Solving flowsheets

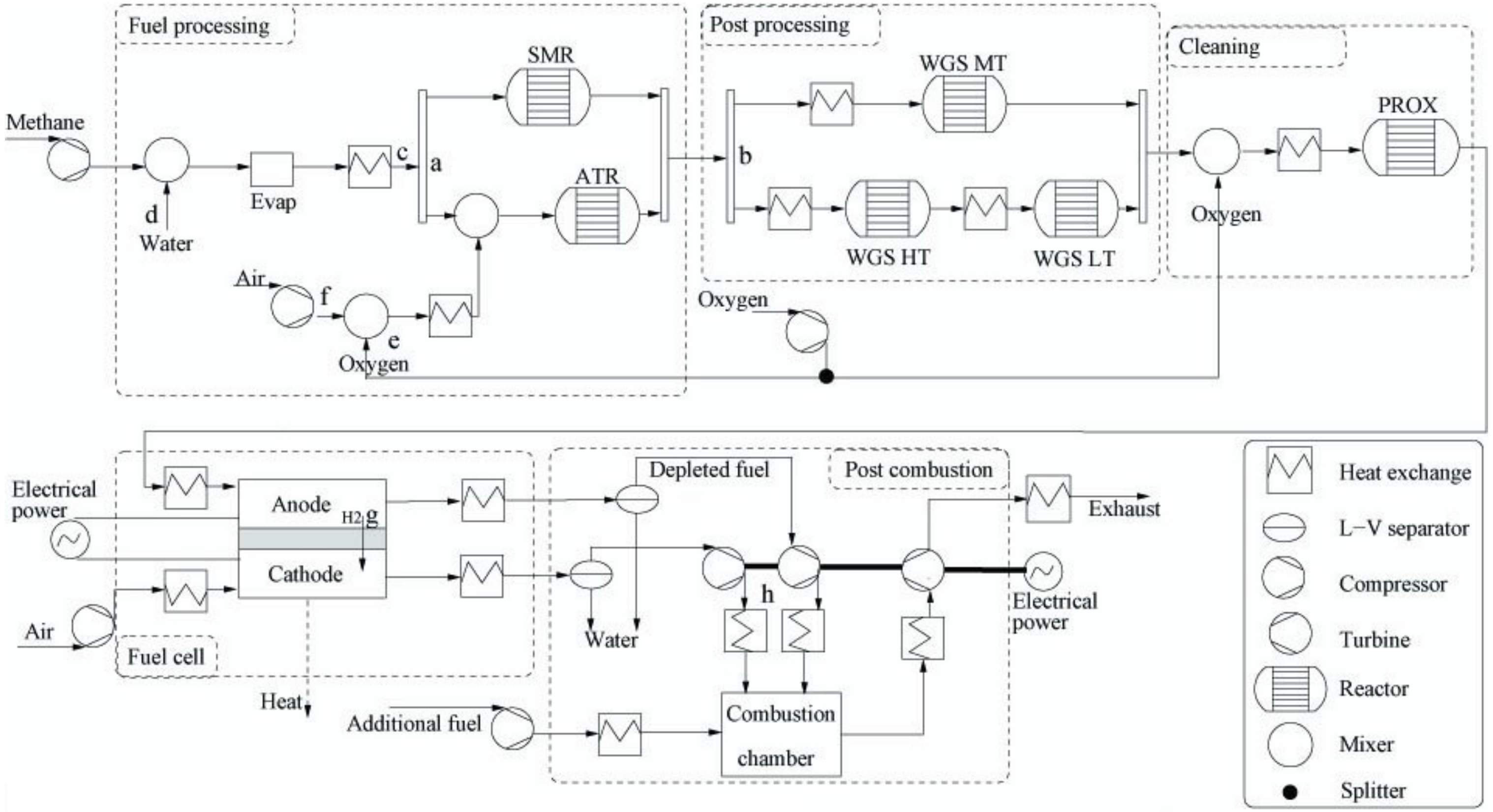
Prof François Marechal
Dr Shivom Sharma

28.03.2022

EPFL
IPESE
Industrial Process and Energy Systems Engineering

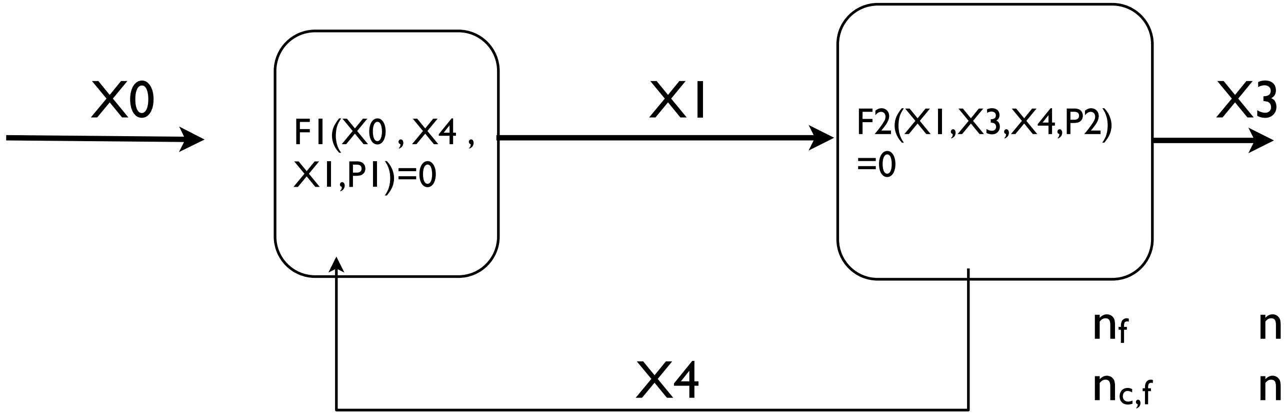
Flowsheeting

- Process models
  - Process Units
  - Interconnections



Flowsheets : Degrees of freedom

- Flowsheet = interconnected modules



n_f nb of flows
$n_{c,f}$ nb of compound in flow f
n_u nb of units
$n_{p,u}$ nb of parameters in unit u

N_e : Equations

$F(X,P)=0$: Models
$X_s - X^* = 0$: Specifications (system)
$P_s - P^* = 0$: Specifications (unit parameters)
$X_i - X_j = 0$: Links (unit interconnections)

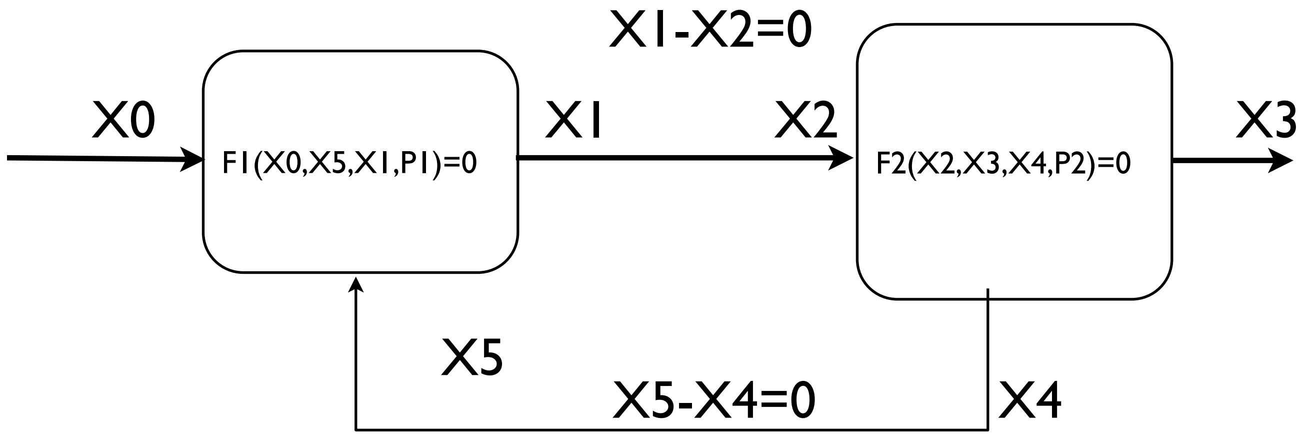
N_v : Variables

$n_f * (2 + n_{c,f})$ state of the flows
$n_u * n_{p,u}$ parameters of unit models

