Manual

The CADET Authors

CADET Version 4.2.0

Created from commit 2a6183cb1d96323a94991a0c49f3bf7bdea7ea1e made on Tue Jun 8 13:00:30 2021 +0200

Contents

List of Figures

List of Tables

1 Simulation

This chapter gives an overview of the simulation process and the different steps involved.

CADET uses a backward-differentiation-formula (BDF) time discretization as implemented by the IDAS solver from SUNDIALS [Hin+05]. Each time step requires the solution of a nonlinear algebraic system of equations is performed by a Newton method. Since chromatographic systems can exhibit strong nonlinearity and stiff systems, the Jacobian of the equation system is always updated (i.e., an “exact” Newton method is used).

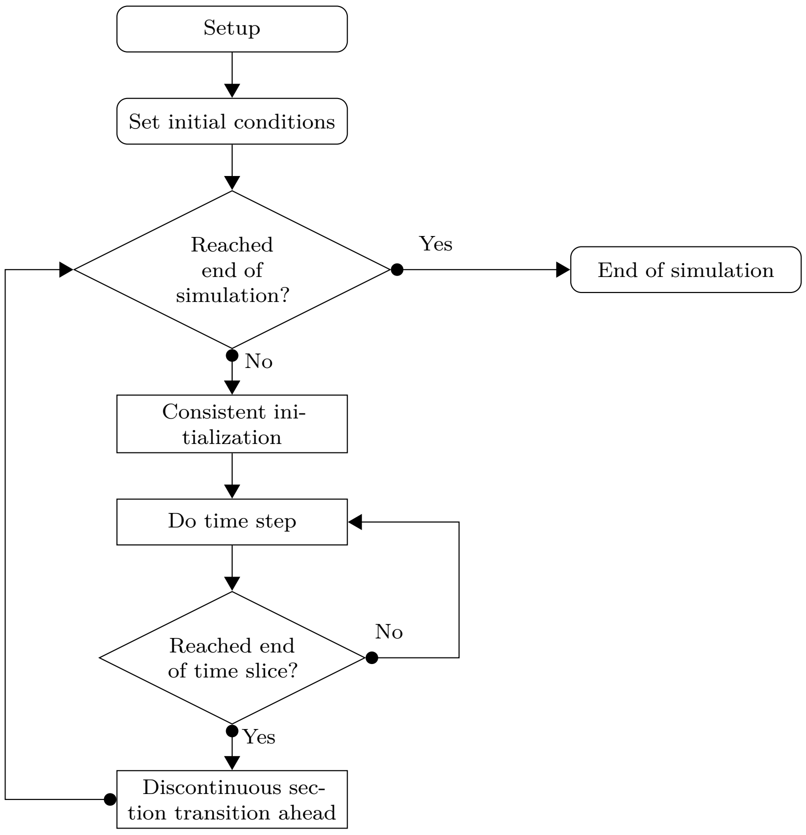

Setup
Set initial conditions
Reached end of simulation?
Yes
End of simulation
No
Consistent initialization
Do time step
Reached end of time slice?
No
Yes
Discontinuous section transition ahead

Figure 1.0.1: *General time integration procedure*

The general control flow of a simulation is shown in Fig. 1.0.1. Some aspects of the tasks involved are discussed below.

1.1 Time sections and transitions

The timespan $[t_0, t_{\text{end}}]$ of the simulation can be divided into multiple time sections $[t_i, t_{i+1}]$ with

$$t_0 < t_1 < \ldots < t_i < t_{i+1} < \ldots < t_{\text{end}}.$$

Time sections are used for various purposes, for example, for defining piecewise polynomials used as inlet profiles or external function, for changing operating conditions such as flow rates, or for changing the connectivity in a network of unit operations. A sequence of (one or more) time sections with smooth transitions is called a time slice (see Fig. 1.1.1).

A transition from one section to the next can either be smooth or discontinuous. On smooth transitions, the time integrator can just step over the section transition. Contrary to smooth transitions, a discontinuous transition requires some extra work in resetting the time integrator and finding consistent initial conditions for the next time slice.

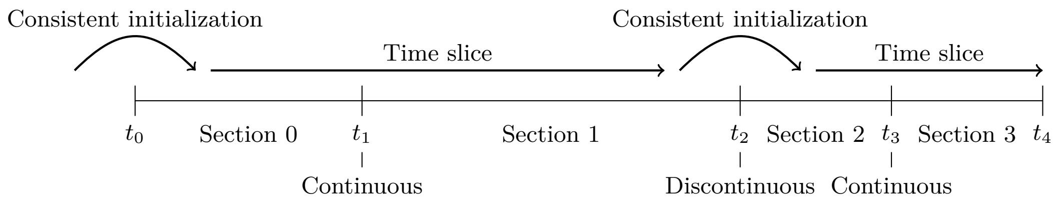


Figure 1.1.1: *Time sections, time slices, and transitions between them*

Time section specification Time sections are specified by an array (`SECTION_TIMES`, see Tab. 3.5.9) which contains the t_i that denote the start- and endpoint of a time section $[t_i, t_{i+1}]$. This means, that the array contains $n + 1$ values if there are n sections. The smoothness of a section transition is indicated by the array `SECTION_CONTINUITY`, which contains $n - 1$ elements if there are n sections.

1.2 Setup

In the setup phase, the model specification is checked and data structures are prepared for the simulation. The simulation can be run multiple times with different initial conditions, which also allows to continue a simulation. However, the model structure must not change after this point. This means, the different model and submodel types (e.g., unit operation model, binding model) as well as their discretization (i.e., number of cells) and structure (i.e., number of components, number of reactions) are fixed. On the contrary, their parameters (e.g., flow rates, porosities, dispersion coefficients) may change on different runs of a prepared simulation.

