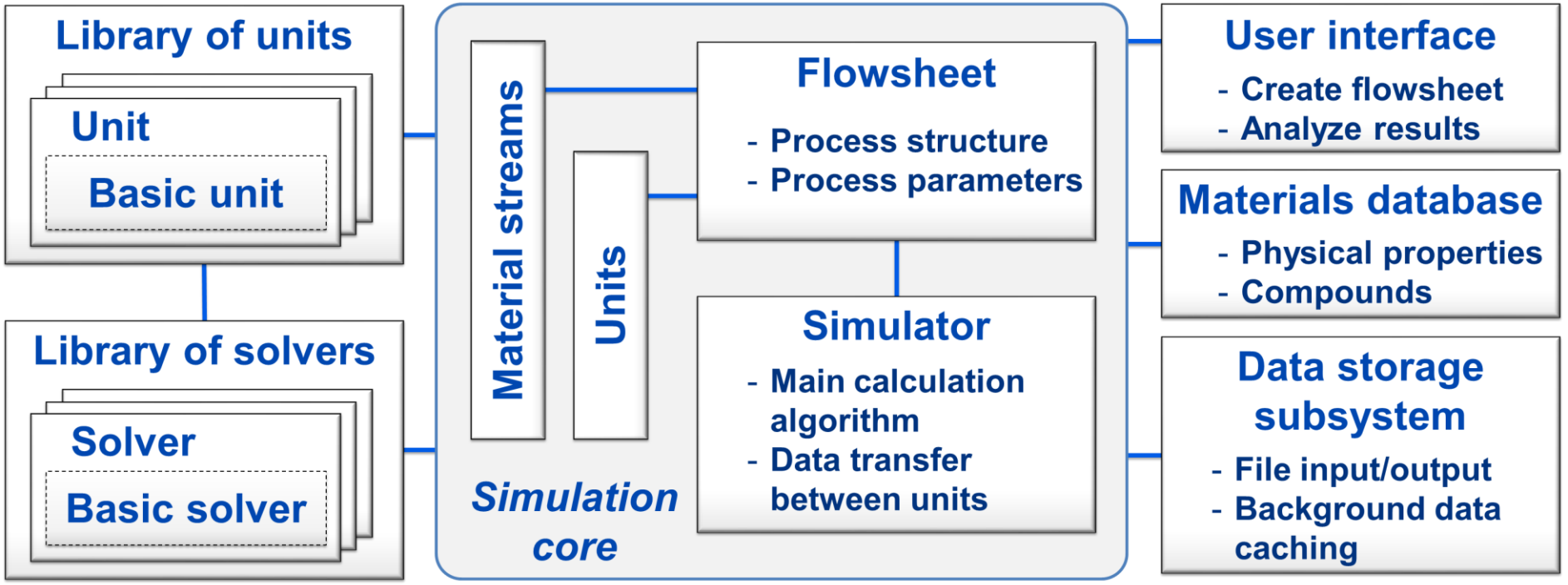
Dynamic flowsheet simulation system Dyssol

Architecture and Algorithms

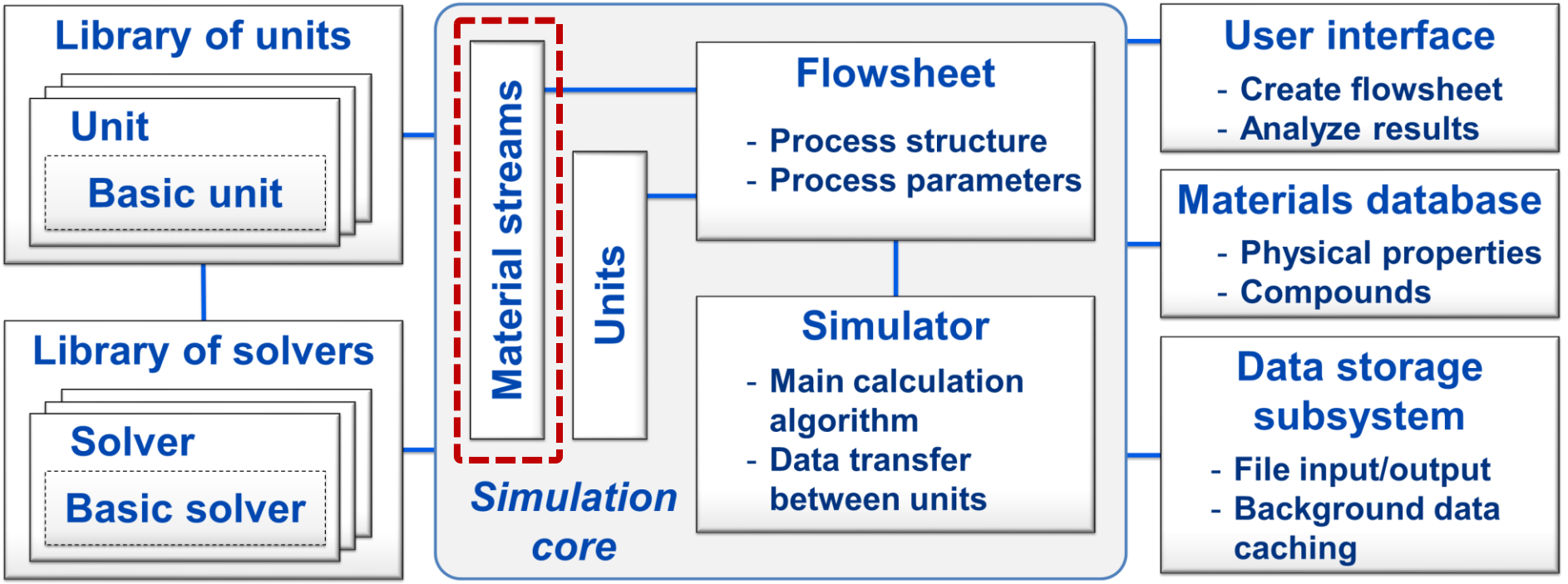
Architecture of the simulation system

General architecture of Dyssol

- The name of the system is Dyssol: **DY**namic **S**imulation of **SOL**ids Processes
- Modular architecture to increase the flexibility of the framework
- Models of units and solvers are not integrated directly into the simulation system
- Standardized interfaces and templates for the implementation of units and solvers
- Simplified user interface to setup simulations and to analyse results
- High modularity and extensibility of the simulation system

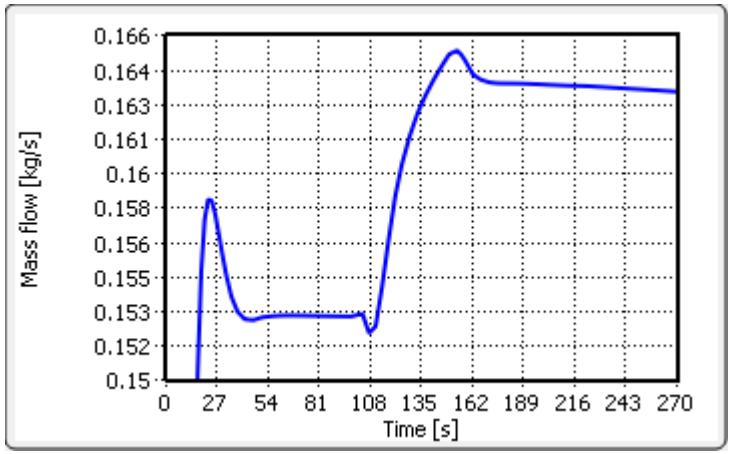


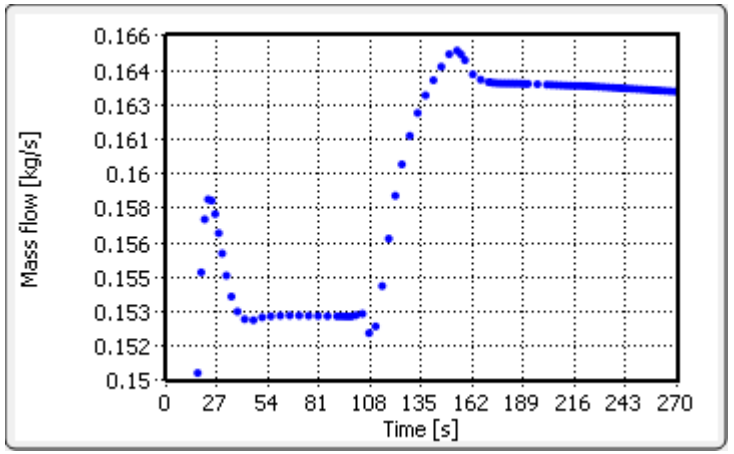
Material streams



Data discretization

- A continuous process is discretized using time points
- Each time point is a snapshot of the process state





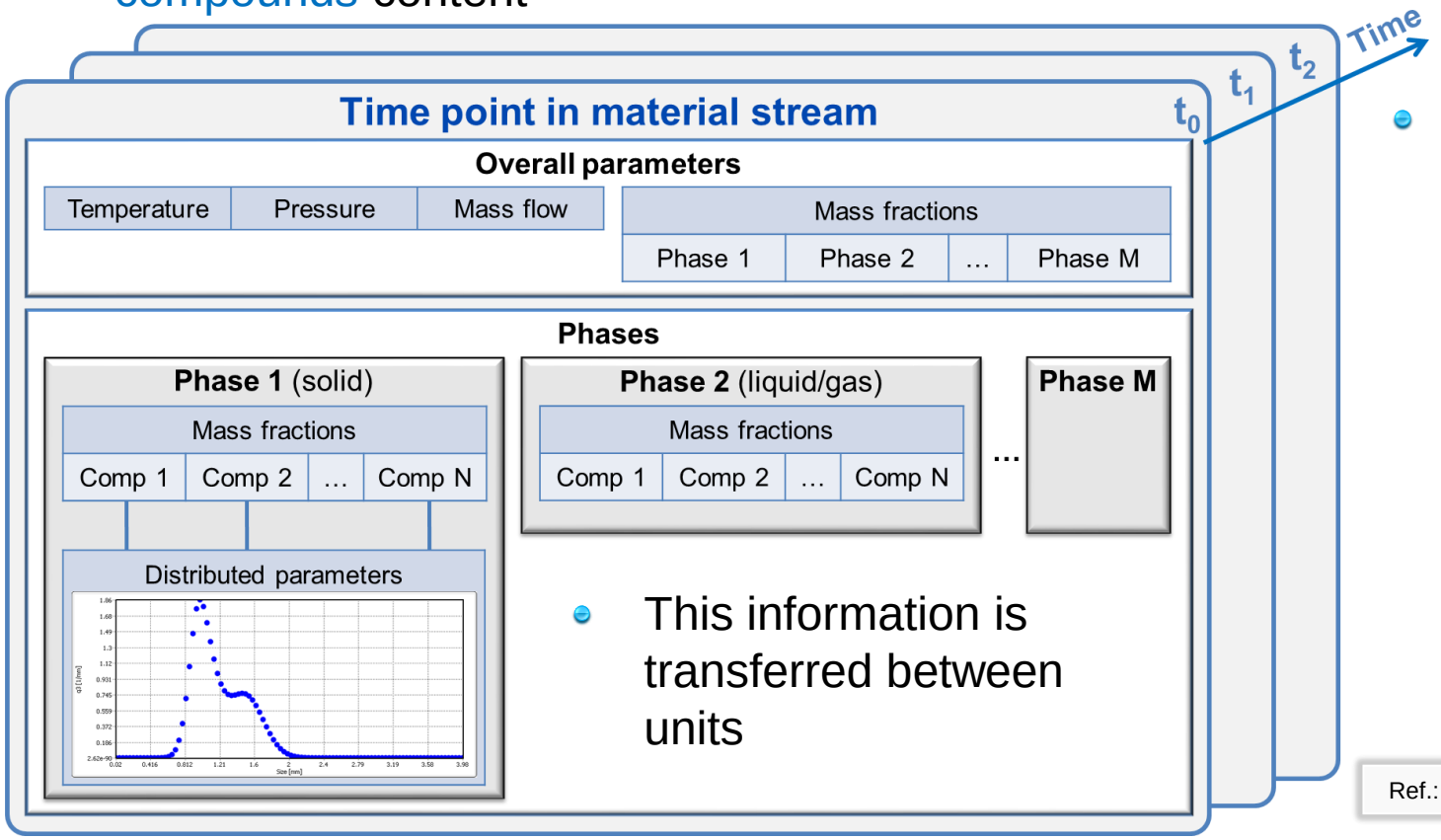
- All units are connected by material streams
- Material streams are described by a set of time points

Material stream

General structure

- Overall properties define parameters for all selected phases
- Solid, liquid and gas phases are available
- Each phase is distributed along compounds content

- Solid phase can be distributed along several multidimensional properties
- Each stream on the flowsheet has the same set of compounds, phases and multidimensional properties