Degrees of freedom : N_{DOF} : $N_v - N_e$

Solving Flowsheets : simultaneous approach

- Flowsheet = interconnected modules



Simultaneous resolution : Solve a set of non linear equations

$F(X,P)=0$: Models (= a set of concatenated equations)
$X_s-X^*=0$: Specifications (= a set of specifications for fixing the system DOF)
$P_s-P^*=0$: Specifications (= a set of unit parameters specs)
$X_i-X_j=0$: Links (unit interconnections (reduces the DOF of system))

Solving method => Newton Raphson : solve a set of non linear equation

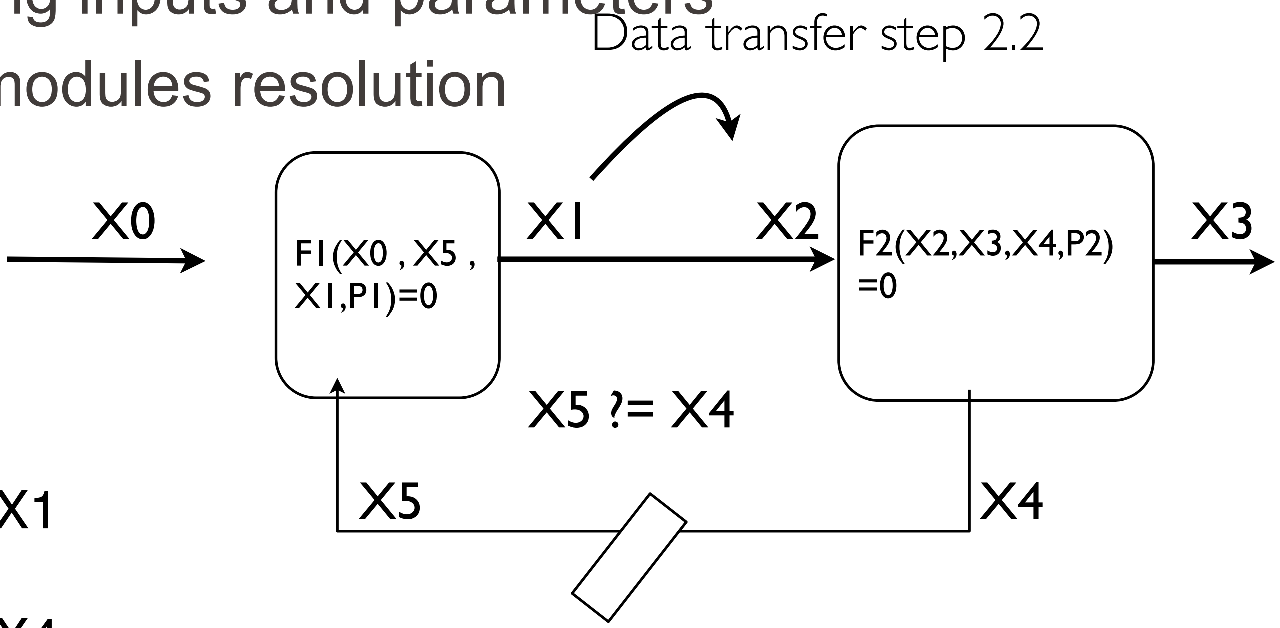
In reality X1=X2 is solved explicitly and only X1 is used
F2((X2,X3,X4,P2)=0 and X1-X2=0 becomes F2(X1,X3,X4,P2)=0

Solving Flowsheets : sequential

- sequential approach = interconnected modules
  - calculate output knowing inputs and parameters
  - define a sequence of modules resolution
  - explicitly solve links

Loops (recycle in flowsheets)

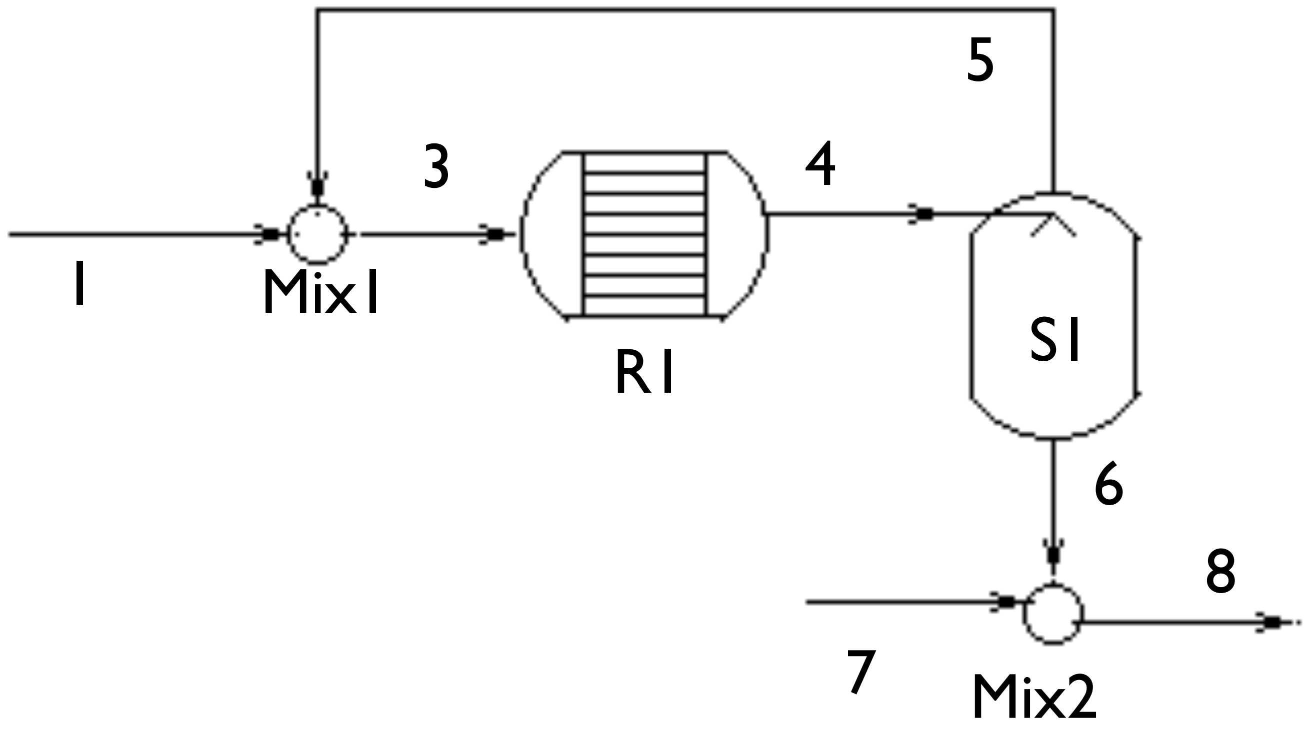
0. Locate X5
1. Guess X5
   2. 2.1 Solve unit 1 => X1
      2.2 X2=X1
   3. 3.1 Solve unit 2 => X4
   4. 4.1 Test X5 ?= X4
      4.2 yes : => out
      4.2 No : goto 5
   5. Propose a new value to X5
   & Goto 2.



