides, C.C., "SpeedUp - recent advances in process simulation", *Computers and Chem. Engng.*, **12**, pp. 745-755, 1988b.

Pantelides, C.C., D. Gritsis, K.R. Morison and R.W.H. Sargent, "The mathematical modelling of transient systems using differential-algebraic equations", *Computers Chem. Engng.,* **12**, pp. 449-454, 1988.

Park T. and P.I. Barton, "State Event Location in Differential-Algebraic Models", *ACM Transactions on Modeling and Computer Simulation*, **6**(2), 1996.

Perkins J.D., R.W.H. Sargent and R. Vasquez-Roman, "Computer generation of process models", In *Proceedings of the 5th International Symposium on Process Systems Engineering*, Kyongju, Korea, 1994.

Petzold, L.R., "Differential/Algebraic Equations are not ODE'S", *SIAM J. Sci. Stat. Comput.*, **3**(3), 367-384, 1982.

Petzold, L.R., "A description of DASSL: a differential/algebraic system solver", in Scientific Computing, eds. R.S. Stepleman et. al., North-Holland, Amsterdam, pp. 65-68, 1983.

Ponton, J.W. and P.J. Gawthrop, "Systematic Construction of Dynamic Models for Phase Equilibrium Processes", *Computers Chem. Engng.*, **15**, pp. 803-808, 1991.

Strauss J. C., "The SCi continuous system simulation language (CSSL)", *Simulation*, **9**, 281-303, 1967.

Stephanopoulos G., G. Henning, and H. Leone, "MODEL.LA. a modeling language for process engineering - I. The formal framework", *Computers Chem. Engng.*, **14**, pp. 813-846, 1990.

Stephanopoulos G., G. Henning, and H. Leone, "MODEL.LA. a modeling language for process engineering - II. Multifaceted modeling of processing systems", *Computers Chem. Engng.*, **14**, pp. 847-869, 1990.

Tolsma J.E. and P. I. Barton, "DAEPACK: A Flexible Environment for Open Modeling", AIChE Annual Meeting, Dallas, Texas, 31st October – 3rd November, 1999.

Industrial Experience with Dynamic Simulation

Cole J.B. and K.B. Yount, "Applications of Dynamic Simulation to Industrial Control Problems", In *Advances in Instrumentation and Control*, **48**(2), pp. 1337-1344, Chicago, 1993.

Debelak K.A., G. Karsai, G. Chen, S. Zitney and F.O. DeCaria, "Application of Dynamic Simulation to an Industrial Process Control System", AIChE Annual Meeting, Miami, November 1995.

Debling J.A, G.C. Han, F. Kuijpers, J. VerBurg, J. Zacca, and W.H. Ray, "Dynamic Modeling of Product Grade Transitions for Olefin Polymerization Processes", *AIChE Journal*, **40**(3), pp. 506-520, 1994.

Grassi V.G., "Dynamic Simulation as a Tool to Integrate Process Design and Process Control", In *Advances in Instrumentation and Control*, **48**(2), pp. 1346-1365, Chicago, 1993.

Longwell E.J., "Dynamic Modeling for Process Control and Operability", In *Advances in Instrumentation and Control*, **48**(2), pp. 1323-1336, Chicago, 1993.

Mani S., S.K. Shoor and H.A. Pedersen, "Experience with a Simulator for Training Ammonia Plant Operators", *Plant/Operations Progress*, **9**(1), pp. 6-10, 1990.

Naess, L., A. Mjaavatten and J.O. Li, "Using Dynamic Process Simulation from Conception to Normal Operation of Process Plants," *Computers Chem. Engng.* **17**, pp. 585-600, 1993.

Zitney S.E., L. Brull, L. Lang and R. Zeller, "Plantwide Dynamic Simulation on Supercomputers: Modeling a Bayer Distillation Process", *AIChE Symposium Series*, **91**(304), pp. 313-316, 1995.

Modelling of Process Dynamics

Franks R. G. E., "Modelling and Simulation in Chemical Engineering", Wiley 1972.