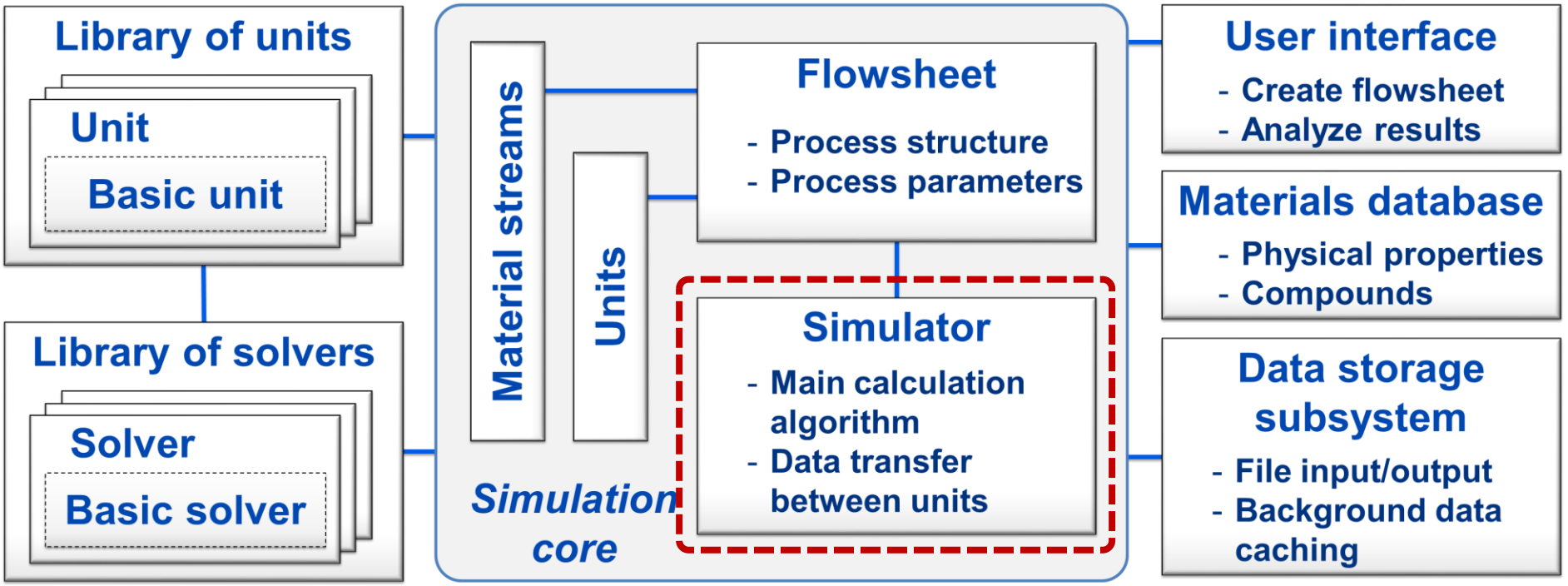


- This information is transferred between units

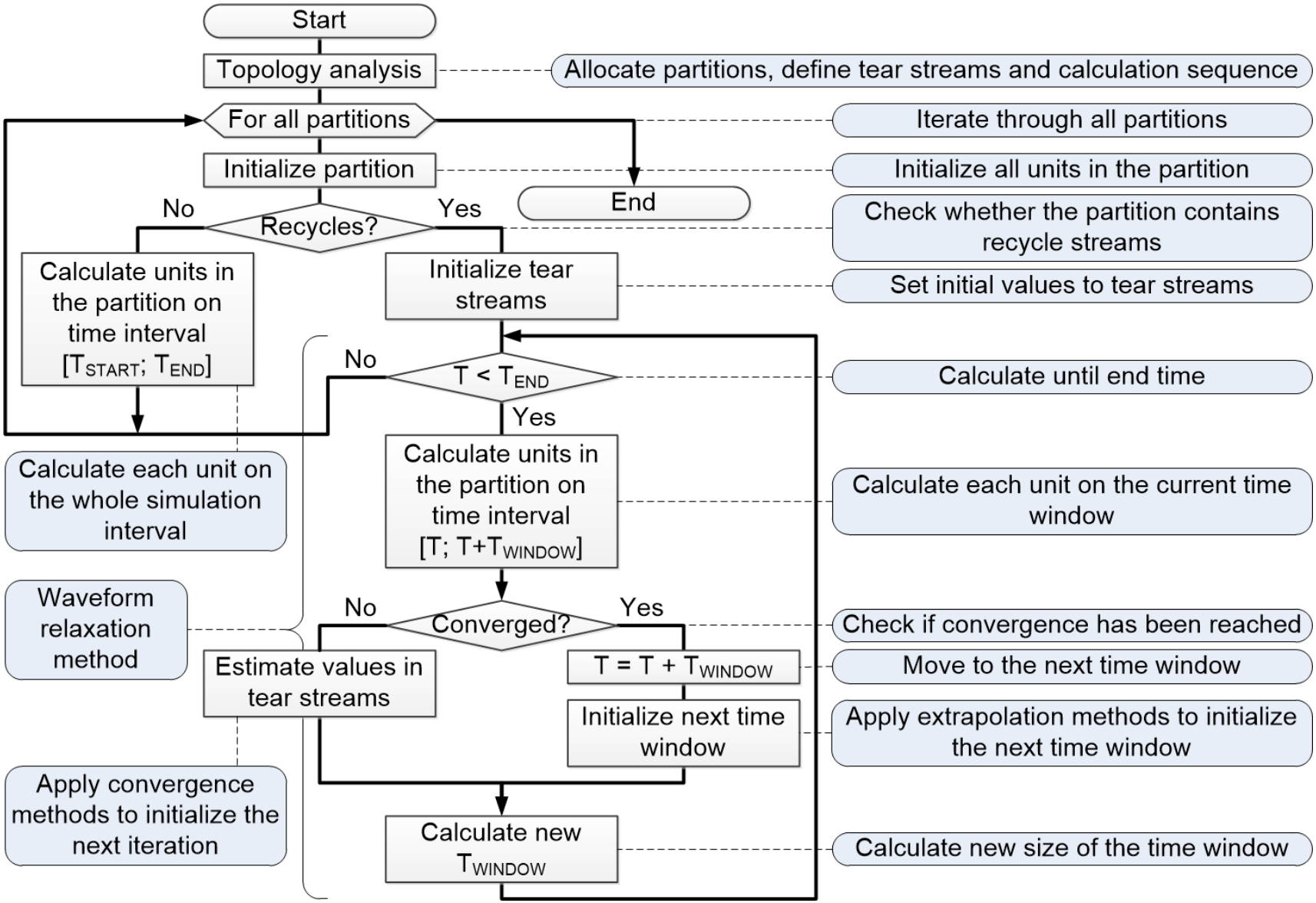
- All variables in material streams are time-dependent

Ref.: Help\Program interfaces\Stream.pdf

Simulator


Library of units
Unit
Basic unit
Library of solvers
Solver
Basic solver
Material streams
Units
Simulation core
Flowsheet
- Process structure
- Process parameters
Simulator
- Main calculation algorithm
- Data transfer between units
User interface
- Create flowsheet
- Analyze results
Materials database
- Physical properties
- Compounds
Data storage subsystem
- File input/output
- Background data caching

General simulation algorithm



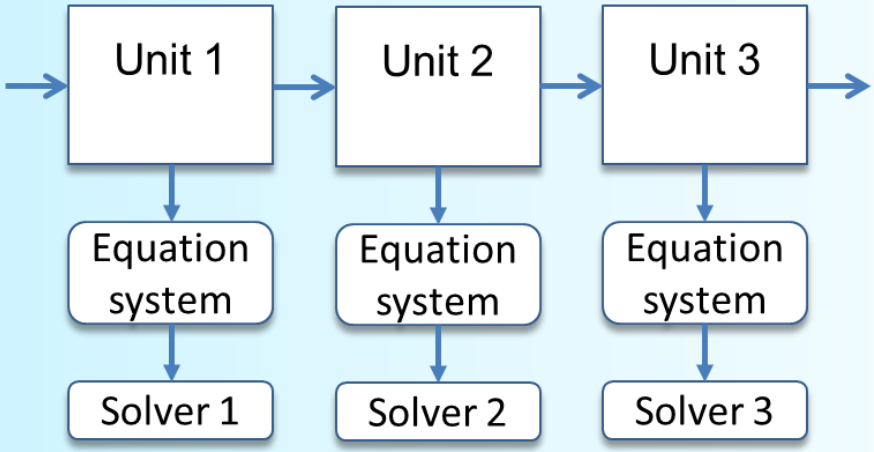
Main methods and approaches

- Sequential-modular approach: each model is solved separately
- Dividing of a flowsheet into partitions and tearing of recycle streams
- Waveform relaxation method (WRM) for dynamic calculation of recycle streams: dividing simulation time into shorter intervals
- Data extrapolation to initialize each time window
- Convergence methods to initialize each iteration of WRM

V. Skorych et al. Novel system for dynamic flowsheet simulation of solids processes (2017)

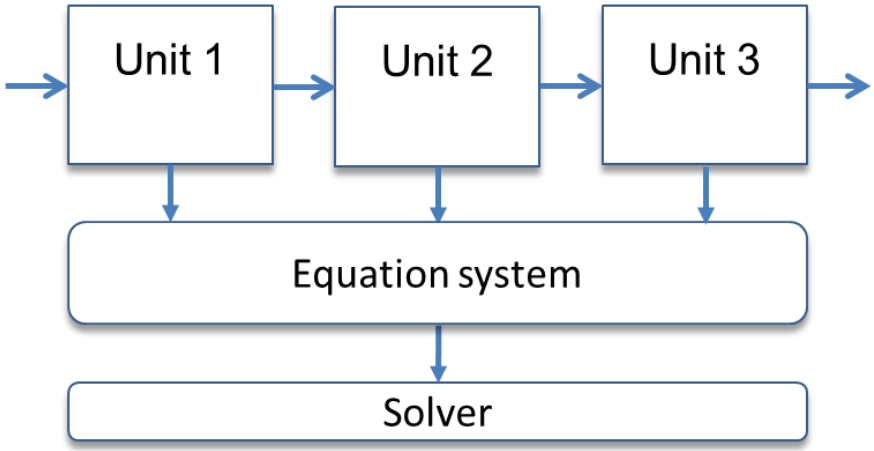
Simulator

Modular vs. simultaneous approach



Sequential-modular

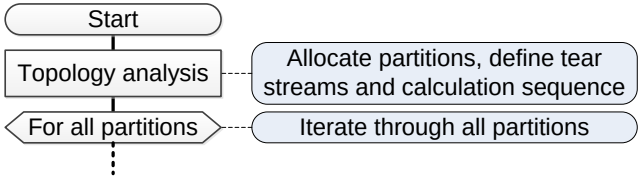
- Conceptual simplicity
- Correspondence to the physical structure of processes
- Higher flexibility
- Possibility to use different numerical methods to calculate models
- ✖ Difficulties in processing flowsheets with recycle streams

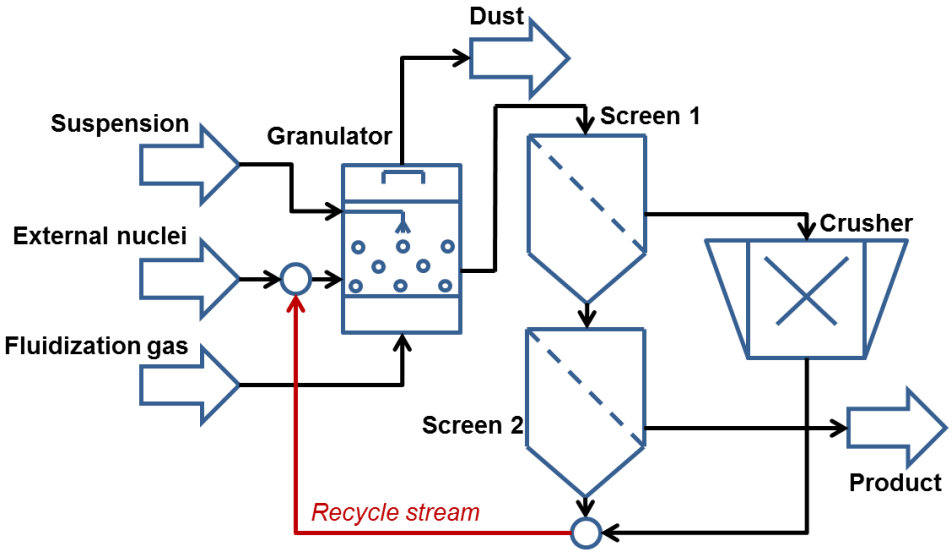


Equation-oriented

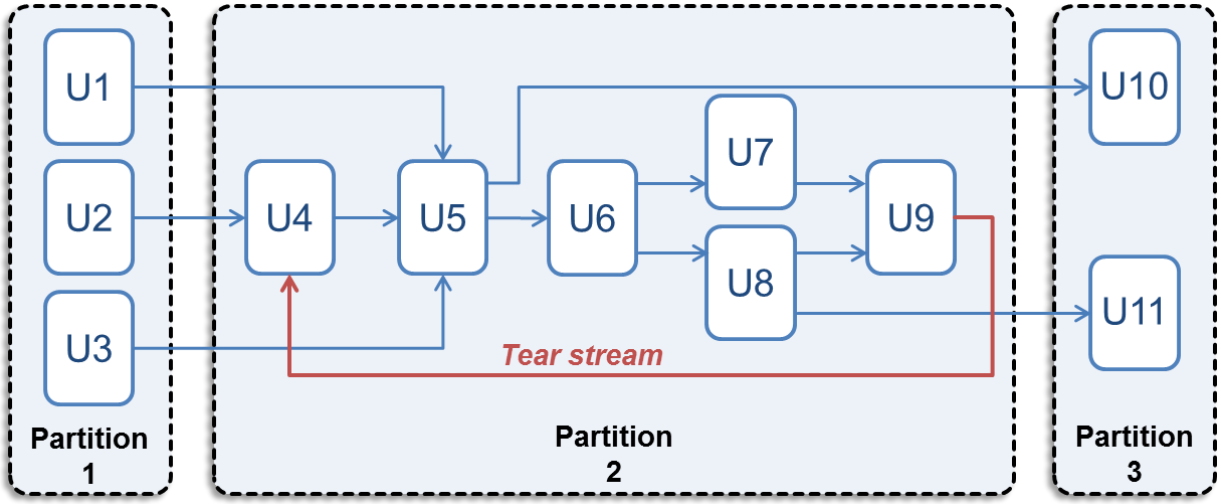
- Equations are homogenized and calculated by a single solver
- Better convergence rate, especially on complex flowsheets with recycles
- ✖ Difficulties with simultaneous stimulation of fast and slow changed components
- ✖ Usage with heterogeneous models is complicated

W. Marquardt. Dynamic process simulation - recent progress and future challenges (1991)

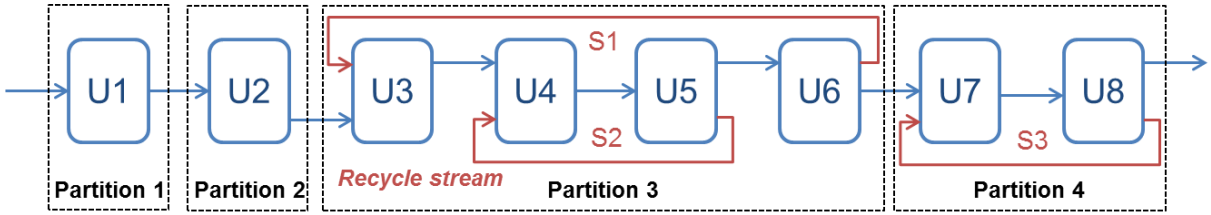




- Convert the process structure into a suitable sequential form
- Initialize recycle streams with some values
- Combine units in recycle loops into partitions
- Solve an entire partition (a recycle system) iteratively over the whole simulation time and separately from other partitions