This assumption allows an accelerated simulation as there is no memory allocation performed during time integration.

1.3 Consistent initialization

At the beginning of the simulation and on discontinuous section transitions, consistent initial conditions have to be computed. In CADET, the general differential-algebraic equation (DAE)

$$F(t, y, \dot{y}, p) = 0$$

consists of purely algebraic equations that do not contain $\dot{y}$ and dynamic equations. Let $\mathcal{I}_d$ be the index set of dynamic equations and $\mathcal{I}_a$ the index set of algebraic equations. The general DAE can be decomposed into

$$M\dot{y}_{\mathcal{I}_d} + G_{\mathcal{I}_d}(t, y, p) = 0, \tag{1.1}$$

$$G_{\mathcal{I}_a}(t, y, p) = 0, \tag{1.2}$$

where M is the so called "mass matrix", which essentially consist of the state time derivative Jacobian $\partial F / \partial \dot{y}$. Furthermore, it is assumed that the Jacobian of the algebraic equations with respect to the algebraic variables $\partial G_{\mathcal{I}_a} / \partial y_{\mathcal{I}_a}$ is invertible. Hence, the original DAE is of differential index 1.

Given $y_{\mathcal{I}_d}$, consistently initialization means finding $y_{\mathcal{I}_a}$, $y_{\mathcal{I}_d}$, and $\dot{y}$ such that the DAE holds at the initial time point t_0 and can be solved for some non-empty time span. Thus, consistent initial conditions

y_0 and $\dot{y}_0$ satisfy

$$\begin{aligned}
0 &= M\dot{y}_{\mathcal{I}_d} + G_{\mathcal{I}_d}\left(t_0, y, p\right), \\
0 &= G_{\mathcal{I}_a}\left(t_0, y, p\right), \\
0 &= \frac{\partial G_{\mathcal{I}_a}}{\partial t}\left(t_0, y, p\right) + \frac{\partial G_{\mathcal{I}_a}}{\partial y}\left(t_0, \dot{y}, p\right).
\end{aligned}$$

The last equation, which determines $\dot{y}_{\mathcal{I}_a}$, arises from taking the total derivative with respect to time t of the second equation.

Concluding, a nonlinear algebraic equation system has to be solved in order to compute consistent initial conditions. The error in the solution of these systems is controlled by the `ALGTOL` setting (see Tab. 3.5.9).

1.4 Time stepping

Given the general differential-algebraic equation

$$F(t, y, \dot{y}, p) = 0, \qquad y\left(t_0\right) = y_0, \qquad \dot{y}\left(t_0\right) = \dot{y}_0$$

with consistent initial values y_0 and $\dot{y}_0$, the BDF discretization uses past time steps $y\left(t_{\tau-i}\right)$ to compute the current $\dot{y}\left(t_\tau\right)$ as

$$\dot{y}\left(t_\tau\right) = \frac{1}{\Delta t_\tau}\sum_{i=0}^{q_\tau} \alpha_{\tau,i} y\left(t_{\tau-i}\right),$$

where t_τ denotes some time step. The IDAS time integrator adaptively changes the step size $\Delta t_\tau = t_\tau - t_{\tau-1}$ and order $1 \leq q_\tau \leq 5$ in each time step and automatically determines the corresponding coefficients $\alpha_{\tau,i}$. This *ansatz* is inserted into the DAE and the resulting nonlinear algebraic equation system is solved for $y_\tau = y\left(t_\tau\right)$ using a Newton method.

$$H(y_\tau) = F\left(t_\tau, y_\tau, \frac{1}{\Delta t_\tau}\sum_{i=0}^{q_\tau} \alpha_{\tau,i}\, y_{\tau-i}, p\right) = 0 \tag{1.3}$$

This requires (possibly many) solutions of linear equation systems involving the Jacobian of H given by

$$\frac{\mathrm{d}H}{\mathrm{d}y_\tau} = \frac{\partial F}{\partial y} + \frac{\alpha_{\tau,0}}{\Delta t_\tau}\frac{\partial F}{\partial \dot{y}}.$$

The Jacobian of H, which is the linear combination of the partial derivatives $\partial F / \partial y$ and $\partial F / \partial \dot{y}$, can be calculated in two ways. The fastest way is the analytic computation, which is to be preferred if available. However, the implementation of the analytic Jacobian can be time consuming and complicated, especially when quickly testing new models. As a remedy, CADET offers to compute the complicated Jacobian $\partial F / \partial y$ automatically by automatic differentiation (AD). The much simpler Jacobian $\partial F / \partial \dot{y}$ has to be implemented manually.

Adaptivity The BDF order q_τ and size of the time step t_τ are chosen such that the error bounds are satisfied with minimal work. Equation (1.3) is solved with a Newton iteration using very few iterations (`MAX_NEWTON_ITER` from Table 3.5.9) since the starting point is assumed to be close to the solution as the time steps are small. If the solution of the Newton iteration does not pass the *convergence test*, the step size t_τ is reduced and the Newton iteration is tried again. This may happen at most `MAX_CONVTEST_FAIL` times, otherwise time integration is aborted with failure. Having a solution of the Newton iteration at hand, a *local truncation error test* is performed which controls the error of the BDF method. If this test fails, step size t_τ and order q_τ are adapted and the process restarts with solving the nonlinear system Eq. (1.3). The error test may be failed at most `MAX_ERRTEST_FAIL` times before time integration is aborted with failure.