Ingham J, I. J. Dunn, E. Heinzle, J. E. Prenosil, "Chemical Engineering Dynamics. Modelling with PC Simulation", VCH, Weinheim, 1994.

Luyben W. L., “Process Modeling, Simulation and Control for Chemical Engineers”, chapters 2 and 3, 2nd edition, McGraw-Hill, 1990.

Review Papers (recent)

Barton, G.W. and J.D. Perkins, "Modelling and simulation in process operation," In G.V. Reklaitis and H.D. Spriggs, editors, Foundations of Computer Aided Process Operations, pp. 287-316, Cache-Elsevier, 1987.

Marquardt, W., "Dynamic process simulation — recent progress and future challenges," Proceedings of CPC IV, Texas, USA, 1991.

Pantelides, C.C. and P.I. Barton, "Equation-Oriented Dynamic Simulation: Current Status and Future Perspectives," *Computers Chem. Engng.*, **17S,** pp. S263-S285, 1993.

Marquardt W., “Numerical Methods for the Simulation of Differential-Algebraic Process Models”, In R. Berber, editor, *Methods of Model-based Process Control*, pp. 41-79, NATO-ASI Series, Kluver Press, 1995.

Numerical solution of initial value problems in ODEs and DAEs

Ascher, U.M. and L.R. Petzold, "Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations", SIAM, Philadelphia, 1998. (*this is the best introductory text to get*).

Lambert, J.D., "Numerical Methods for Ordinary Differential Systems — the Initial Value Problem," Wiley, 1991.

Brenan, K.E., S.L. Campbell and L.R. Petzold, "Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations," SIAM, Philadelphia, 1996.

Hainer, E., S.P. Norsett and G. Wanner, "Solving Ordinary Differential Equations: Volume 1 - Nonstiff problems; Volume 2 - Stiff and differential-algebraic problems," Springer Verlag, 1987-1991.

Extensions of Sequential Modular Approach to Dynamic Simulation

Aylott M.R., J.W. Ponton and D.H. Lott, “Development of a Dynamic Flowsheeting Program”, ”, Proceedings 2nd International Symposium on Process Systems Engineering, Institute of Chemical Engineers, Rugby, England, 1985.

Cook W.J., J. Klatt, C.B. Brosilow, “Simulation of Large Scale Dynamic Systems - II. A Modular Simulator for the Dynamics of Distillation Systems”, *Computers Chem. Engng.*, **11**(3), 255-264, 1987.

Hillestad M. and T. Hertzberg, “Dynamic Simulation of Chemical Engineering Systems by the Sequential Modular Approach”, *Computers Chem. Engng.*, **10**, 377-388, 1986.

Fagley J.C. and B. Carnahan, “The Sequential-Clustered Method for Dynamic Chemical Plant Simulation”, ”, *Computers Chem. Engng.*, **14**(2), 161-177, 1990.

Laganier F., J.M. LeLann, X. Joulia and B. Koehret, “Use of the Dynamic Sensitivity Matrix Concept for the Efficient Solution of Dynamic Process Simulation”, Proceedings 4th International Symposium on Process Systems Engineering, III.12.1, Montebello, Quebec, August 5-9th, 1991.

Liu Y.C. and C.B. Brosilow, “Simulation of Large Scale Dynamic Systems - I. Modular Integration Methods”, *Computers Chem. Engng.*, **11**(3), 241-253, 1987.

Ponton J.W., “Dynamic Process Simulation Using Flowsheet Structure”, *Computers Chem. Engng.*, **7**(1), 13-17, 1983.

Ponton J.W. and V. Vasek, “A Two-Level Approach to Chemical Plant and Process Simulation”, *Computers Chem. Engng.*, **10**(3), 277-286, 1986.

Ponton J.W., E.M. Johnston, J.M. Forsyth, A. Matheson, and F.M. Rutherford, “Real Time Dynamic Simulation Using Networked Computers”, *Computers Chem. Engng.*, **13**(11/12), 1245-1253, 1989.

Shinohara T., “A Block Structured Approach to Dynamic Process Simulation”, In: *Computer Aided Process Operations*, G.V. Reklaitis and H.D. Spriggs, eds, Cache-Elsevier, 1987.