Tearing loops
Loops are identified when a value that is needed by a module is the result of the calculation of the module in the sequence

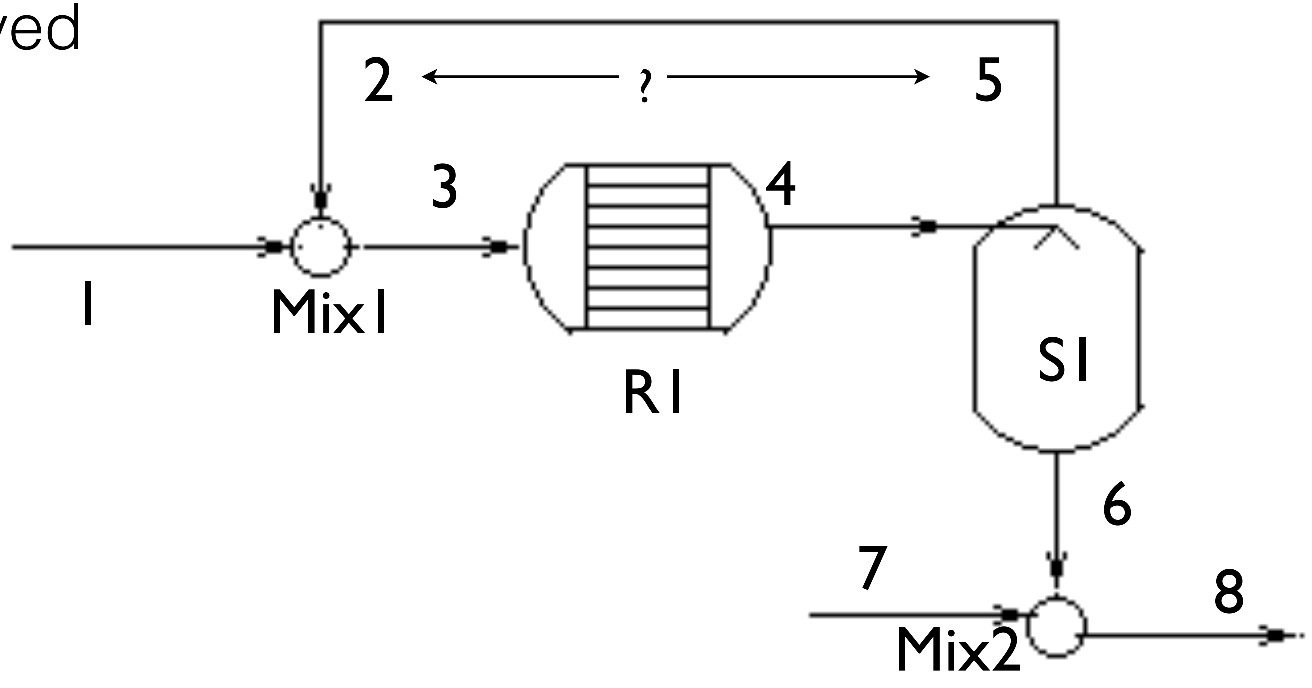
Step 1 : describe what you want to know

- Name the streams
  - Your goal is to know the associate state $(\dot{m}, x_i, T, P)$
- Name the units
  - The units are transforming states



Step 1l : define a sequence

- A unit model $f_u(X_{in}^u)$ calculates the output knowing the input :
  $X_{out}^u = f_u(X_{in}^u)$
- Order of resolution sequence
  1. set all units to unsolved and set $X_{in}^{Flowsheet}$ as known
  2. identify the unit **u** that is not solved with all X_{in}^u known :
  3. Calculate $X_{out}^u = f_u(X_{in}^u)$
  4. mark X_{out}^u as known and **u** as solved
  5. back to 1.



Solving flowsheets : tearing loops

Guess 2

solve mix1 => 3
solve R1 => 4
solve S1 => 5,6
test 2 = 5 ?