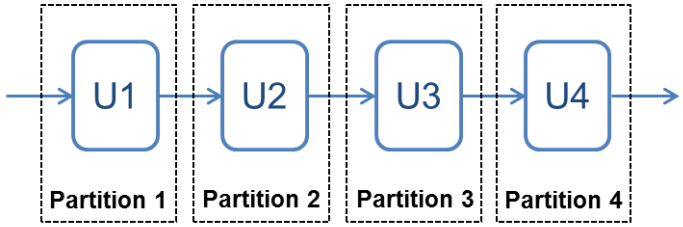
A single scheme on a flowsheet

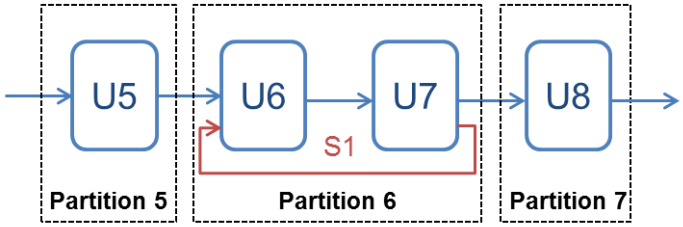


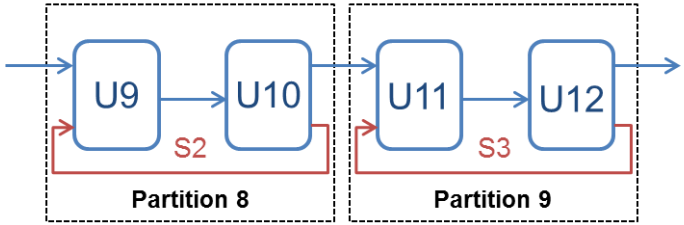
Calculation sequence

- Partition 1: U1, U2
- Partition 2: U3, U4, U5, U6, S1, S2
- Partition 3: U7, U8, S3

Several schemes on a flowsheet



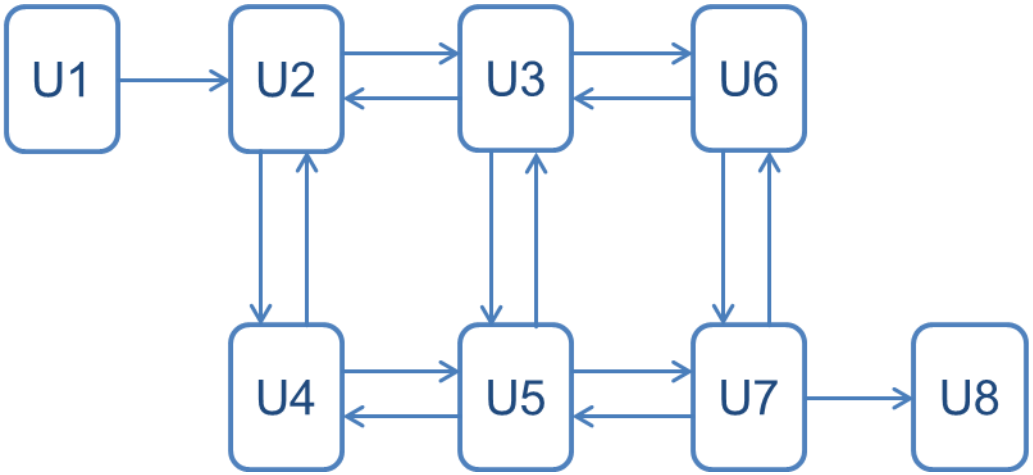


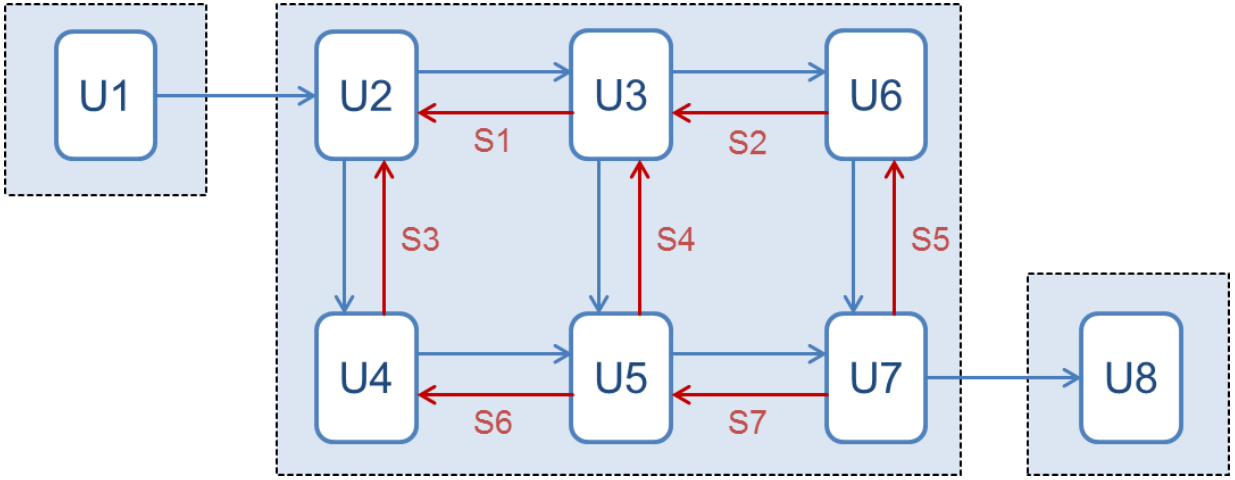


Calculation sequence:

- Partition 1: U1, U2, U3, U4
- Partition 2: U5
- Partition 3: U6, U7, S1
- Partition 4: U8
- Partition 5: U9, U10, S2
- Partition 6: U11, U12, S3

Bi-directionally connected units





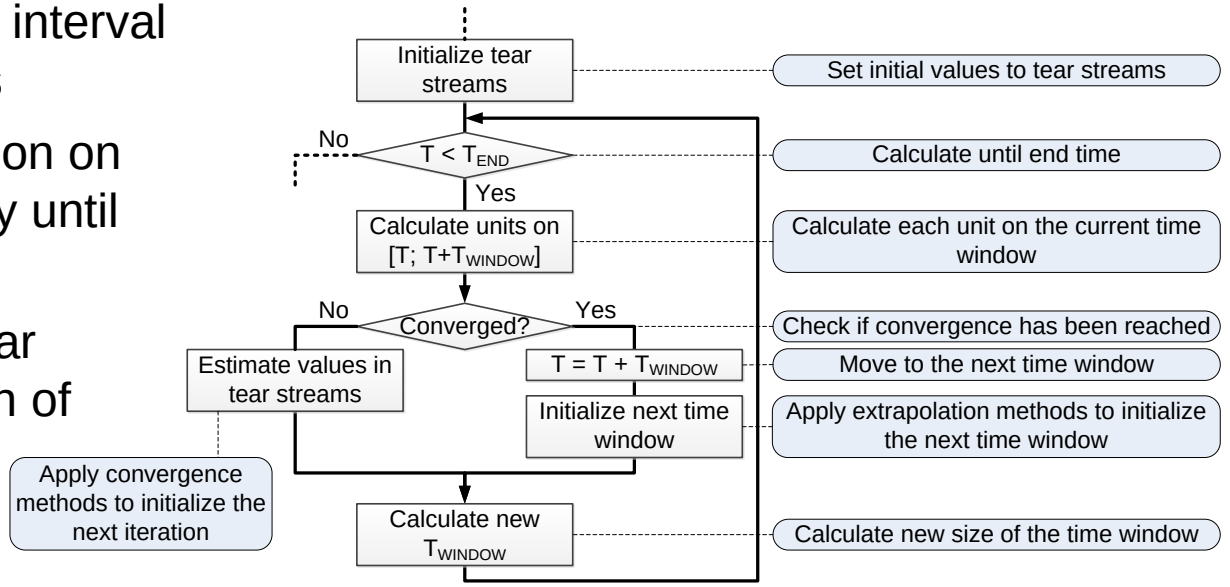
- Bi-directional connection can be represented in Dyssol as two independent material streams
- One of the streams will eventually be selected as a tear stream

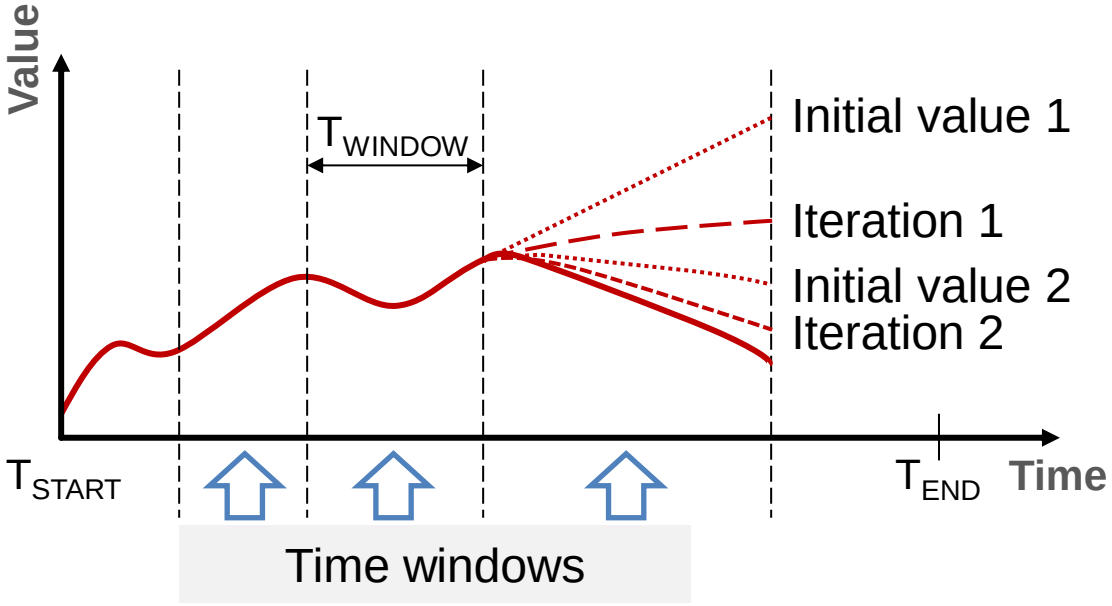
Calculation sequence

- Partition 1: U1
- Partition 2: U2, U3, U4, U5, U6, U7, S1, S2, S3, S4, S5, S6, S7
- Partition 3: U8

Waveform relaxation method

- Split the whole simulation interval into smaller time windows
- Calculate the entire partition on the time window iteratively until convergence
- Initialize parameters of tear streams before calculation of each time window



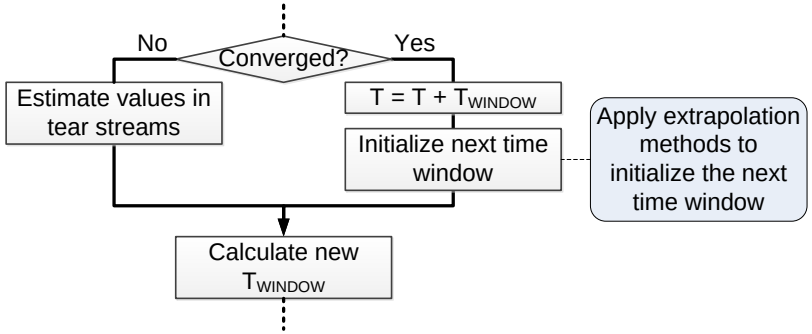


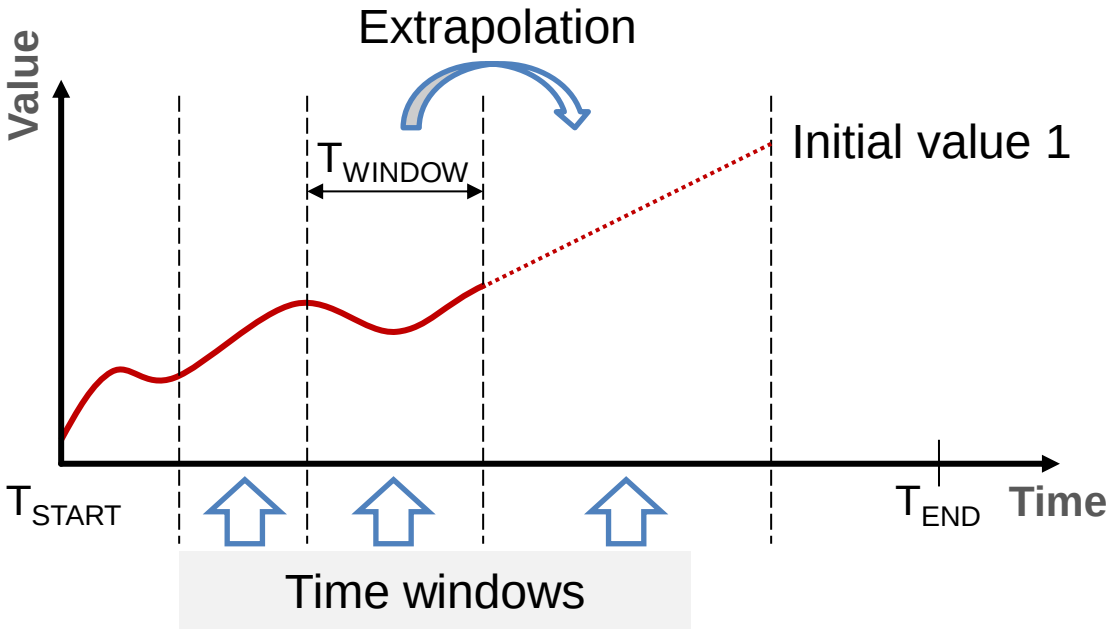
- Size of the time window varies depending on the convergence rate
- The convergence criterion is the difference between values on successive iterations