Error bounds for the local truncation error test are specified by an absolute tolerance (`ABSTOL`) and a relative tolerance (`RELTOL`). Note that the relative tolerance only works for non-zero values, whereas zero values are accounted for by the absolute tolerance. For example, a relative tolerance of 10^{-4} and absolute tolerance of 10^{-8} requests 3 significant digits (correct digits after the comma in scientific notation) and considers all numbers with magnitude smaller than 10^{-8} as 0.

The size of the first time step in a time slice is given by `INIT_STEP_SIZE` from Table 3.5.9. If a simulation fails on the first time step, it might help to reduce the initial step size. It can also help to increase the error tolerances, at the cost of imprecise results over the whole course of the simulation.

It can happen that due to severe stiffness or ill-posed models a simulation may exceed its computational budget given by the maximum number of time steps (per time slice) `MAX_STEPS`. In this case, the formulation of the model should be checked and, if necessary, the error tolerances increased. Note that the number of time steps is independent of possible `USER_SOLUTION_TIMES` and refers to internal steps of the time integrator.

On the other extreme, the time integrator might jump over an important change in the solution or not capture some feature of the solution because of too large time steps. This can be avoided by specifying the maximum time step size `MAX_STEP_SIZE`. However, such a situation is unlikely to occur and can also be alleviated by placing a discontinuous section transition at the time of the event.

Note that while the error in time integration is controlled and can (in theory) be made arbitrarily small, the spatial discretization error is not. Hence, it is important and necessary to test different spatial resolutions (number of cells) in order to find a sufficiently accurate (but minimal) number of grid cells. This is particularly relevant for problems with steep fronts as a low spatial resolution adds a substantial amount of numerical dispersion to the solution.

1.5 Parameter sensitivities

Parameter sensitivities $s = \partial y / \partial p$ of a solution y to the DAE with respect to some parameter p are required for various tasks, for example, parameter estimation, process design, and process analysis. The CADET simulator implements the forward sensitivity approach which creates a linear companion DAE for each sensitive parameter

$$\begin{aligned} 0 = \frac{\mathrm{d}}{\mathrm{d}p} F(t, y, \dot{y}, p) &= \frac{\partial F}{\partial y}(t, y, \dot{y}, p) \frac{\partial y}{\partial p} + \frac{\partial F}{\partial \dot{y}}(t, y, \dot{y}, p) \frac{\partial \dot{y}}{\partial p} + \frac{\partial F}{\partial p}(t, y, \dot{y}, p) \\ &= \frac{\partial F}{\partial y}(t, y, \dot{y}, p) s + \frac{\partial F}{\partial \dot{y}}(t, y, \dot{y}, p) \dot{s} + \frac{\partial F}{\partial p}(t, y, \dot{y}, p). \end{aligned}$$

These linear DAEs depend on the solution y, $\dot{y}$ of the original DAE system. Consequently, the sensitivity systems are solved together with the original DAE system in a staggered approach [FTB97].

After the nonlinear Eq. (1.3) has been successfully solved using Newton iteration (i.e., it has passed the convergence test), each sensitivity is solved using the same Newton iteration. If direct linear solvers were used, this iteration would convergence with a single iteration as there is no nonlinearity. The Newton method for the sensitivities performs at most `MAX_NEWTON_ITER_SENS` (see Table 3.5.9) iterations. Assuming the convergence test has passed for each sensitivity, the local truncation error test is performed for the full set of variables (i.e., original system and sensitivities). The sensitivities can be excluded from the local truncation error test by setting `ERRORTEST_SENS` appropriately. Note that time integration step size is always affected by sensitivities due to possible convergence test failures.

While the Jacobians $\partial F / \partial y$ and $\partial F / \partial \dot{y}$ can be computed either analytically or via AD, the last term $\partial F / \partial p$ is always computed by AD. In fact, the terms $\partial F / \partial y$ and $\partial F / \partial p$ can be computed by one AD-enabled evaluation of F using the vector mode with appropriate seed vectors [Püt+16].

A parameter sensitivity is specified by identifying the involved parameters, which can be more than one (see below). Parameters are identified by name and multiple indices, which may not all be used (see Tab. 3.5.8). Indices that are not used for identification (e.g., flow rate is independent of bound phase or component) are set to -1. All other indices are zero-based.

Note that the sensitivity systems need to be consistently initialized, too. However, since only linear systems are involved, no nonlinear equation system has to be solved and the procedure is much simpler computationally.

Joint parameters The use of AD also enables support of meta parameters or joint parameters. Consider the situation in which several parameters $p_1, p_2, \ldots, p_n$ depend (linearly) on a single meta parameter p

$$p_i = p_i(p) = \alpha_i p \qquad \text{for } i = 1, \ldots, n$$

and some coefficients $\alpha_i \in \mathbb{R}$. Hence, the DAE residual function F becomes

$$F(t, y, \dot{y}, p_1(p), \ldots, p_n(p)) = 0$$

and the partial derivative $\partial F / \partial p$ can be computed automatically by AD:

$$\frac{\partial F}{\partial p} = \frac{\partial F}{\partial p_1}\alpha_1 + \ldots + \frac{\partial F}{\partial p_n}\alpha_n.$$

In order to use this functionality, all parameters involved have to be specified in a single sensitivity instance (`param_XXX` group, see Tables 3.5.8 and 3.5.8) by using arrays for the name and indices instead of scalar values. The coefficients α_i are provided in the `SENS_FACTOR` vector.