No → Update 2

Yes

solve Mix2=>8

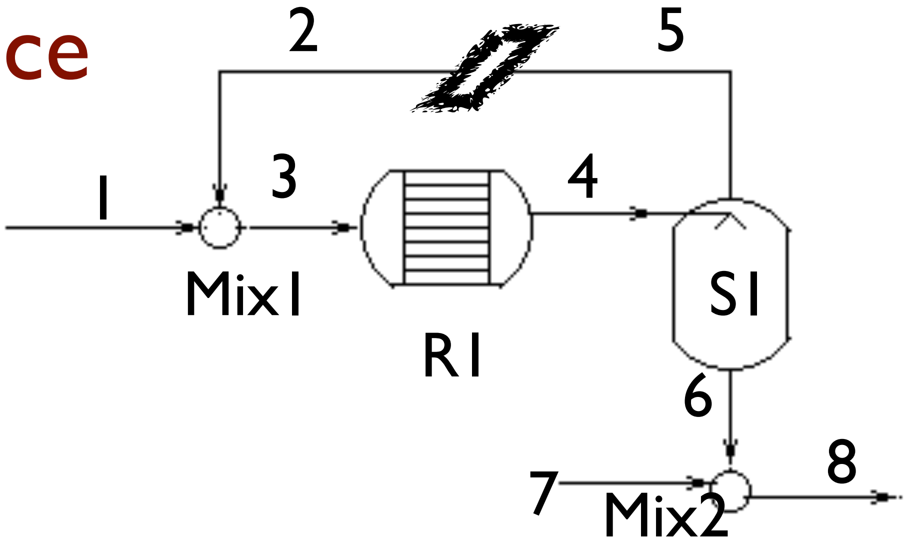
Tearing 2/5

Sequence

Loop : Iterations

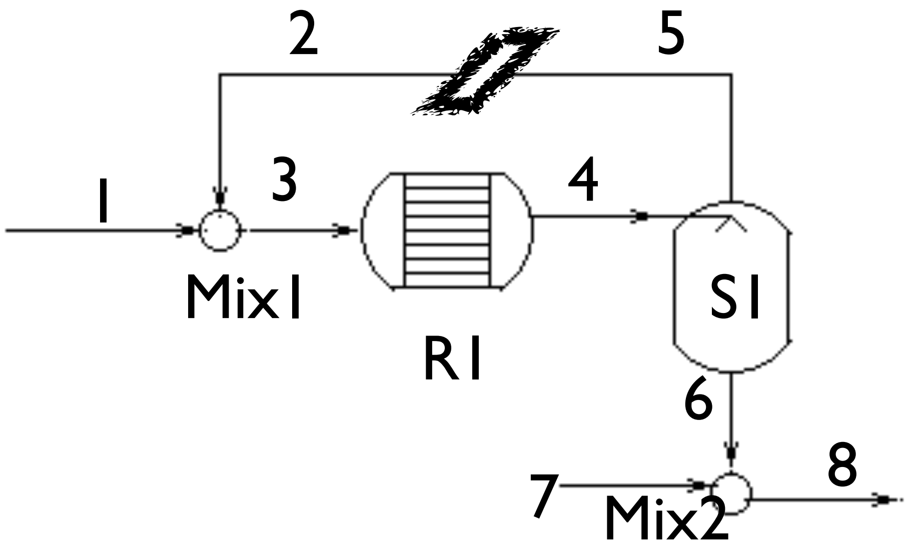
Test convergence

Continue Sequence



Resolution sequence

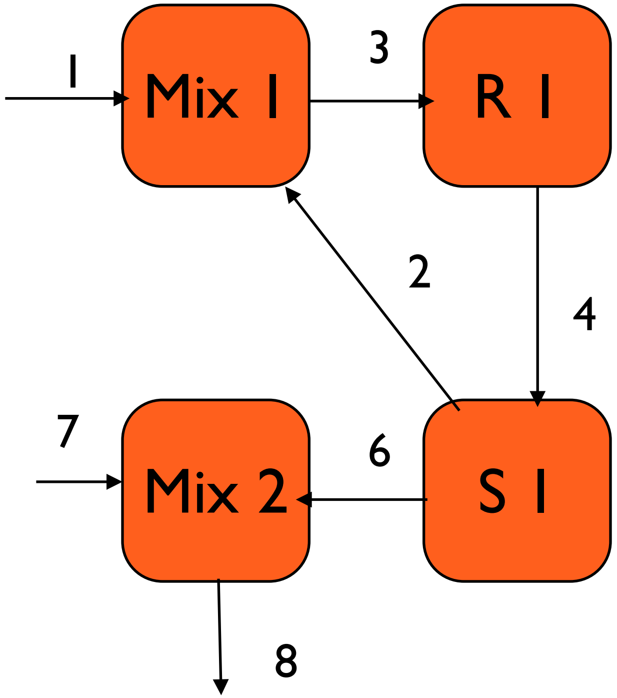
- Define an **ordered** list (sequence) of **units** solving
- Define the order of the **streams** calculated by the **units**
- Identify the tear streams (open loops)
- Consider loops as units
- Define the convergence loops
    - choose a convergence meth
    - Define sequences in loops
    - Define sequence of loops



Systematic definition of sequences : Graph representation

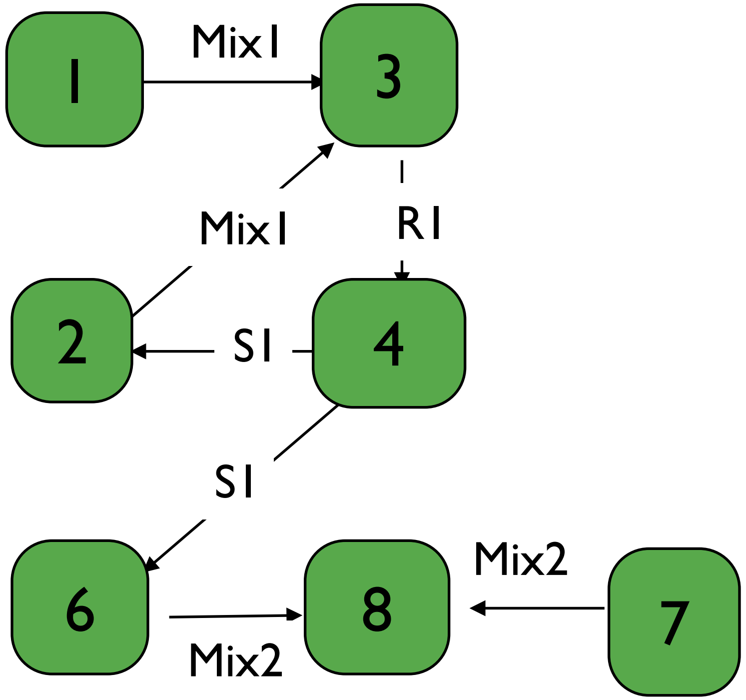
Rheogram

primal representation



Streams to compute units
Variables to compute equations

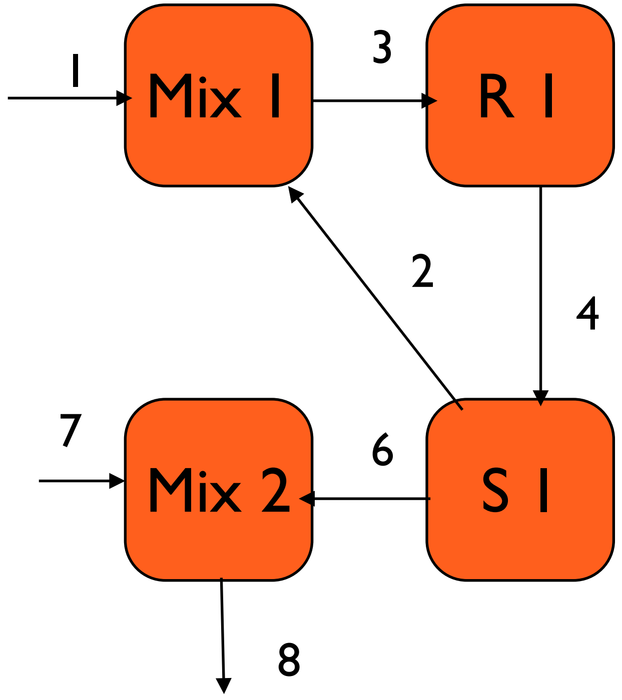
dual representation



Units to compute streams
Equations to compute variables

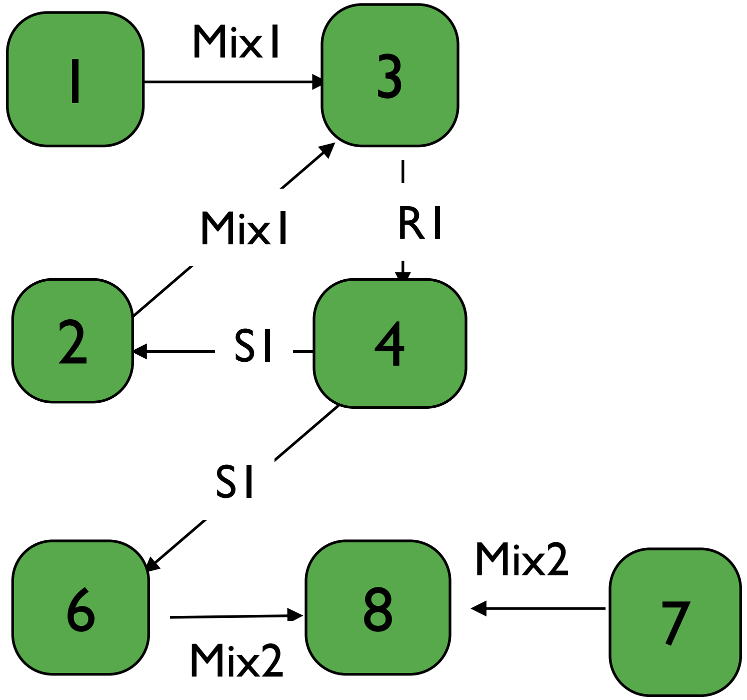
Systematic definition : Defining anteriors

primal representation



Streams to compute units
Variables to compute equations

dual representation



Units to compute streams
Equations to compute variables

- Stream 3 : needs 1 & 2 and is obtained by solving Mix1
  - streams 1 and 2 are anteriors of stream 3
  - => Stream 3 can be replaced by 1 and 2

Mottard Algorithm

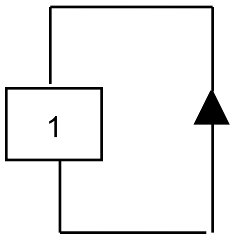
From streams anteriority table

For each stream : what are the streams needed to compute its value by solving 1 unit