E. Lelarasmee. The waveform relaxation method for time domain analysis of large scale integrated circuits (1982)

Extrapolation methods

- Initialize all parameters of tear streams within the time window before its calculation
- Use extrapolated values from the previous time intervals as initial data
- Prediction accuracy affects the number of iterations required to achieve convergence





- Implemented methods:
  - Nearest- neighbour
  - Linear
  - Cubic spline

Convergence methods

- Use convergence methods to initialize parameters of tear streams before each iteration
- Perform calculations iteratively
- Stop iterative calculations when the convergence is reached

Implemented methods:

- Direct substitution:

$$x_{k+1} = (1-\lambda)F(x_{k-1}) + \lambda F(x_k)$$

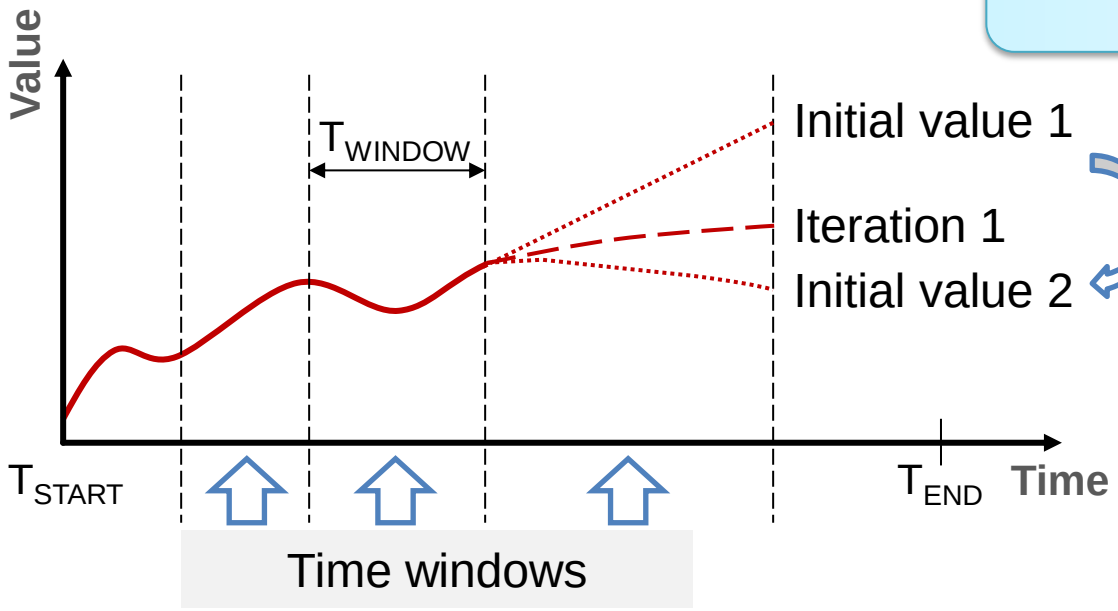
- Wegstein's method:

$$x_{k+1} = qx_k + (1-q)F(x_k)$$

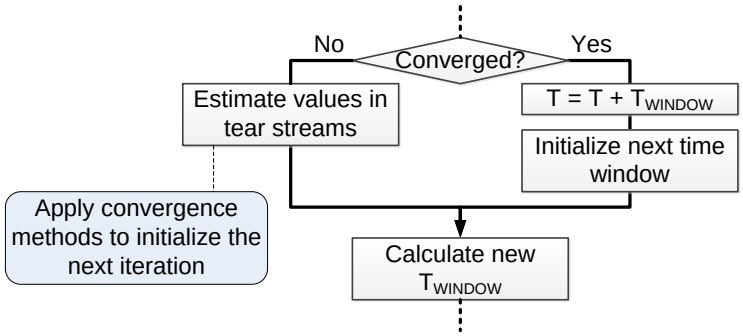
- Steffensen's method:

$$x_{k+3} = x_k - \frac{(x_{k+1} - x_k)^2}{x_{k+2} - 2x_{k+1} + x_k}$$

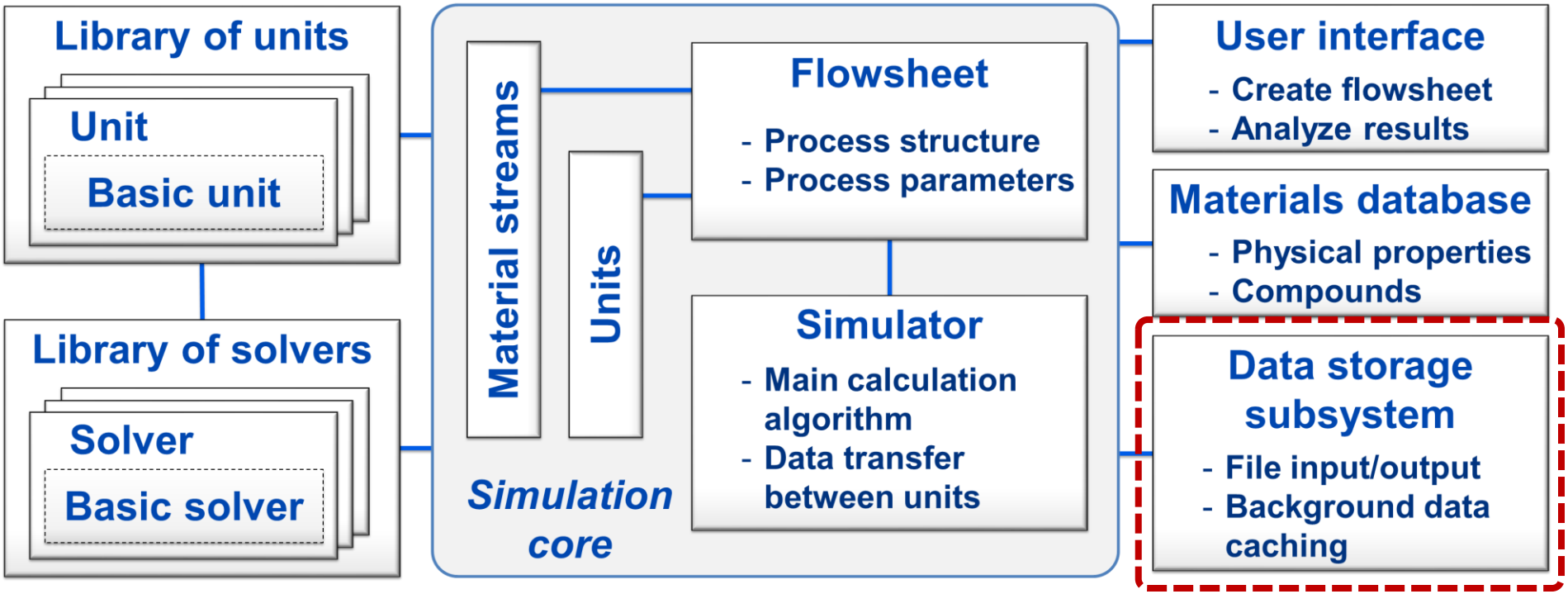
Ref.: Help\Convergence.pdf



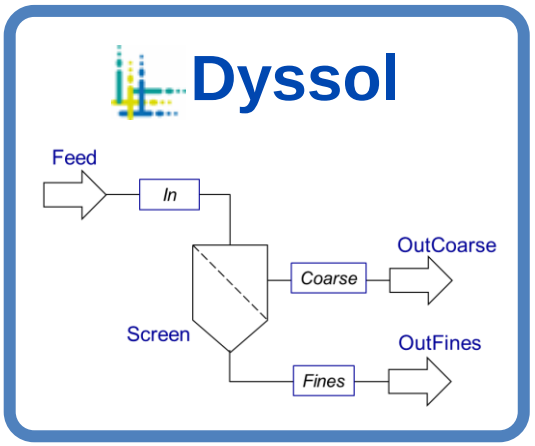
Convergence methods



Data storage subsystem



HDF5 data format



Save data

Load data

HDF5 library

***.dflw file** on HDD

- Process structure
- Unit parameters
- Process parameters
- Simulation results

Set parameters

Get results

External program

via MATLAB, Mathematica, C++, Python…

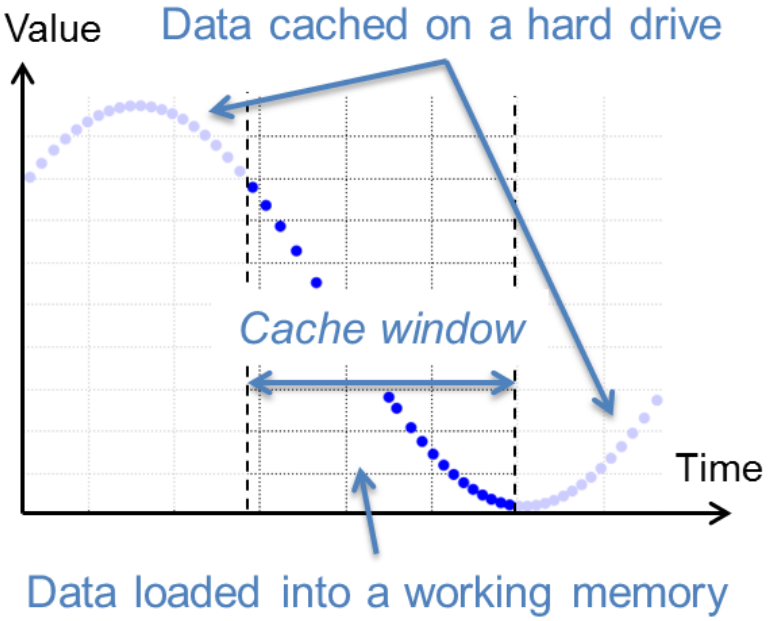
HDF5 library

- Dyssol uses HDF5 to store all data
- HDF5 is a hierarchical data format, library and data model for storing and organizing of large amounts of numerical data
- External programs can access saved data via HDF5 interfaces

https://www.hdfgroup.org/solutions/hdf5/

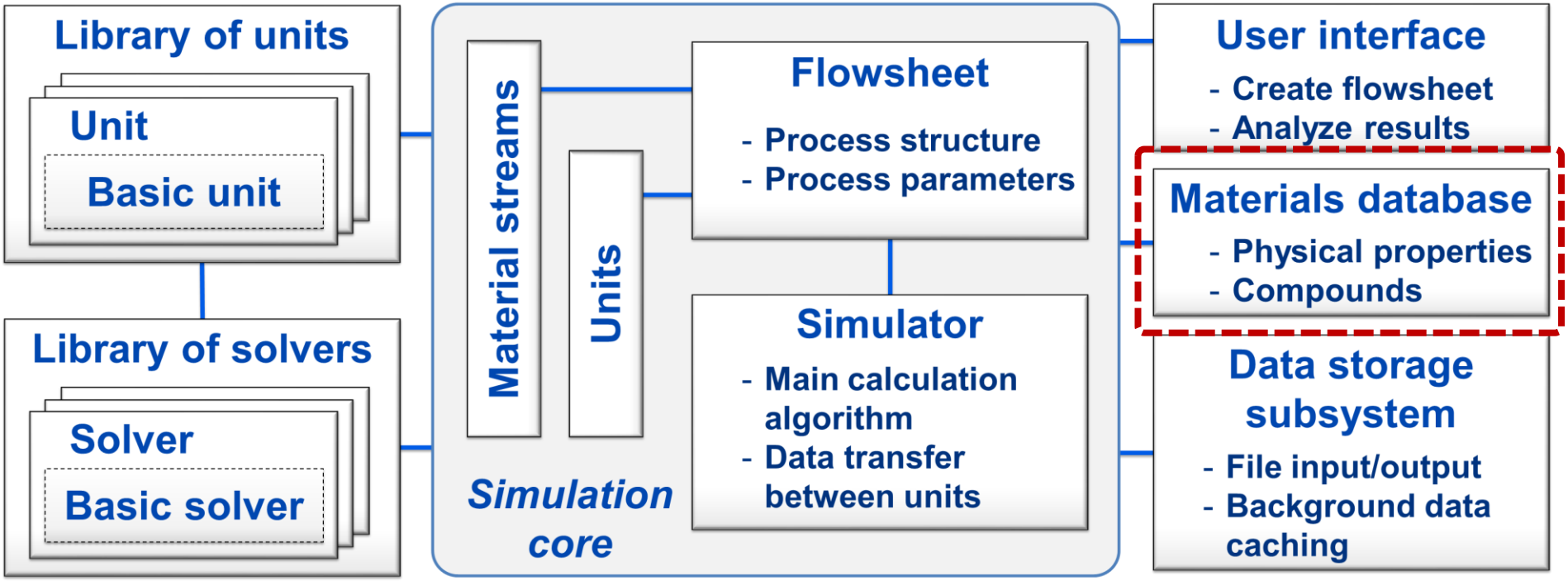
Data caching

- Only several time points of material stream are kept in memory at every moment
- The rest are temporary on disk
- Reduces memory consumption
- Slightly increases computation time (at default settings)



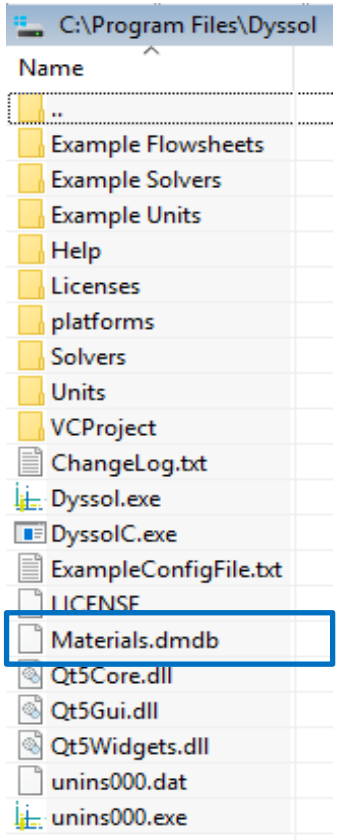
V. Skorych et al. Novel system for dynamic flowsheet simulation of solids processes (2017)