A simple example for a situation, which benefits from having a single meta parameter, would be a chain of unit operations in a network in which the sensitivity with respect to the flow rate is to be computed. Instead of computing the sensitivity with respect to each single flow rate and fusing them together in a postprocessing step, a meta parameter that maps to all flow rates can be introduced.

Note that nonlinear relationships between original parameter and meta parameters are supported by updating the coefficients α_i before each simulation:

$$\alpha_i = \frac{\partial p_i}{\partial p}.$$

2 Models

2.1 Network of unit operation models

Unit operation models can be composed into a network or graph, in which a node represents a unit operation and an edge denotes a connection between two unit operations. When utilized to full extent, this allows the simulation of complicated setups and processes (e.g., SMB, MCSGP). A more simple use case is the addition of plug flows and stirred tanks up- and downstream of a column in order to account for dead volume and additional dispersion from the tubing.

In a network, outlet ports of unit operations can be connected to any number of inlet ports of unit operations. Even direct cycles, where an outlet port of a unit operation is connected to its own inlet, are possible. A unit operation does not have to possess both inlet and outlet, but it has to have at least one of them. Pseudo unit operations such as inlet and outlet serve as sources and sinks for the network. However, the latter is not strictly required as any terminal node (i.e., a unit operation that possesses an outlet but does not have an outgoing connection) serves as a sink.

Each connection between two unit operation ports (i.e., an edge in the graph) is equipped with a volumetric flow rate that determines the mass flow from source to target port. These flow rates are used to determine the weight of the different incoming feeds at a unit operation's inlet port. Some unit operations can infer their internal flow rate (e.g., interstitial velocity) from their total incoming volumetric flow rate. In general, the mass balance at a unit operation has to be closed, except for unit operations that act as source or sink in the network and variable volume units (e.g., stirred tanks).

The network of unit operations uses "connection"-variables c_{con} to connect the different unit operation ports with each other. The inlet port variables $c_{\mathrm{in},n,k}$ of unit operation n are attached to $c_{\mathrm{con},n}$ via

$$c_{\mathrm{in},n,k,i} = c_{\mathrm{con},n,k,i}, \qquad k = 1, \dots, N_{\mathrm{port,in},n}, \quad i = 1, \dots, N_{\mathrm{comp},n}. \tag{2.1}$$

While $N_{\mathrm{port,in},n}$ denotes the number of inlet ports of unit operation n, the number of outlet ports is given by $N_{\mathrm{port,out},n}$. The connection variables $c_{\mathrm{con},n,k,i}$ collect all inflows of component i into port k of unit operation n:

$$c_{\mathrm{con},n,k,i} = \frac{\sum_{m=1}^{N_{\mathrm{units}}} \sum_{\ell=1}^{N_{\mathrm{port,out},n}} \sum_{j=1}^{N_{\mathrm{comp},m}} S_{(n,k,i),(m,\ell,j)} Q_{m,\ell} c_{\mathrm{out},m,\ell,j}}{\sum_{m=1}^{N_{\mathrm{units}}} \sum_{\ell=1}^{N_{\mathrm{port,out},m}} \hat{S}_{(n,k),(m,\ell)} Q_{m,\ell}}, \tag{2.2}$$

where $Q_{m,\ell}$ denotes the volumetric flow rate from outlet port ℓ of unit operation m, $S_{(n,k,i),(m,\ell,j)} \in \{0,1\}$ is a connection matrix indicating whether component i at outlet port k of unit operation n is connected to component j at inlet port ℓ of unit operation m, and $\hat{S}_{(n,k),(m,\ell)} \in \{0,1\}$ is another connection matrix indicating whether outlet port k of unit operation n is connected to inlet port ℓ of unit operation m, that is

$$\hat{S}_{(n,k),(m,\ell)} = \begin{cases} 1 & \text{if } \sum_{i=1}^{N_{\mathrm{comp},n}} \sum_{j=1}^{N_{\mathrm{comp},m}} S_{(n,k,i),(m,\ell,j)} \geq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Note that for each unit operation the number of inlet ports may be different from the number of outlet ports. Hence, the mass balance of a single unit operation is taken with respect to all its ports combined.

Specification of network connections The connections between the different unit operations in the network are specified by a table. There are two table formats:

- The long format includes seven columns. The first two columns specify source and destination unit operation id. The next two columns give source and destination port indices. Source and destination component indices are given by the following two columns. Finally, the seventh column specifies the volumetric flow rate of this connection.

- The short format includes five columns. The first two columns specify source and destination unit operation id. Source and destination component indices are given by the following two columns. Finally, the fifth column specifies the volumetric flow rate of this connection. Here, the omitted port indices default to -1, which connects all ports of the source unit operation to the corresponding ports of the target.

By default, the short format is used (i.e., a table with five columns is expected). However, if a unit operation with multiple ports is present, a table with seven columns is required. The default format can be overruled by setting a field (see Table 3.5.1).

With this setup, it is possible to connect single components of unit operations with each other yielding a maximum in flexibility. However, the predominant case is to connect all components of the source unit operations with their respective counterparts in the destination unit. This can easily be done by setting both component indices to -1 instead of writing a separate row for each component of the connection. The same setting (i.e., setting both port indices to -1) can be used to connect all ports of one unit operation with all corresponding ports of another one.

Note that in case of multiple rows for one connection between two unit operation ports (e.g., in case of separate component connections) the flow rate of the first row of that connection is used and all following flow rates are ignored. Consequently, there can only be one flow rate for a connection between two unit operations regardless of which components are connected.