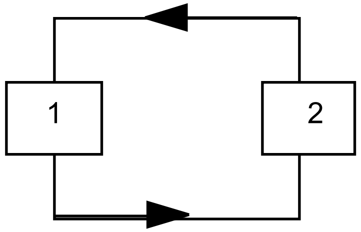
1.- Suppress streams that have no anteriors (you know them)

2.- Replace the streams that have only one anterior by their anterior

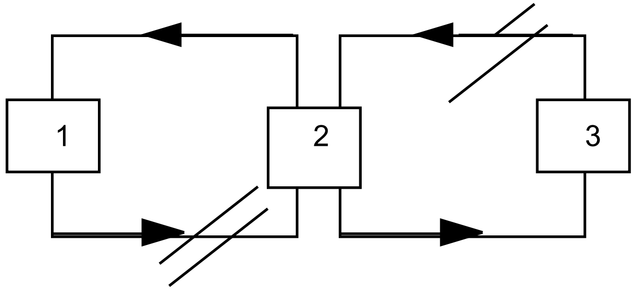
3.- when a stream depends of himself
Open loop by tearing
mark the teared stream
consider that the teared stream has no anterior back to 1.



4.- when an anterior stream depends of his anterior
Open loop by tearing



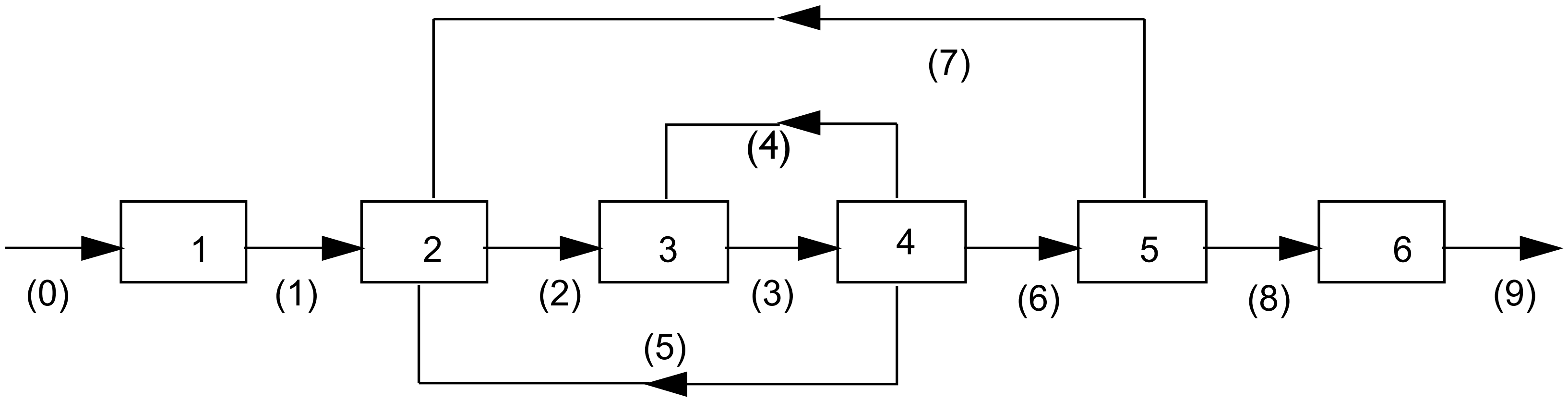
5.- Tear streams with the highest number of anterior



A teared stream has no anterior
Guess the value and restart in 1

APPLICATION OF THE MOTTARD ALGORITHM

- Define the sequence of this flow sheet : assumption $X_{out}^{u} = f_u(X_{in}^{u})$



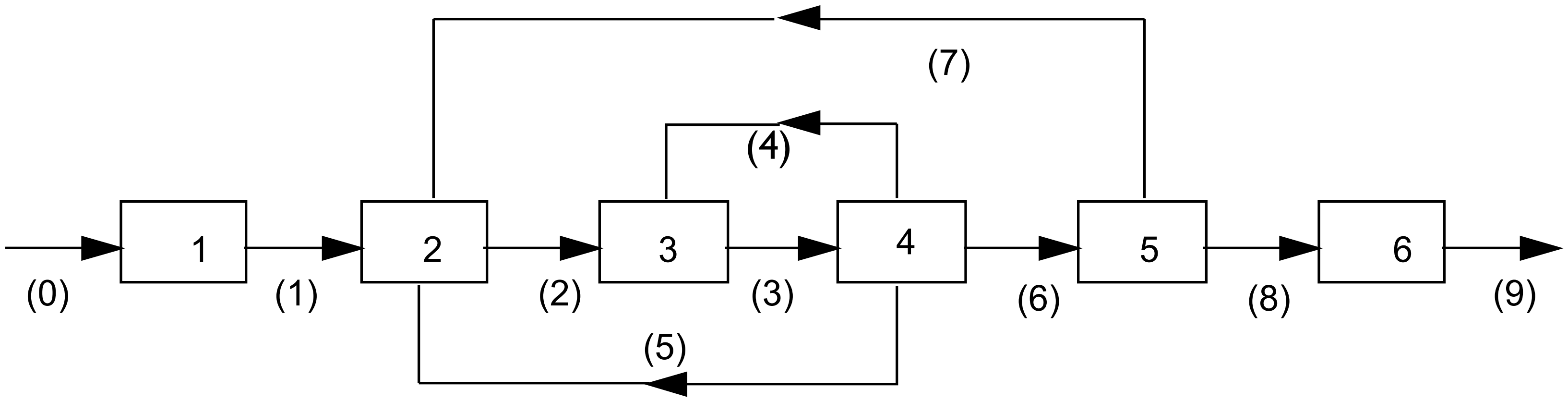
STREAMS	ANTERIOR STREAMS	
0	-	-
1	0	-
2	1,5,7	1,5,7
3	2,4	2,4
4	3	3
5	3	3
6	3	3
7	6	6
8	6	6