Materials database



DMDB file

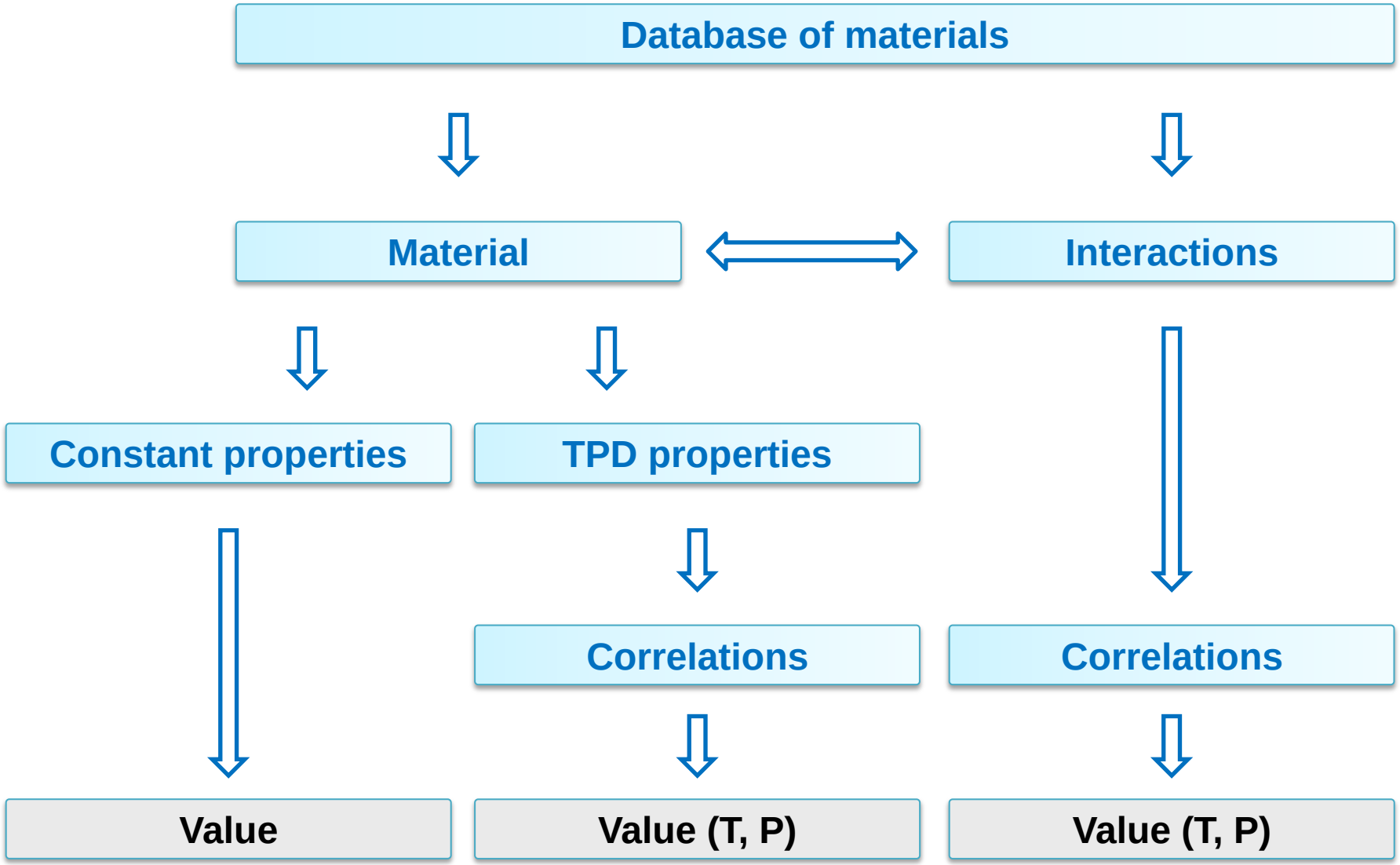
Materials.dmdb – stores global list of **materials** and their **parameters**



- Air
- Coal
- Sand
- Urea
- Water
- …

Can be freely extended with new materials

Materials database structure



Constant parameters

Material

⇩

Constant properties

- Critical pressure
- Critical temperature
- Heat of fusion
- Heat of vaporization
- Molar mass
- Normal boiling point
- Normal freezing point
- Reactivity type
- Formation enthalpy
- State of aggregation →
  - 0 – solid
  - 1 – liquid
  - 2 – gas
  - 3 – unknown
- *User defined properties*

- Not all properties have influence on internal calculations
- Molar mass is mandatory, since many parameters can be calculated on mole-basis
- State of aggregation can be used to define several aggregation states of the same material
- Can be accessed through units and material streams

Dependent parameters

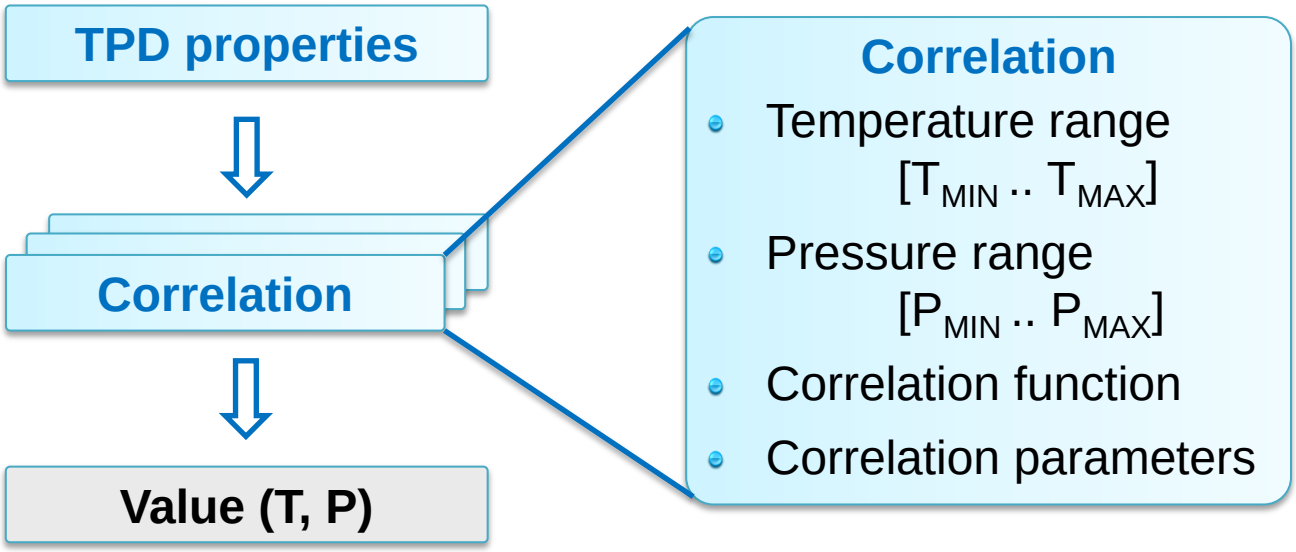
Material

- Not all properties have influence on internal calculations
- Density is mandatory, since it participates in PSD transformations and T-equilibrium
- Enthalpy is mandatory, since it participates in T-equilibrium
- Can be accessed through units and material streams

Temperature- and/or Pressure-dependent properties

- Enthalpy
- Thermal conductivity
- Vapor pressure
- Viscosity
- Density
- Permittivity
- *User defined properties*

Correlation search



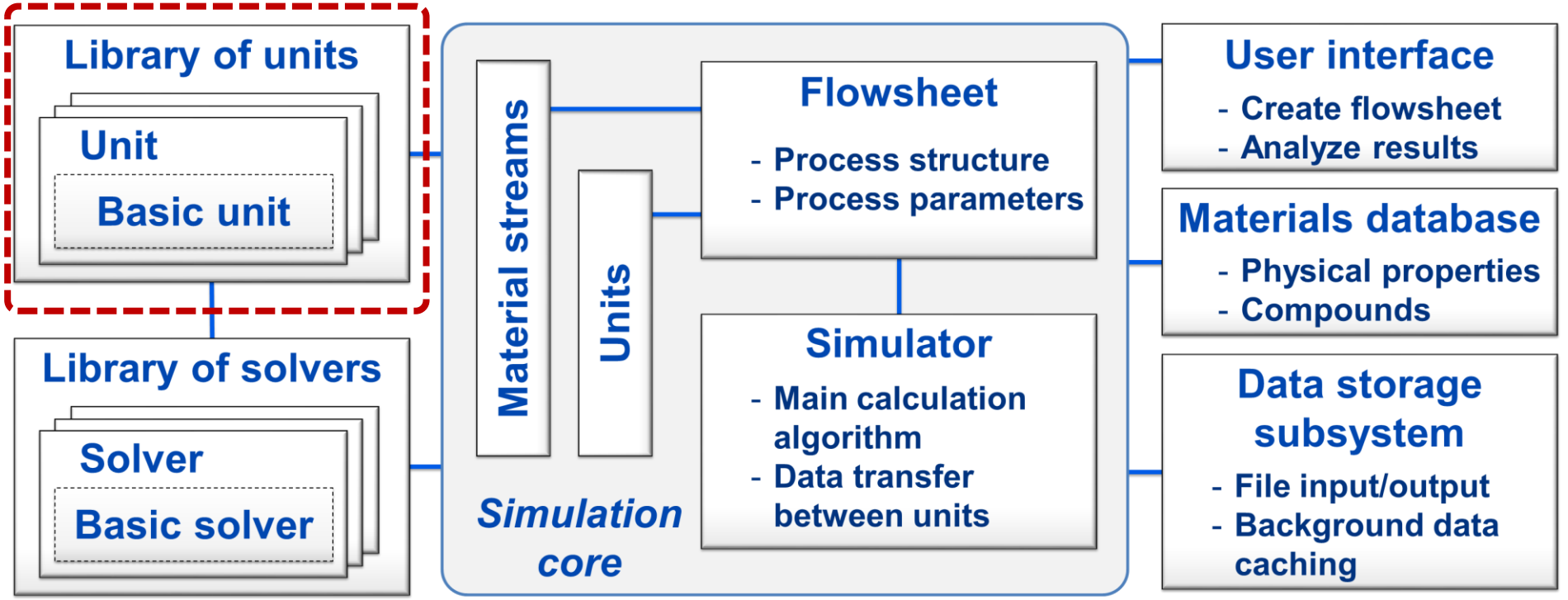
Value calculation algorithm:

1. Find the first correlation with **T & P** within
   - ✓ Calculate value according to the function
2. Find the first correlation with only **T** within
   - ✓ Calculate value according to the function
3. Find the nearest correlation, taking only **T** into account
   - ✓ Perform nearest neighbour extrapolation

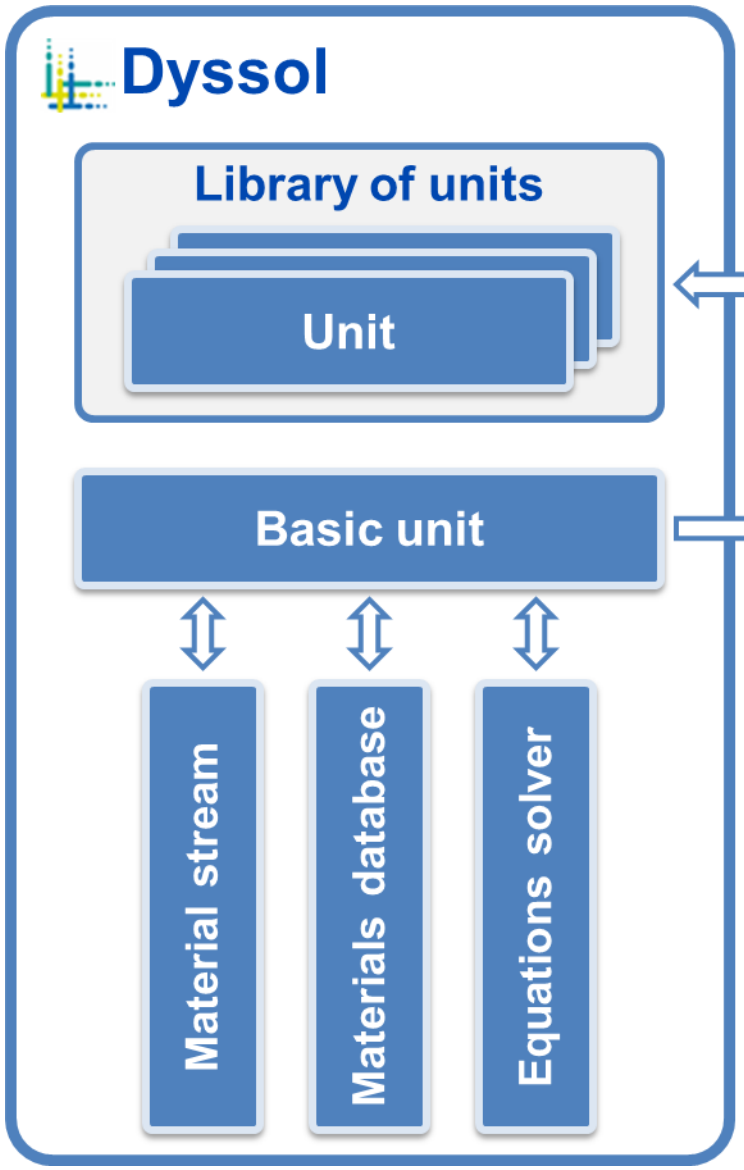
Correlation functions

- Constant $y = a$
- Linear $y = aT + bP + c$
- Exponential $y = ab^{c+dT+\frac{eT+f}{gT+h}} + i$
- Power function $y = aT^b$
- Polynomial $y = a + bT + cT^2 + dT^3 + eT^4 + fT^5 + gT^6 + hT^7$
- Shomate heat capacity $y = a + bT + cT^2 + dT^3 + \frac{e}{T^2}$
- Shomate standard enthalpy $y = aT + b\frac{T^2}{2} + c\frac{T^3}{3} + d\frac{T^4}{4} - \frac{e}{T} + f - g$
- Shomate standard entropy $y = a \cdot \ln(T) + bT + c\frac{T^2}{2} + d\frac{T^3}{3} - \frac{e}{2T^2} + f$
- List of T-values $y = \{T1: val1, T2: val2, T3: val3, \dots\}$
- List of P-values $y = \{P1: val1, P2: val2, P3: val3, \dots\}$