The connection table is expected in row-major storage format (i.e., the rows are appended to one long array). See Table 3.5.1.

Valve switches The connectivity of the network can only change on a discontinuous section transition. Such a transition with changing connectivity is referred to as valve switch and the connectivity itself as valve configuration.

A list of valve configurations with at least one entry is required. Each valve configuration consists of a network connectivity table as described in Section 2.1 and a section index. The latter denotes the section in which the connectivity table becomes active. Hence, the one required (i.e., the first) entry must have a section index of 0 denoting the initial connectivity.

Note that the section index has to be monotonically increasing throughout the list of valve configurations. See Tables 3.5.1 and 3.5.1.

Dynamic flow rates The volumetric flow rates may vary over time while the valve configuration is active. The rates are assumed to be cubic polynomials,

$$Q = Q_0 + Q_1(t - t_s) + Q_2(t - t_s)^2 + Q_3(t - t_s)^3,$$

where t_s is the beginning of the time section that triggers the valve switch.

Note that the denominator in Eq. 2.2 must always be positive. That is, the flow rate coefficients have to be chosen such that the flow into every connected inlet port is strictly positive at all times.

Solution of the linear systems Each time step in the simulation requires the solution of a nonlinear system Eq. (1.3) (see Sec. 1.4). The nonlinear problem is solved by a Newton iteration, which, in turn, requires the solution of a linear system that essentially consists of the Jacobians of the unit operations and some coupling matrices from Eqs. (2.1) and (2.2).

These linear systems are either solved in parallel or sequentially. The parallel method first solves each unit operation (in parallel) to compute the solution at its outlet. Using these values, the inlets are adjusted and the unit operations are solved again. This is iterated until the system is fully solved.

In contrast, the sequential method first determines an ordering of the unit operations such that each unit only receives inflow from the previous units in the ordering. Such an ordering requires an acyclic graph of unit operations. Finally, the linear system is solved by solving the unit operations in the ordering determined above. Before a unit is solved, its inlet is calculated from the outlets of the previously solved units. This means, the system is solved from system inlets to system outlets.

The parallel method works regardless of the network topology (i.e., cycles in the graph), but requires to solve each unit operation at least twice. The sequential method solvs each unit exactly once, but is restricted to acyclic networks and works best for small graphs. By default, CADET uses a heuristic to select an appropriate solution method. This default can be overridden by a flag (see Table 3.5.1).

The solution method is selected for each valve switch individually. If some network configurations contain cycles, the parallel method is chosen for them regardless of the method used for the other configurations.

2.2 Unit operation models

A short comparison of the most prominent unit operation model features is given in Table 2.2.1.

Unit operation model	Radial diffusion	Particle diffusion	Film diffusion	Multiple-particle	Particle geome-tries
2D General rate model	✓	✓	✓	✓	✓
General rate model	✗	✓	✓	✓	✓
Lumped rate model with pores	✗	✗	✓	✓	✓
Lumped rate model without pores	✗	✗	✗	✗	✗
Continuous stirred tank	✗	✗	✗	✓	✗

Table 2.2.1: *Supported features of the different unit operation models*

2.2.1 Inlet

A system inlet unit operation is a pseudo unit operation since there is no physical correspondence. The inlet serves as a mass source in the network of unit operations. Consequently, it only possesses an outlet port and no inlet port. Note that an inlet unit operation can provide arbitrary many components and there can be arbitrary many inlet unit operations in a network.

An inlet unit operation provides a feed in which the concentration of each component is given by a profile. The most common profile is a piecewise cubic polynomial, which can both represent discontinuous signals (e.g., pulse or step) and smooth C^2 signals (cubic spline):

$$c_i(t) = \sum_{k=1}^{N_{\text{sect}}} \mathbb{1}_{[t_k, t_{k+1})}(t) \left[a_{k,i} \left(t - t_k\right)^3 + b_{k,i} \left(t - t_k\right)^2 + d_{k,i} \left(t - t_k\right) + f_{k,i} \right],$$

where $0 \leq t_1 < t_2 < \ldots < t_{N_{\text{sect}}+1} \leq T_{\text{sim}}$ is a decomposition of the simulation time interval $[0, T_{\text{sim}}]$ into pieces $[t_k, t_{k+1})$. On each piece, the profile is given by a cubic (fourth order) polynomial shifted to the beginning t_k of the piece.

See Tables 3.5.2 and 3.5.2.

2.2.2 Outlet

A system outlet unit operation is a pseudo unit operation since there is no physical correspondence. The outlet serves as a sink (terminal node) in the network of unit operations. Since any terminal node in the network is a sink (see Section 2.1), outlet unit operations are not strictly necessary. However, in some applications (e.g., SMB) only a certain fraction of a unit operation's output is taken out of the system and the rest is recycled. In this case, outlet unit operations are required in order to avoid unbalanced mass flow in the other unit operations.

Outlets can also be of help if the output of multiple unit operations merges together leaving the network. Instead of manually adding the streams together in a post-processing step, the unit operations can be connected to the same outlet unit.

See Table 3.5.2.

2.2.3 General rate model (GRM)

The general rate model is the most comprehensive model of mass transfer in column liquid chromatography, when only the axial coordinate in the column and the radial coordinate in the beads are considered [Kuč65; Gu95; Gui+06; FG04].