Eliminate streams with no anterior

STREAMS	ANTERIOR STREAMS
2	5, 7
3	2,4
4	3
5	3
6	3
7	6
8	6

Eliminate streams with no anterior

APPLICATION OF THE MOTTARD'S ALGORITHM



STREAMS	ANTERIOR STREAMS	
2	5,7	3,3
3	2,4	2,3
4	3	3
5	3	3
6	3	3
7	6	3
8	6	3

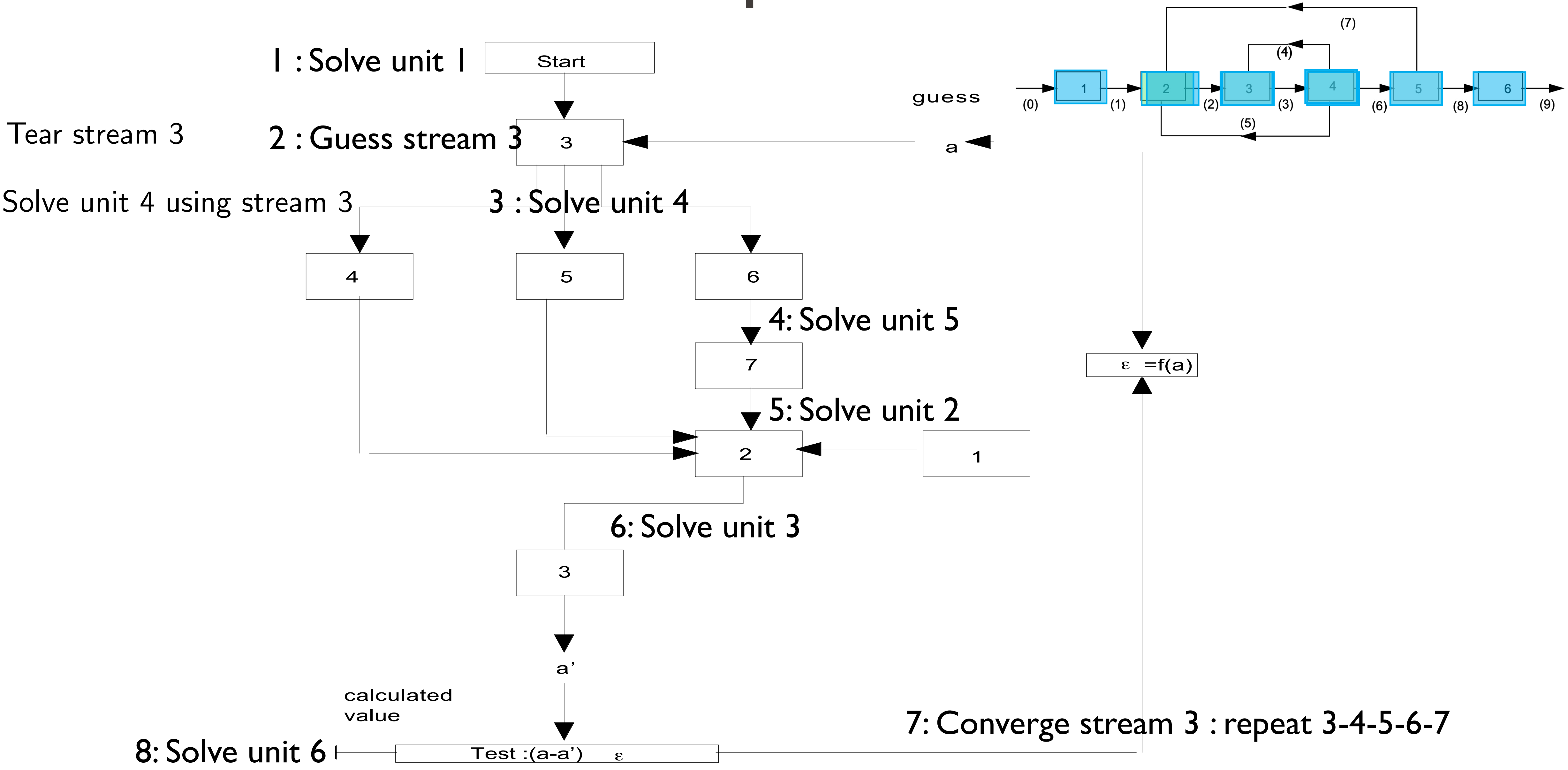
replace streams with only one anterior

STREAMS	ANTERIOR STREAMS
2	3
3	2,3

STREAMS	ANTERIOR STREAMS
3	3

tear stream 3

Define the calculation sequence



Types of tearing streams

- Depending on the process unit calculation model
  - Mass flow (N_c Variables)
    - e.g. if the temperature is specified
  - Temperature, Pressure (2 Variables)
    - e.g. if the flow is specified
  - Total (N_c+2 variables)
    - Typical case
- Tearing Equations : (X is an array)
  - Substitution form

$$X_{tear}^{k+1} = X_{tear}(X_{tear}^{k})$$

  - Equation form

$$X_{tear}^{k+1} - X_{tear}(X_{tear}^{k}) = 0$$

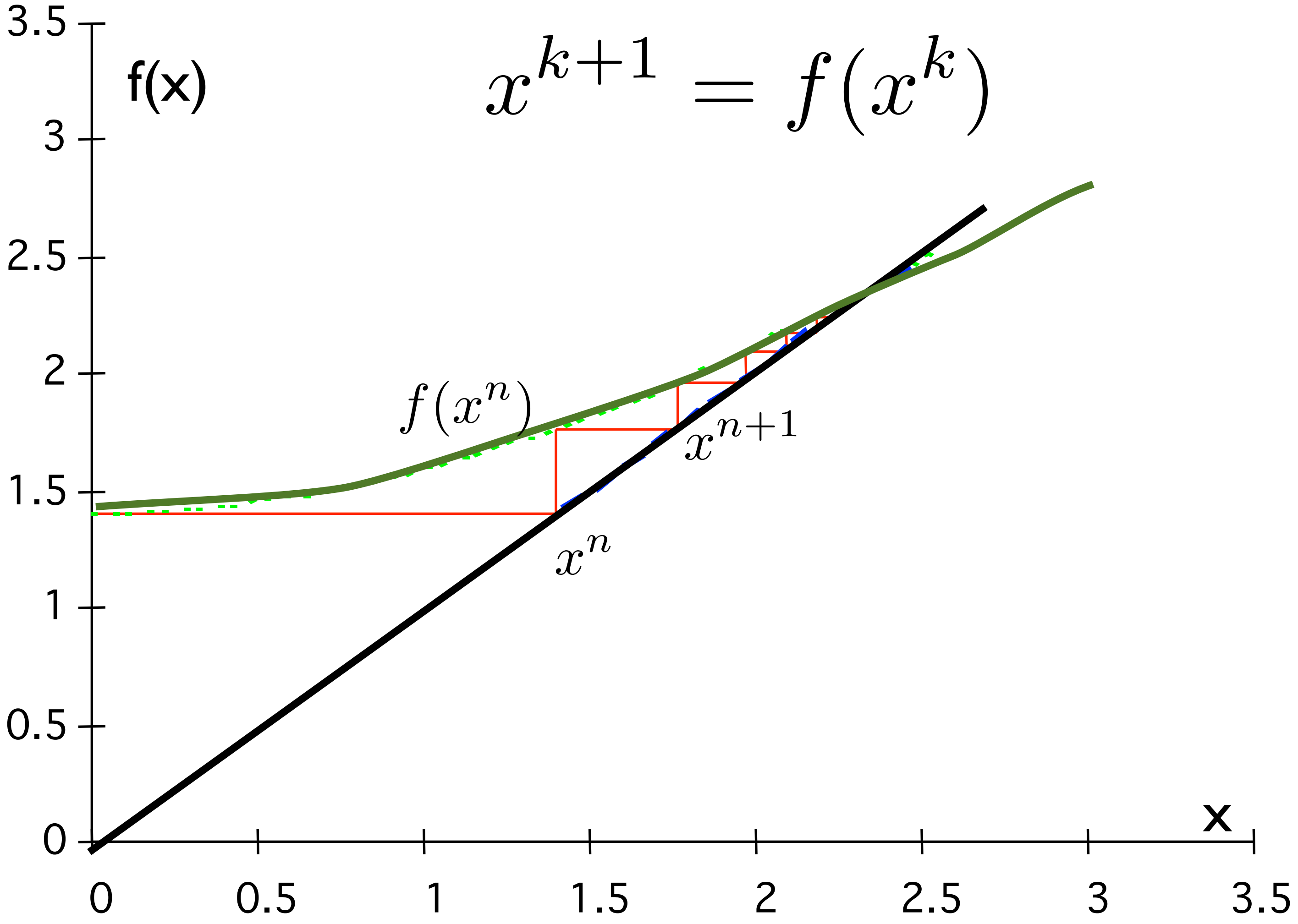
Some remarks

- The units are not always out=f(in)
  - calculation mode or specifications can sometimes calculate in=f(out)
- Understanding the problem helps to improve the method
- A flow sheet can have imbricated loops
  - sequence of iterations