Units


Library of units
Unit
Basic unit
Library of solvers
Solver
Basic solver
Material streams
Units
Simulation core
Flowsheet
- Process structure
- Process parameters
Simulator
- Main calculation algorithm
- Data transfer between units
User interface
- Create flowsheet
- Analyze results
Materials database
- Physical properties
- Compounds
Data storage subsystem
- File input/output
- Background data caching

Interfaces of units



- Streams processing: inlets, outlet, mixer, splitter
- Steady-state: screen, mill, …
- Dynamic: granulator, agglomerator, bunker, …

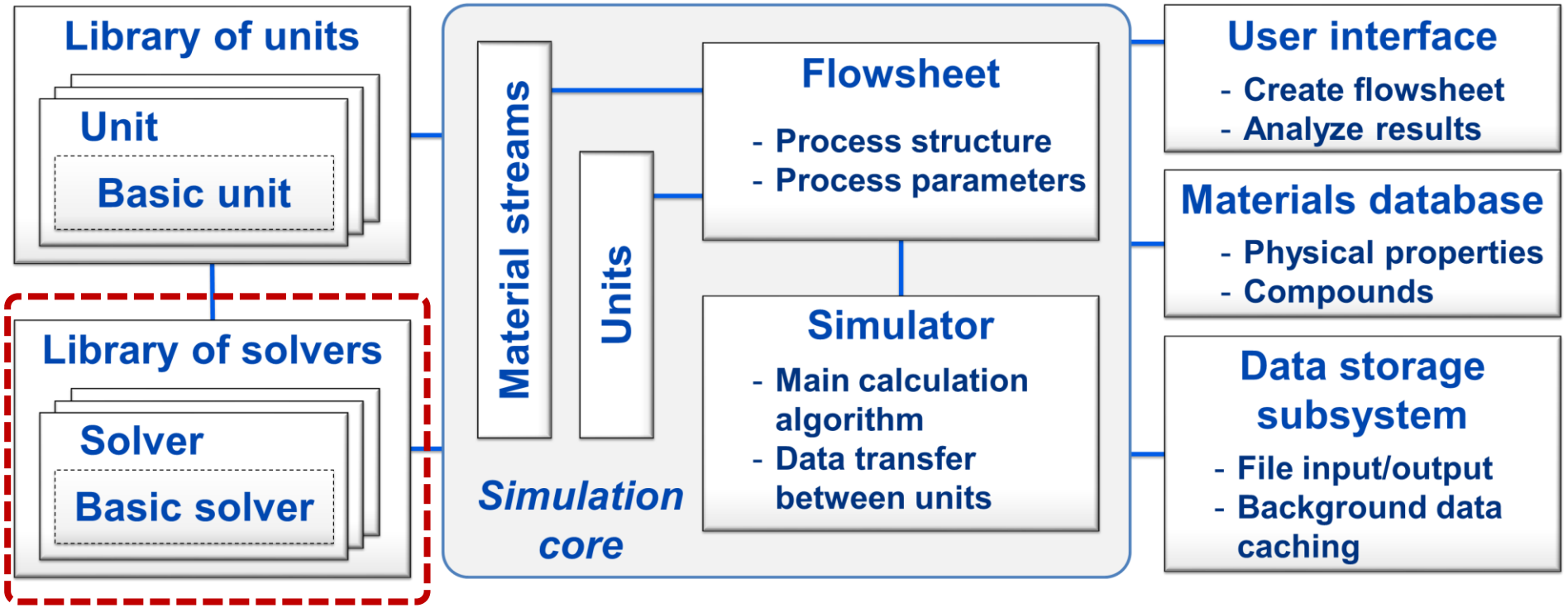
- Template for creating of custom units: can be implemented by users and added to the library

- Basic unit provides interfaces to the simulation system
- Each unit contains program interfaces to:
  - Material streams
  - Equation solvers
  - Materials database

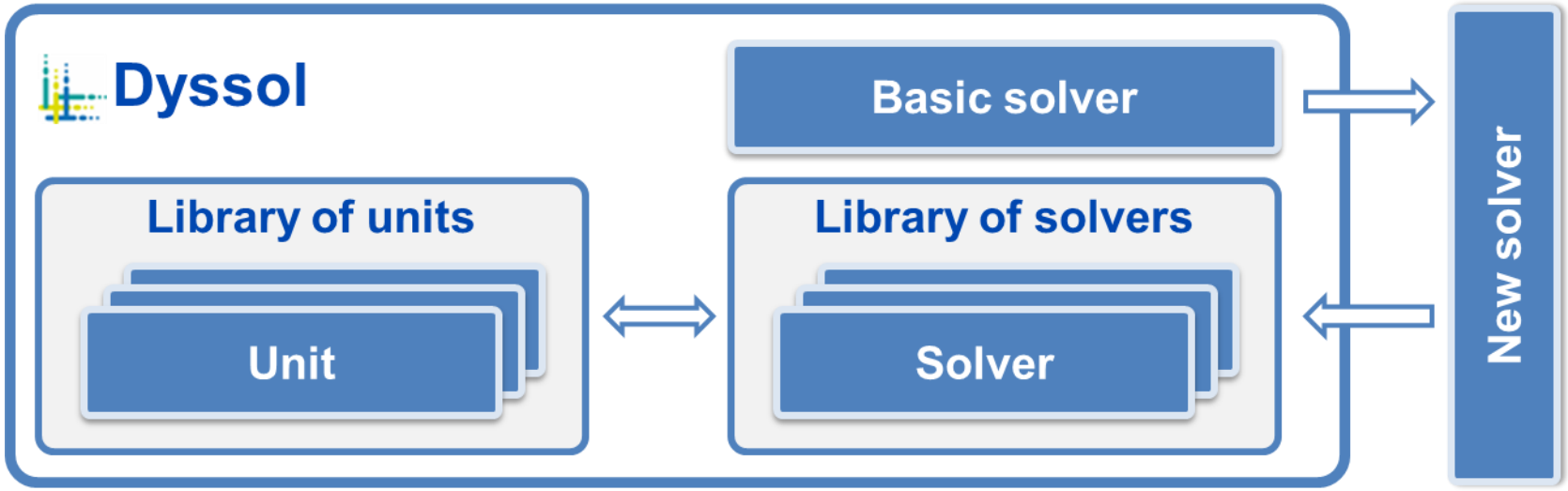
Ref.: Dyssol\Help\Units

Ref.: Dyssol\Help\Units development.pdf

Solvers


Library of units
Unit
Basic unit
Library of solvers
Solver
Basic solver
Material streams
Units
Simulation core
Flowsheet
- Process structure
- Process parameters
Simulator
- Main calculation algorithm
- Data transfer between units
User interface
- Create flowsheet
- Analyze results
Materials database
- Physical properties
- Compounds
Data storage subsystem
- File input/output
- Background data caching

Interfaces of solvers



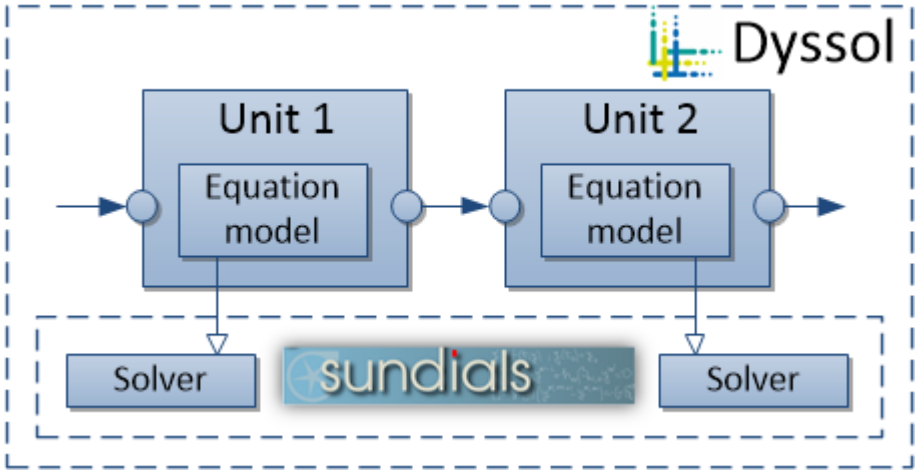
- Basic solver provides interfaces to the simulation system and to units
- New solvers can be implemented by users and added to the solvers library
- Can be added to a unit as a parameter
- Several aggregation solvers are available

Ref.: Dyssol\Help\Solvers.pdf

Ref.: Dyssol\Help\Solvers development.pdf

Build-in equation solvers

- IDA solver from SUNDIALS package can be used for automatic calculation of DAE systems inside the units
  - Applies variable-order, variable-coefficient backward differentiation formulas, in fixed-leading-coefficient form
- KINSOL solver from SUNDIALS package for calculation of nonlinear algebraic systems
  - Applies a fixed-point iteration with Anderson acceleration



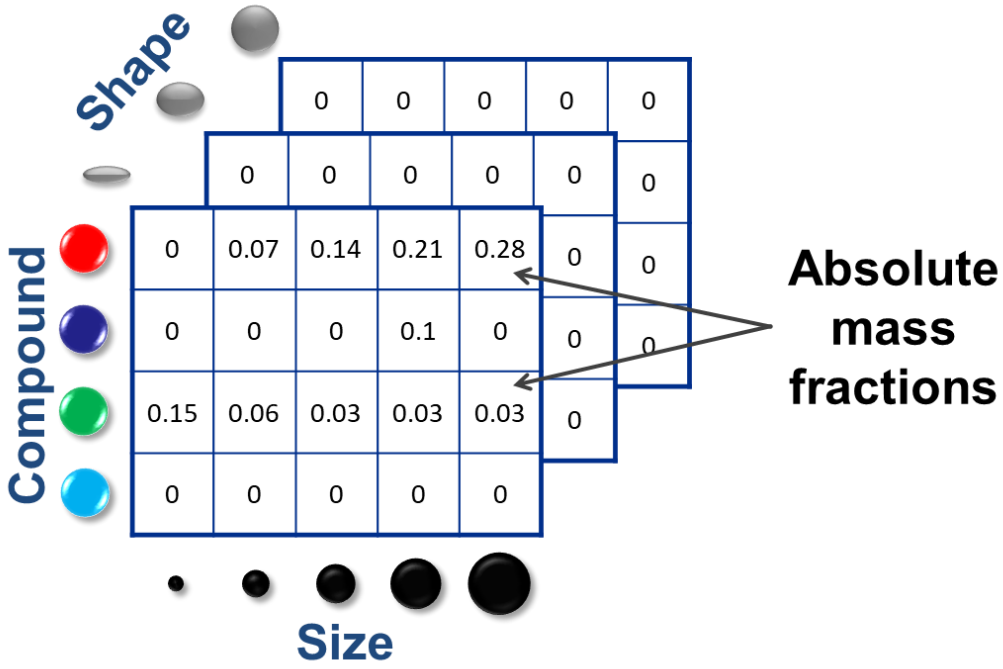
Skorych et al. Investigation of an FFT-based solver applied to dynamic flowsheet simulation of agglomeration processes. Advanced Powder Technology 30 (2019)

https://computation.llnl.gov/projects/sundials

Multidimensional distributed parameters of solids

Multidimensional distributed parameters

- Solid phase is described with a set of distributed parameters
- Parameters can be interdependent
- All interdependent distributed parameters form a multidimensional matrix
- Each entry of the matrix describes mass fractions of the solid material with the specified combination of parameters



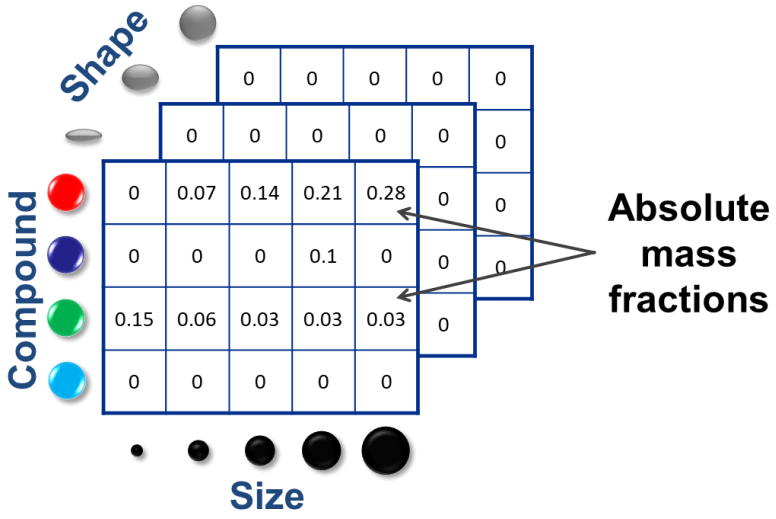
Example of a 3-dimensional distributed set of parameters

Currently available distributions:

- Compound
- Size
- Porosity
- Form factor
- Color
- User-defined

Tree data structures

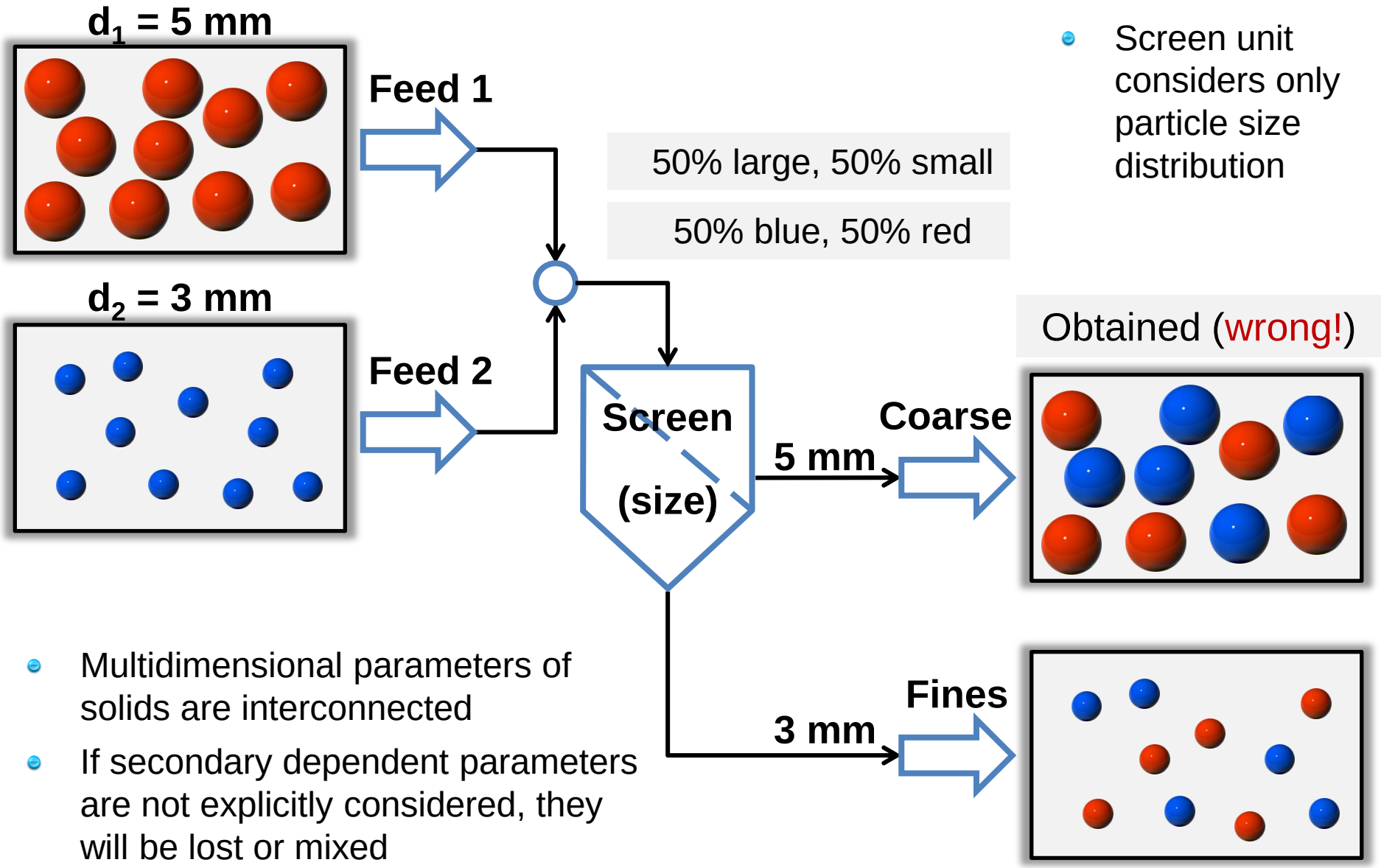
- Many entries of multidimensional matrices are equal to zero
- Amount of data can be reduced using sparse data formats
- Dyssol applies tree structures for data processing and storage
  - Reduction of memory consumptions
  - Improvement of computational efficiency



Skorych et al. Novel system for dynamic flowsheet simulation of solids processes. Powder Technology 314 (2017)

Treatment of dependent parameters

Explicit calculation



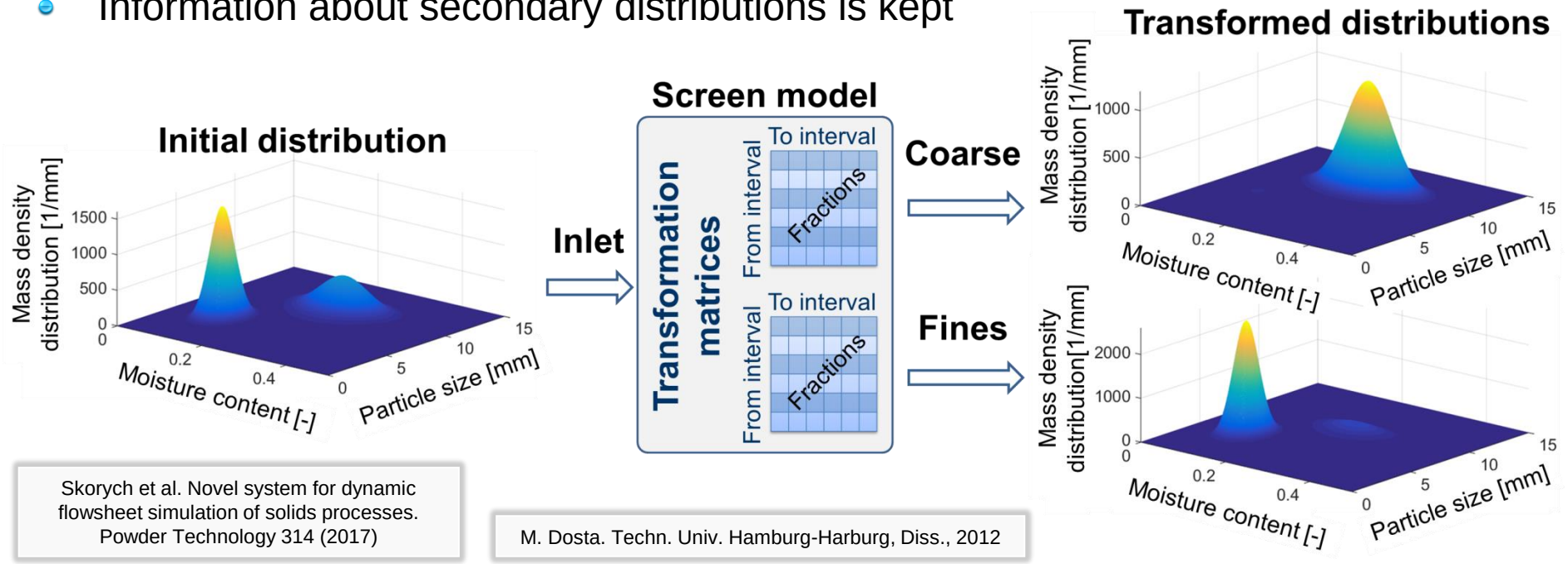
- Screen unit considers only particle size distribution

- Multidimensional parameters of solids are interconnected
- If secondary dependent parameters are not explicitly considered, they will be lost or mixed

Treatment of dependent parameters

Transformation matrices

- Transformation matrices (TM) instead of direct calculation of unit's outlets
- TM describe laws of material transition between classes
- Each entry of TM describes a fraction of material that passes from one class to another
- Calculate transformation matrixes based on model functions
- Apply TM to get distributions in holdups and outlets
- All dependent parameters of solids are calculated automatically
- Information about secondary distributions is kept

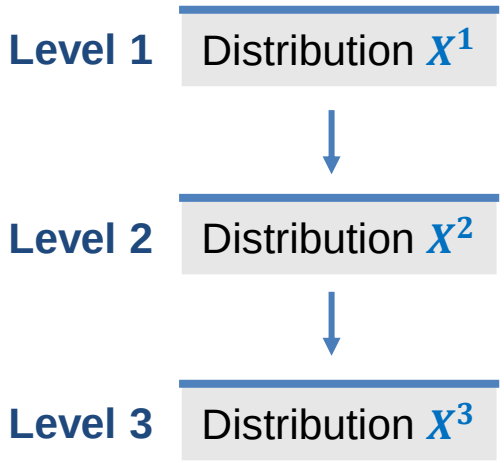


Skorych et al. Novel system for dynamic flowsheet simulation of solids processes. Powder Technology 314 (2017)

M. Dosta. Techn. Univ. Hamburg-Harburg, Diss., 2012

Application of transformation matrix

$$TreeStructure(t) \otimes T(t+\Delta t) = TreeStructure(t+\Delta t)$$

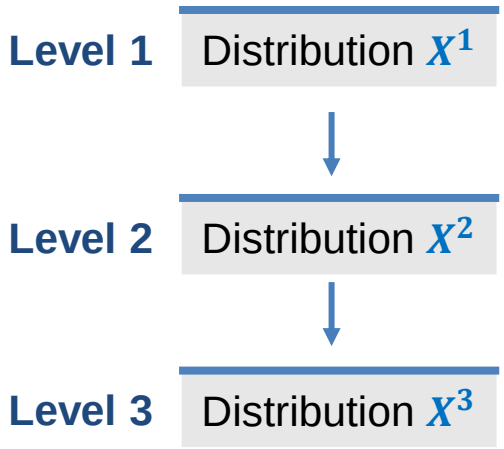


$$\otimes\, T\left[X^1\right] =$$

$$Z^1{}_{j_1} = \sum_{i_1=1}^{n_1} X^1_{i_1} \cdot T_{i_1,j_1}$$

$$Z^2{}_{j_1 j_2} = \frac{\sum_{i_1=1}^{n_1} X^2_{i_1 j_2} \cdot X^1_{i_1} \cdot T_{i_1,j_1}}{Z^1{}_{j_1}}$$

$$Z^3{}_{j_1 j_2 j_3} = \frac{\sum_{i_1=1}^{n_1} X^3_{i_1 j_2 j_3} \cdot X^2_{i_1 j_2} \cdot X^1_{i_1} \cdot T_{i_1,j_1}}{Z^2{}_{j_1 j_2} \cdot Z^1{}_{j_1}}$$



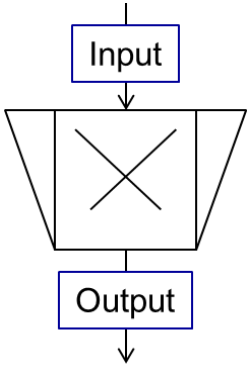
$$\otimes\, T\left[X^1 X^2\right] =$$

$$Z^1{}_{j_1} = \sum_{i_1=1}^{n_1} X^1_{i_1} \cdot T_{i_1,j_1}$$

$$Z^2{}_{j_1 j_2} = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} X^2_{i_1 i_2} \cdot X^1_{i_1} \cdot T_{i_1 i_2, j_1 j_2}$$

$$Z^3{}_{j_1 j_2 j_3} = \frac{\sum_{i_1=1}^{n_1} X^3_{i_1 j_2 j_3} \cdot X^2_{i_1 j_2} \cdot X^1_{i_1} \cdot T_{i_1,j_1}}{Z^2{}_{j_1 j_2} \cdot Z^1{}_{j_1}}$$

Example of transformation matrix



Output = TM ⊗ Input

Input — Particle size

Form factor

	1	2	3	4	5	6
1	0.000	0.010	0.015	0.050	0.010	0.090
2	0.030	0.025	0.030	0.070	0.030	0.100
3	0.020	0.030	0.050	0.018	0.025	0.110
4	0.030	0.040	0.034	0.010	0.010	0.080
5	0.020	0.020	0.024	0.000	0.005	0.014

Transformation matrix for size distribution

To interval

From interval

	1	2	3	4	5	6
1	1.00	0.00	0.00	0.00	0.00	0.00
2	0.50	0.50	0.00	0.00	0.00	0.00
3	0.25	0.25	0.50	0.00	0.00	0.00
4	0.00	0.25	0.25	0.50	0.00	0.00
5	0.00	0.00	0.25	0.25	0.50	0.00
6	0.00	0.00	0.00	0.33	0.33	0.34

0.009 = 0.000 * 1.00 + 0.010 * 0.50 + 0.015 * 0.25

0.050 = 0.030 * 1.00 + 0.025 * 0.50 + 0.030 * 0.25

0.036 = 0.050 * 0.50 + 0.018 * 0.25 + 0.025 * 0.25

Output — Particle size

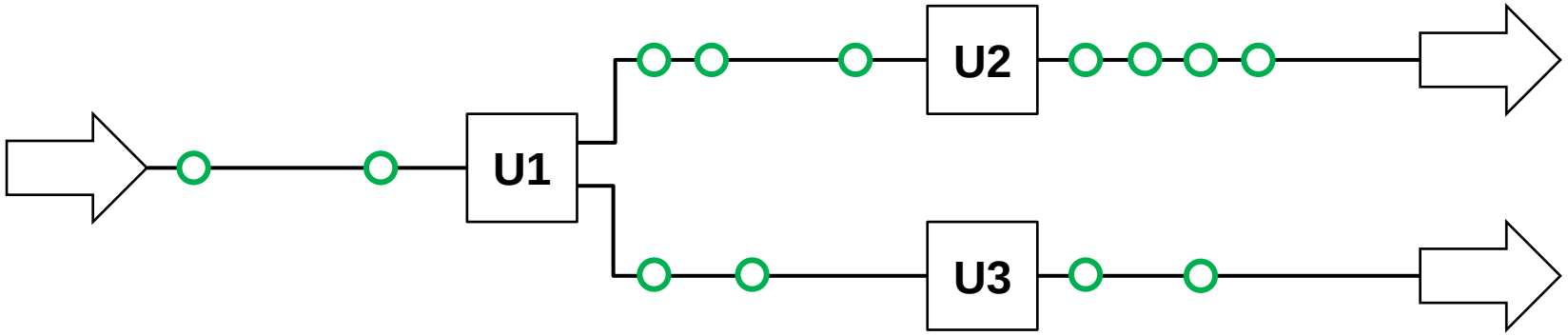
Form factor

	1	2	3	4	5	6
1	0.009	0.021	0.023	0.057	0.035	0.030
2	0.050	0.038	0.040	0.076	0.048	0.033
3	0.048	0.032	0.036	0.052	0.049	0.037
4	0.059	0.031	0.022	0.034	0.032	0.027
5	0.036	0.016	0.013	0.006	0.007	0.005

Time points management

Time points in streams

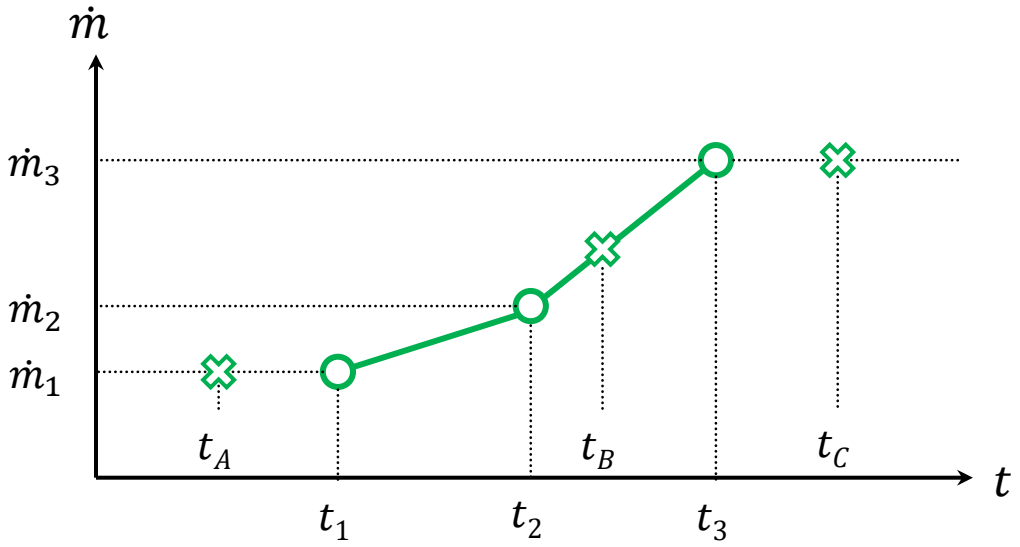
- Each stream may have unique combination of time points
- New time points may be generated during simulation
- Several sources of time points exist
  - Inlets
  - Dynamic units
  - Unit parameters