The main assumptions are:

1. The cross sections of the column are homogenous in terms of interstitial volume, fluid flow, and distribution of components. Thus, only one spatial coordinate in axial direction is needed and radial transport is neglected in the column bulk volume.
2. The bead radii r_p are much smaller than the column radius r_c and the column length L. Therefore, the beads can be seen as continuously distributed inside the column (i.e., at each point there is interstitial and bead volume).

Variable	Domain	Description
i	$\{0, \dots, N_{\text{comp}} - 1\}$	Component index
j	$\{0, \dots, N_{\text{partype}} - 1\}$	Particle type index
$m_{j,i}$	$\{0, \dots, N_{\text{bnd},j,i} - 1\}$	Bound state index of ith component in jth particle type
m_j	$\left\{0, \dots, \sum_{i=0}^{N_{\text{comp}}-1} N_{\text{bnd},j,i} - 1\right\}$	Total bound state index in particle type j
t	$[0, T_{\text{end}}]$	Time coordinate
z	$[0, L]$	Axial coordinate
r	$[r_{c,j}, r_{p,j}]$	Generic bead radial coordinate
$c^l_i(t, z)$	$[0, T_{\text{end}}] \times [0, L]$	Interstitial concentration of the ith component
$c^p_{j,i}(t, z, r)$	$[0, T_{\text{end}}] \times [0, L] \times [r_{c,j}, r_{p,j}]$	Mobile phase concentration of the ith component in the jth particle type
$c^s_{j,i,m_{j,i}}(t, z, r)$	$[0, T_{\text{end}}] \times [0, L] \times [r_{c,j}, r_{p,j}]$	Solid phase concentration of the ith component's $m_{j,i}$th bound state in particles of type j
$j_{f,j,i}(t, z)$	$[0, T_{\text{end}}] \times [0, L]$	Flux of the ith component through stagnant film into the bead of type j

Table 2.2.2: *Variables and unkowns*

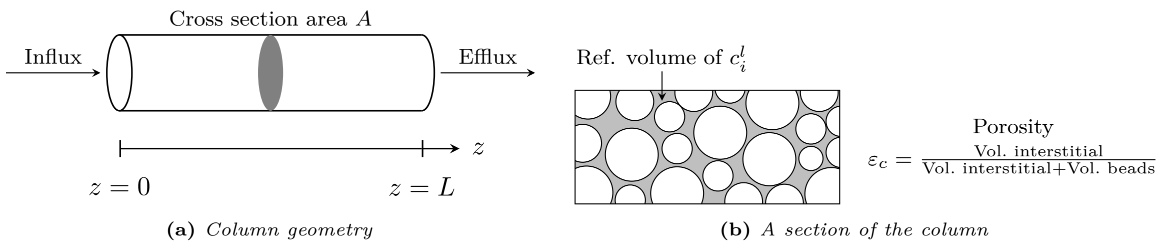

Cross section area A
Influx
Efflux
z
$z = 0$
$z = L$
(a) Column geometry
Ref. volume of c^l_i
Porosity
$\varepsilon_c = \frac{\text{Vol. interstitial}}{\text{Vol. interstitial+Vol. beads}}$
(b) A section of the column

Figure 2.2.1: *Column bulk model*

The GRM describes transport of solute molecules through the interstitial column volume by convective flow, band broadening caused by axial dispersion, mass transfer resistance through a stagnant film around the beads, pore (and surface) diffusion in the porous beads [MWW96; SS68; Miy07], and adsorption to the inner bead surfaces.

Consider a column of length $L > 0$ filled with spherical beads of (possibly) multiple types with radius $r_{p,j} \ll L$ (see Fig. 2.2.1), where j is the particle type index. The mass balance in the interstitial column volume is described by

$$\frac{\partial c^l_i}{\partial t} = -u \frac{\partial c^l_i}{\partial z} + D_{\text{ax},i} \frac{\partial^2 c^l_i}{\partial z^2} - \frac{1}{\beta_c} \sum_j d_j \frac{3}{r_{p,j}} k_{f,j,i} \left[c^l_i - c^p_{j,i}(\cdot, \cdot, r_{p,j}) \right] + f^l_{\text{react},i}\left(c^l\right). \tag{2.3}$$

Here, $c^l_i\colon\ [0, T_{\text{end}}]\times[0, L] \to \mathbb{R}^{\geq 0}$ denotes the concentration in the interstitial column volume, $c^p_{j,i}\colon\ [0, T_{\text{end}}]\times [0, L] \times [r_{c,j}, r_{p,j}] \to \mathbb{R}^{\geq 0}$ the liquid phase concentration in the beads, $k_{f,j,i}$ the film diffusion coefficient, $D_{\text{ax},i}$ the dispersion coefficient, u the interstitial velocity, d_j the volume fraction of particle type j, and $\beta_c = \varepsilon_c/(1 - \varepsilon_c)$ the column phase ratio, where ε_c is the column porosity (ratio of interstitial volume to total column volume). If reactions are considered, the term $f^l_{\text{react},i}\left(c^l\right)$ represents the net change of concentration c_i due to reactions involving component i.

Danckwerts boundary conditions [Dan53] are applied to inlet and outlet of the column:

$$u c_{\text{in},i}(t) = u c^l_i(t, 0) - D_{\text{ax},i} \frac{\partial c^l_i}{\partial z}(t, 0) \qquad \forall t > 0, \tag{2.4}$$

$$\frac{\partial c^l_i}{\partial z}(t, L) = 0 \qquad \forall t > 0. \tag{2.5}$$

Note that the outlet boundary condition Eq. (2.5) is also known as “do nothing” or natural outflow condition.