Monotonic convergence

Solving x=f(x) : substitution method

- Simple substitution method



Convergence criteria

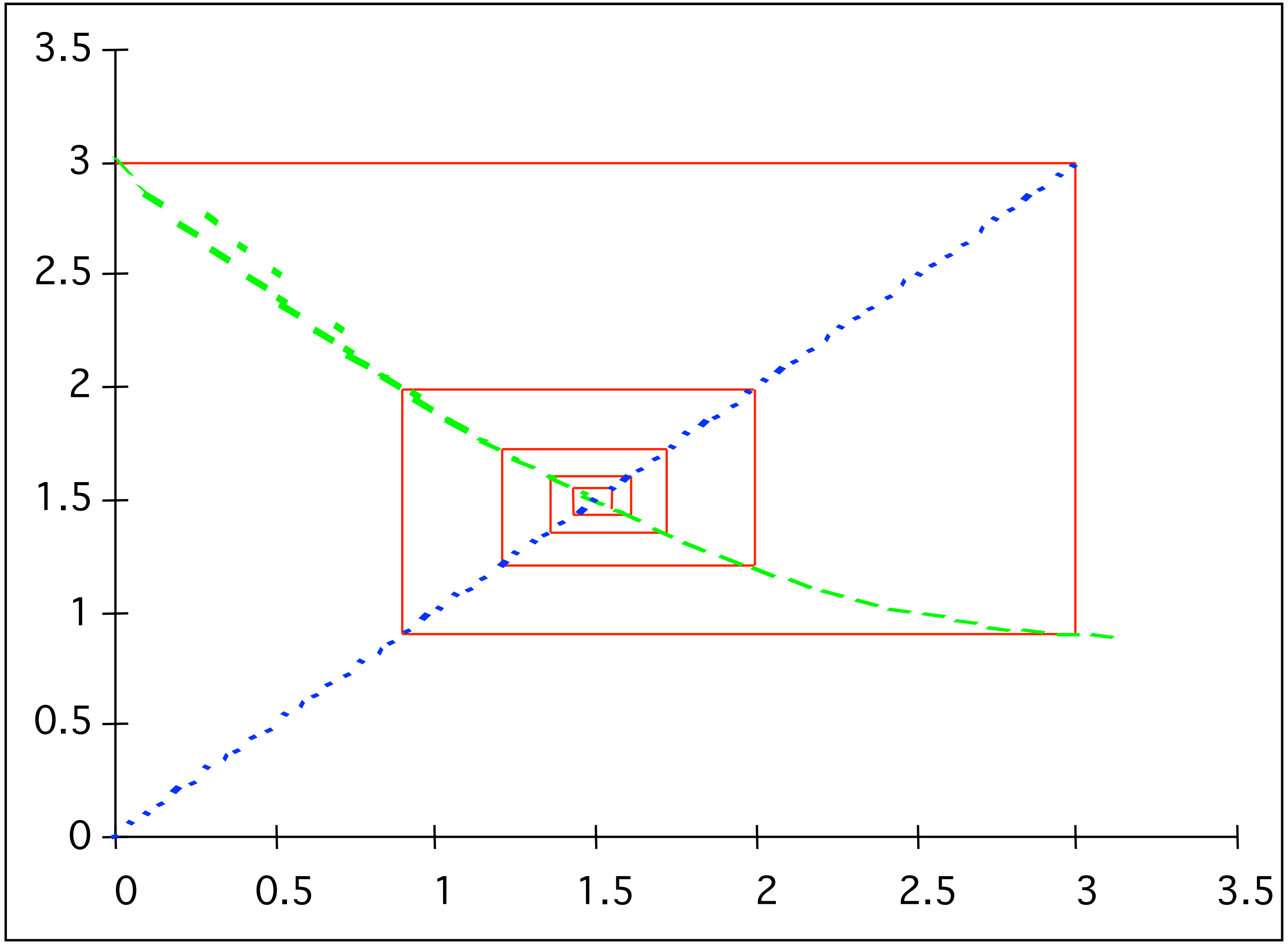
- Choose a small value well chosen value