Interpolation of data

Data interpolation for all time-dependent parameters:

- Linear interpolation of data between existing time points
- Nearest-neighbor extrapolation outside of existing interval
- If only one time point is defined, it is assumed to be constant on the whole interval



$$\dot{m}(t_A) = \dot{m}(t_1) = \dot{m}_1$$

$$\dot{m}(t_B) = \dot{m}_2 + \frac{\dot{m}_3 - \dot{m}_2}{t_3 - t_2}(t_B - t_2)$$

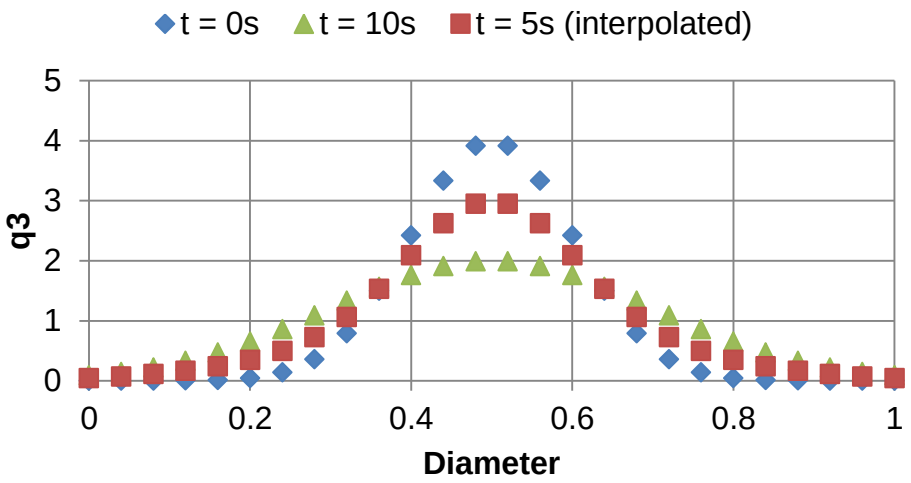
$$\dot{m}(t_C) = \dot{m}(t_3) = \dot{m}_3$$

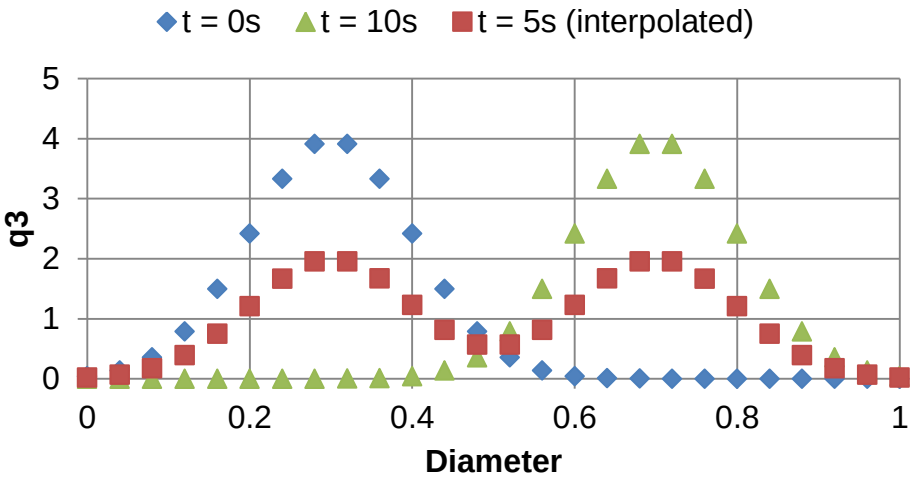
Interpolation of distributed parameters

- Distributed parameters are represented as a set of classes
- Interpolation is applied for each class

$$PSD_i = Interpol(PSD_{i-1}, PSD_{i+1})$$

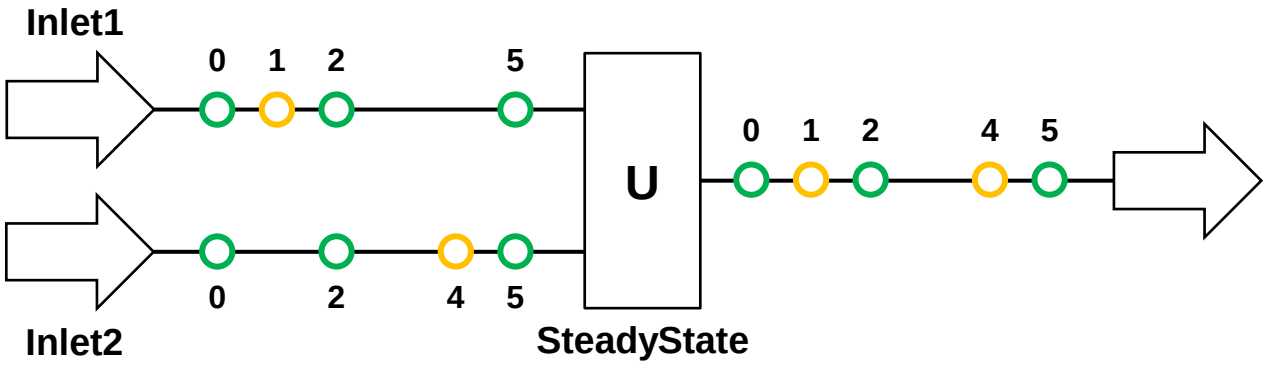
- Linear interpolation and nearest-neighbor extrapolation





Steady state units

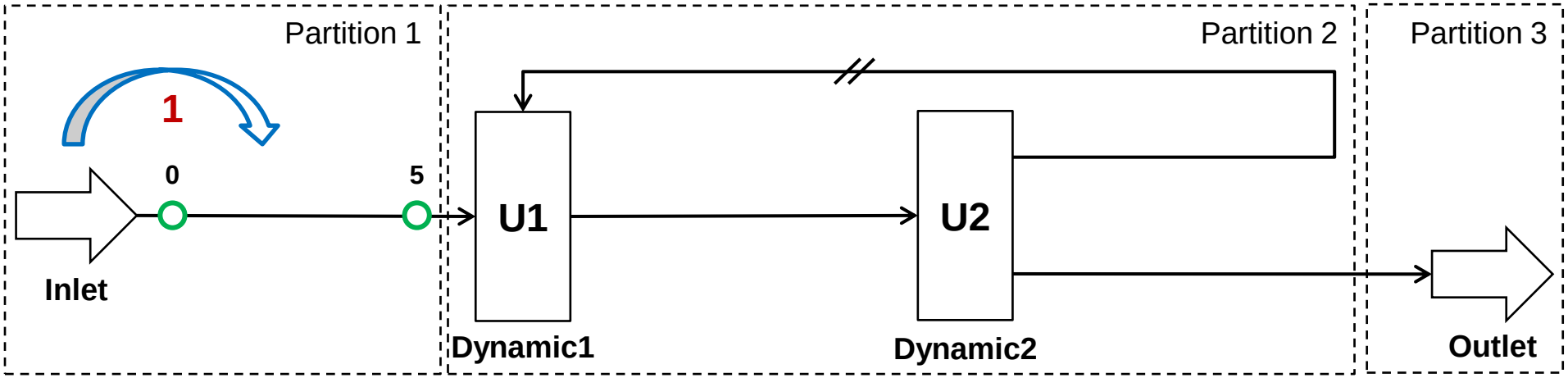
- Steady state unit is calculated on a union of time points from all its inlets
- Steady state unit does not produce new time points
- Data interpolation is used to obtain values from not existing time points



Dynamic units with recycle

Simulate on the time interval [0..5] s

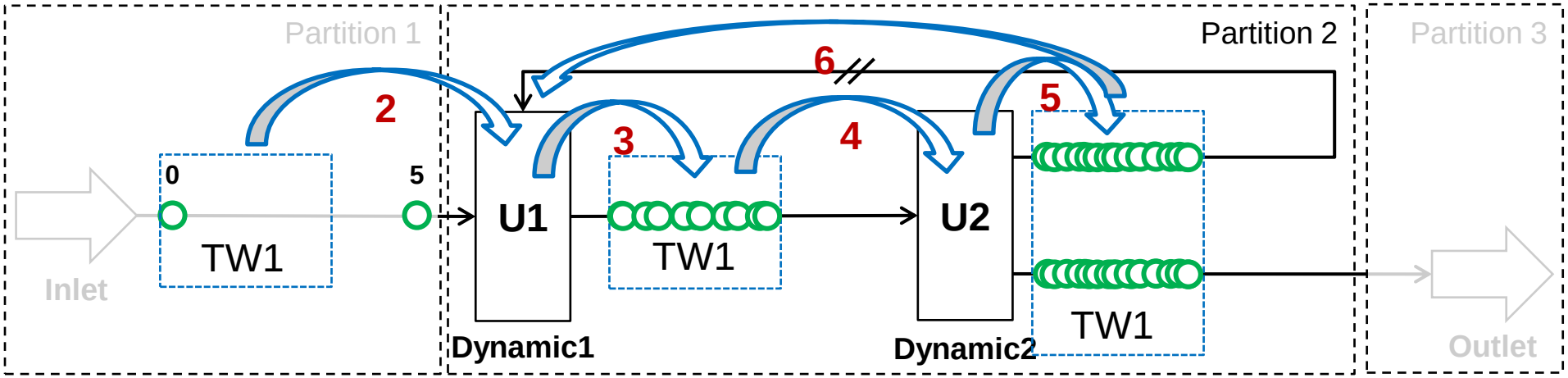
- Find recycle streams and split into partitions
- Simulate **Partition 1** on the whole time interval [0..5] s
  - `Inlet::Simulate(0, 5)`



Dynamic units with recycle

- Simulate **Partition 2** on the whole time interval [0..5] s
  - Apply waveform relaxation method: T_{WINDOW} (TW) = 3s
  - Simulate Partition 2 on the time interval [0..3] s
    - `Dynamic1::Simulate(0, 3)`
    - `Dynamic2::Simulate(0, 3)`
    - `Dynamic1::Simulate(0, 3)`
    - `Dynamic2::Simulate(0, 3)`

Repeat until convergence

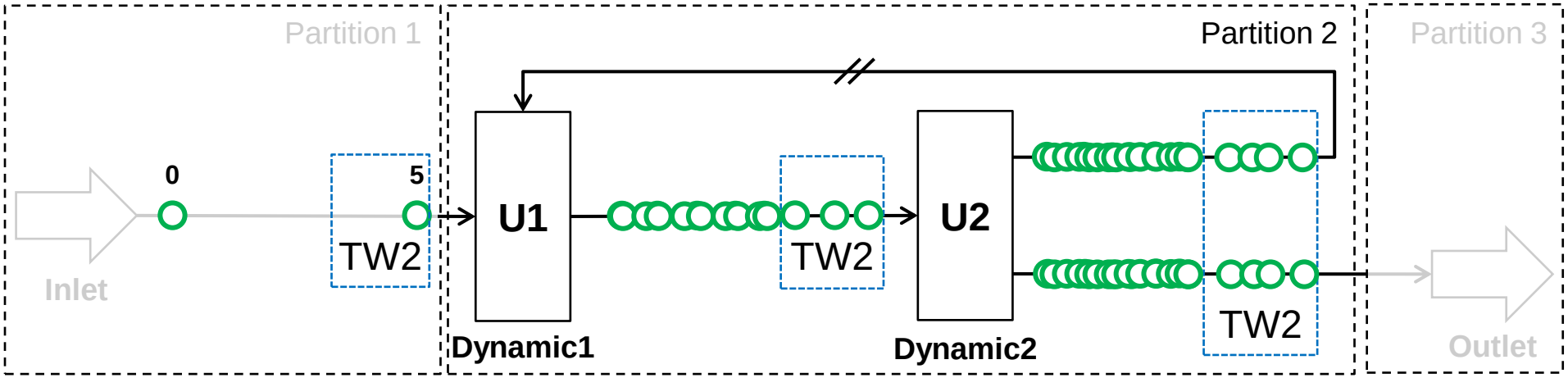


Dynamic units with recycle

- Simulate **Partition 2** on the whole time interval [0..5] s
  - Apply waveform relaxation method: T_{WINDOW} (TW) = 3s
  - Simulate Partition 2 on the time interval [0..3] s
    - `Dynamic1::Simulate(0, 3)`
    - `Dynamic2::Simulate(0, 3)`
    - `Dynamic1::Simulate(0, 3)`
    - `Dynamic2::Simulate(0, 3)`

    Repeat until convergence
  - Simulate Partition 2 on the time interval [3..5] s
    - `Dynamic1::Simulate(3, 5)`
    - `Dynamic2::Simulate(3, 5)`
    - `Dynamic1::Simulate(3, 5)`
    - `Dynamic2::Simulate(3, 5)`

    Repeat until convergence



Dynamic units with recycle

Simulate on the time interval [0..5] s

- Find recycle streams and split into partitions
- Simulate **Partition 1** on the whole time interval [0..5] s
- Simulate **Partition 2** on the whole time interval [0..5] s
  - Apply waveform relaxation method: T_{WINDOW} = 3s
  - Simulate Partition 2 on the time interval [0..3] s
  - Simulate Partition 2 on the time interval [3..5] s
- Simulate **Partition 3** on the whole time interval [0..5] s