In the liquid phase of the porous beads (see Fig. 2.2.2) the mass balance is given by

$$\begin{aligned}\frac{\partial c^p_{j,i}}{\partial t} + \frac{1 - \varepsilon_{p,j}}{F_{\text{acc},j,i}\varepsilon_{p,j}} \frac{\partial}{\partial t} \sum_{m_{j,i}} c^s_{j,i,m_{j,i}} =& \underbrace{D_{p,j,i} \left[\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right] c^p_{j,i}}_{\text{Pore diffusion}} \\ &+ \underbrace{\frac{1 - \varepsilon_{p,j}}{F_{\text{acc},j,i}\varepsilon_{p,j}} D_{s,j,i} \left[\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right] \sum_{m_{j,i}} c^s_{j,i,m_{j,i}}}_{\text{Surface diffusion}} \\ &+ f^p_{\text{react},j,i}\left(c^p_j, c^s_j\right) + \frac{1 - \varepsilon_{p,j}}{F_{\text{acc},j,i}\varepsilon_{p,j}} f^s_{\text{react},j,i}\left(c^p_j, c^s_j\right),\end{aligned} \tag{2.6}$$

where $c^s_{j,i,m_{j,i}}\colon\ [0, T_{\text{end}}] \times [0, L] \times [r_{c,j}, r_{p,j}] \to \mathbb{R}^{\geq 0}$ denotes the solid phase concentration of the ith component’s $m_{j,i}$th bound state in the beads of jth type, $D_{p,j,i}$ the effective diffusion coefficient in the beads, $D_{s,j,i}$ the surface diffusion coefficient, $F_{\text{acc},j,i} \in [0, 1]$ the pore accessibility factor, and $\varepsilon_{p,j}$ the particle porosity (ratio of pore volume to total bead volume). The inner bead radius $r_{c,j} \in [0, r_{p,j})$ is assumed to be 0 by default, but can be positive in order to account for core-shell particles that have an impermeable core. Reaction terms in liquid and solid phase are collected in $f^p_{\text{react},j,i}(c^p_j, c^s_j)$ and $f^s_{\text{react},j,i}(c^p_j, c^s_j)$, respectively.

The GRM is used with both quasi-stationary (Eq. (2.7)) and dynamic (Eq. (2.8)) binding models.

$$\text{quasi-stationary:} \quad 0 = f_{\text{ads},j}\left(c^p_j, c^s_j\right), \tag{2.7}$$

$$\text{dynamic:} \quad \frac{\partial c^s_j}{\partial t} = D_{s,j}\left[\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right] c^s_j + f_{\text{ads},j}\left(c^p_j, c^s_j\right) + f^s_{\text{react},j}\left(c^p_j, c^s_j\right). \tag{2.8}$$

Note that c^p_j and c^s_j denote the vector of all $c^p_{j,i}$ and $c^s_{j,i,m_{j,i}}$, respectively.

The boundary conditions of the bead model the film diffusion and are given for all $t \in (0, \infty)$ and $z \in [0, L]$ by

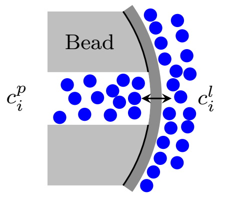
$$k_{f,j,i}\left[c^l_i - c^p_{j,i}(\cdot, \cdot, r_{p,j})\right] = F_{\text{acc},j,i}\varepsilon_{p,j} D_{p,j,i} \frac{\partial c^p_{j,i}}{\partial r}(\cdot, \cdot, r_{p,j}) + (1 - \varepsilon_{p,j})\, D_{s,j,i} \sum_{m_{j,i}} \frac{\partial c^s_{j,i,m_{j,i}}}{\partial r}(\cdot, \cdot, r_{p,j}), \tag{2.9}$$

$$\frac{\partial c^p_{j,i}}{\partial r}(\cdot, \cdot, r_{c,j}) = 0. \tag{2.10}$$

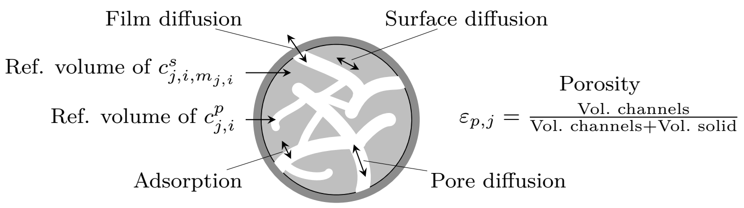
By default, the following initial conditions are applied for all $z \in [0, L]$ and $r \in [r_{c,j}, r_{p,j}]$:

$$c^l_i(0, z) = 0, \qquad c^p_{j,i}(0, z, r) = 0, \qquad c^s_{j,i,m_{j,i}}(0, z, r) = 0. \tag{2.11}$$

See Table 3.5.2.



(a) *Mass transport into beads*


Film diffusion
Surface diffusion
Ref. volume of $c^s_{j,i,m_{j,i}}$
Ref. volume of $c^p_{j,i}$
Porosity
$\varepsilon_{p,j} = \frac{\text{Vol. channels}}{\text{Vol. channels+Vol. solid}}$
Adsorption
Pore diffusion

(b) *Mass transport inside bead*

Figure 2.2.2: *Column bead model*

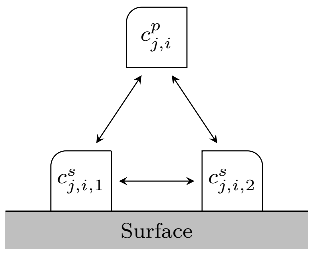


Figure 2.2.3: *Binding with multiple bound states*

Particle geometry In the model above, spherical particles are considered. Other supported particle forms are cylinders and slabs. For cylinders, it is assumed that molecules can only enter through the lateral surface (i.e., the caps are sealed). Slabs are assumed to have two large sides such that molecules enter through the two large faces (i.e., the remaining four small faces are sealed).