$$\epsilon = 1.0E - 06$$

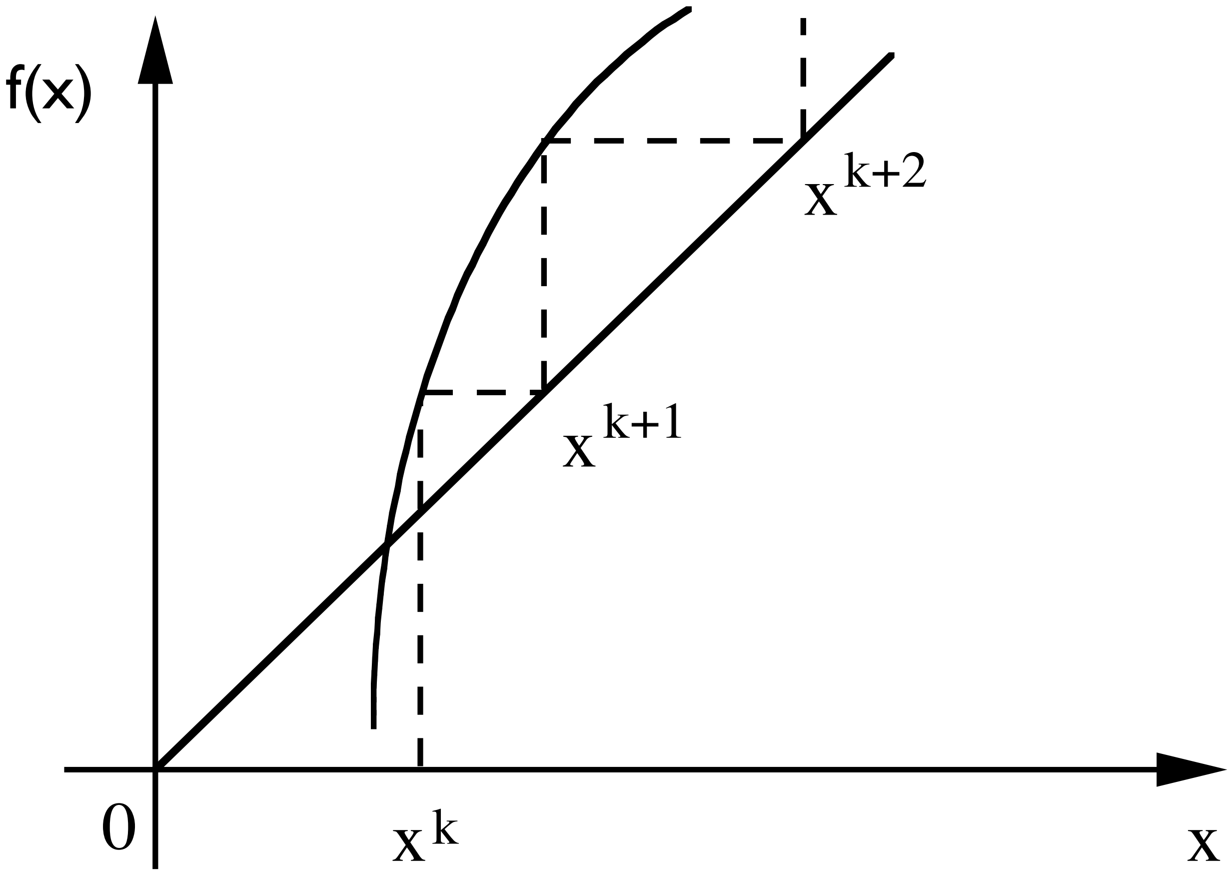
- Test variable variations

$$\mid x^{k+1} - x^{k} \mid \leq \epsilon * (1 + \mid x^{k} \mid)$$

Oscillating Convergence

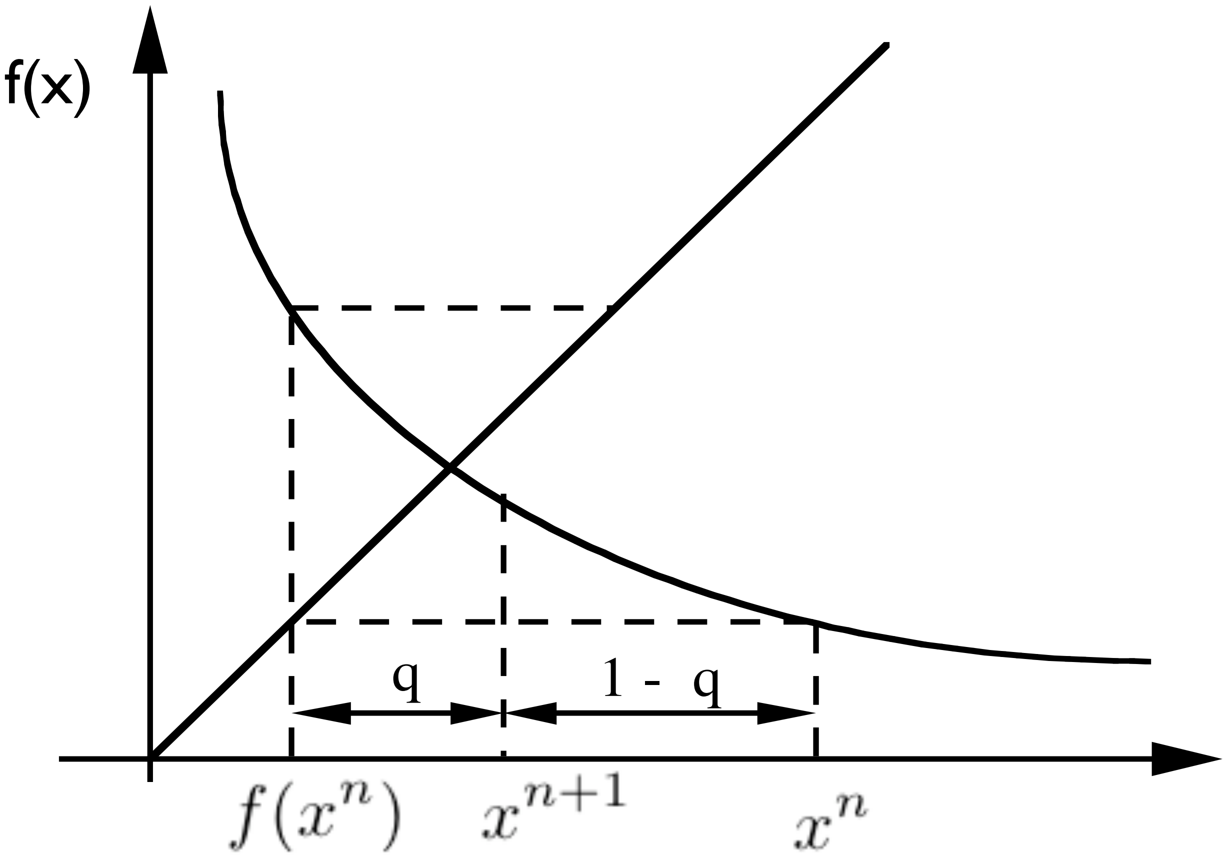


x=f(x) : Substitution may diverge



Solution : Apply relaxation

$$x^{n+1} = x^n \cdot q + f(x^n) \cdot (1 - q)$$



Wegstein method

- System to be solved : intersection of two curves

$$y - y^k = \frac{y^{k+1} - y^k}{x^{k+1} - \tilde{x}^k} . (x - \tilde{x}^k) = \psi . (x - \tilde{x}^k)$$

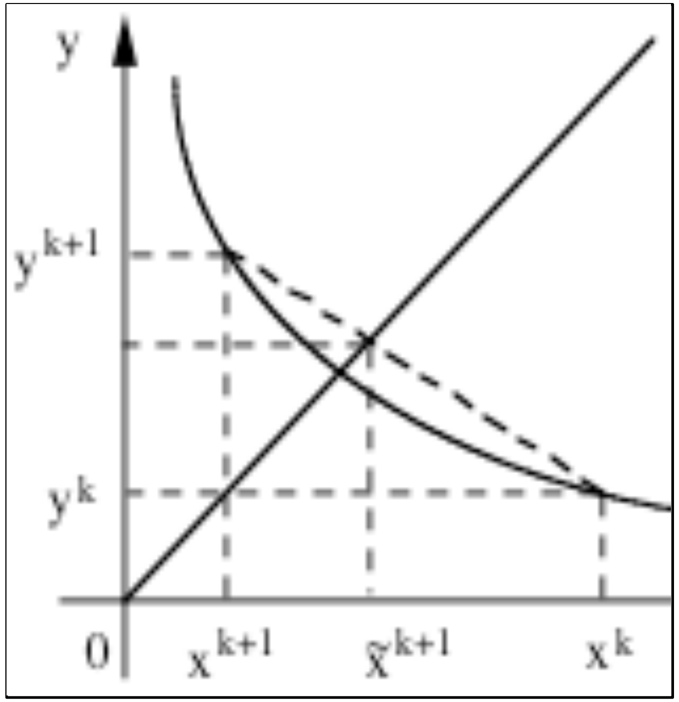
$$y = x$$

- ψ = Chord slope
- Therefore

$$\tilde{x}^{k+1} = \frac{1}{1-\psi} . (x^{k+1} - \psi . \tilde{x}^k)$$

-
- relaxation of substitution

$$\tilde{x}^{k+1} = q . \tilde{x}^k + (1-q) . x^{k+1}$$



$$q = \frac{-\psi}{1-\psi} \quad et \quad 1 - q = \frac{1}{1-\psi}$$

- Increased speed of convergence
- More robust

Solving f(x)= 0 : Newton-Raphson method

find x such that F(x) = 0 => here $F(X) = X - \psi(X) = 0$

TAYLOR development to approximate any function :

$$f(x) = f(x^0) + (x - x^0) \cdot f'(x^0) + \frac{(x - x^0)^2}{2!} f''(x^0) + \cdots$$

1rst order limitation (approximate with a straight line) near x^0

$$f(x^0) + (x^* - x^0) \, . f'(x^0) \cong f(x^*) = 0$$

$$x^* = x^0 - \frac{f(x^0)}{f'(x^0)}$$

Conclusions : Solving non linear equations

- Iterative procedure
  - Initial guess is important
  - Does not always converge !
  - Bounds on variables + safe guards
- Initialisation point with good values
- Direction + Step length
  - Derivatives
    - Numerical calculations => step length of perturbation
    - Numerical calculation cost (time)
  - Matrix inversion
    - algorithms try to reuse direction for several steps
- Test of convergence
  - on both equations and variables
  - one criteria for several equations => scaling