All particle forms support core-shell beads that have an impermeable core. The particles are characterized by their (outer) “radius” $r_{p,j}$ and their (inner) core “radius” $r_{c,j} \in [0, r_{p,j})$. See Fig. 2.2.4.

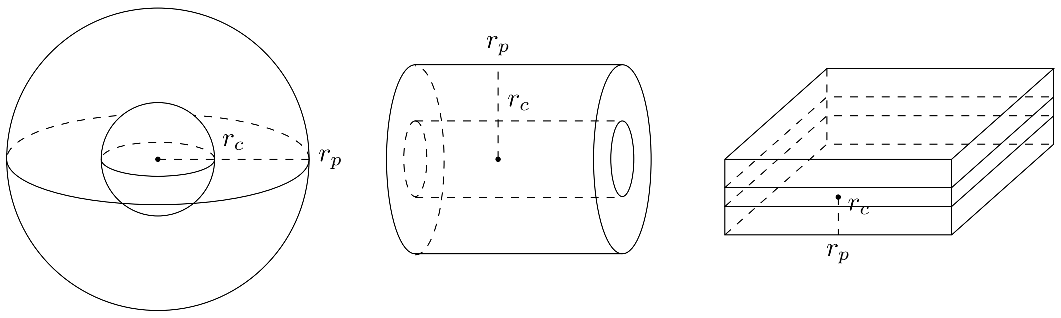


Figure 2.2.4: *Particle geometries*

For cylinders, the factor $3/r_{p,j}$ in (2.3) changes to $2/r_{p,j}$ and the diffusion operator in (2.6) and (2.8) changes as

$$\left[\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right] \quad \to \quad \left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right].$$

For slabs, the factor $3/r_{p,j}$ in (2.3) changes to $1/r_{p,j}$ and the diffusion operator in (2.6) and (2.8) changes as

$$\left[\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right] \quad \to \quad \frac{\partial^2}{\partial r^2}.$$

Multiple particle types A particle type has its own set of mass transfer and geometry parameters $\varepsilon_{p,j}$, $D_{p,j}$, $D_{s,j}$, etc. (see Eq. (2.6)) and its own binding model f_{ads} (including a possibly differing number of bound states). This allows, for example, modeling of particle size distributions or potential applications with differently functionalized beads (e.g., immobilized enzymes).

The distribution of the particle types is governed by their volume fractions d_j in Eq. (2.3). The volume fractions have to sum to 1:

$$\sum_{j=0}^{N_{\text{partype}}-1} d_j = 1.$$

The particle type volume fractions can be spatially constant throughout the column, or depend on the position inside the column bulk volume. In the latter case, the user can specify a set of volume fractions for each discretized finite volume cell. This allows, for example, the placement of smaller particles near the frits.

Size exclusion chromatography The general rate model can be used to simulate size exclusion chromatography (SEC) [Gu95]. The particle porosity $\varepsilon_{p,j}$ on the mobile phase side of the transport equations is replaced by a component-dependent accessible porosity

$$\varepsilon_{p,j,i} = F_{\text{acc},j,i}\varepsilon_{p,j}, \tag{2.12}$$

where the pore accessibility factor $F_{\text{acc},j,i}$ ranges in $(0,1]$.

Small molecules that can enter any pore have $F_{\text{acc},j,i} = 1$, whereas larger molecules that can enter some, but not small pores, have values $0 < F_{\text{acc},j,i} < 1$. The other extreme is given by molecules so large that they cannot enter any pore and, consequently, $F_{\text{acc},j,i} = 0$. Note that $F_{\text{acc},j,i} = 0$ is not allowed in a simulation, which can be circumvented by setting $k_{f,j,i} = 0$.

By default, $F_{\text{acc},j,i} = 1$ for all components i and all particle types j, which disables size exclusion chromatography. In order to simulate pure SEC, binding is disabled by setting $N_{\text{bnd},i} = 0$ for all components i and applying no binding model. If adsorption is present, it is important to note that any saturation capacity (e.g., $q_{\max}$ of Langmuir-type binding models) is subject to the full pore volume fraction $\varepsilon_{p,j}$.

Note that multiple particle types can also be used to aid in modeling size exclusion effects, see Section 2.2.3.

Specification of flow rate / velocity and direction Since volumetric flow rates are specified for each network connection, the unit operation can infer its interstitial velocity via

$$u = u_{\text{int}} = \frac{F_{\text{in}}}{A\varepsilon_c},$$

where F_{in} denotes the volumetric flow rate and A the cross section area. Note that without the bulk porosity ε_c, the superficial velocity would be obtained.

The direction of flow inside the unit operation is governed by the sign of the interstitial velocity u. A positive sign results in (standard) forward flow, whereas a negative sign reverses the flow direction. Note that in case of reversed flow, the chromatogram is returned at the unit operation's *inlet*, which may not be returned from simulation by default.

The final behavior is controlled by the interplay of cross section area and interstitial velocity:

1. If cross section area A is given and u is not, u is inferred from the volumetric flow rate.
2. If u is given and A is not, the volumetric flow rate is ignored and the provided interstitial velocity is used.
3. If both cross section area A and interstitial velocity u are given, the magnitude of the actual interstitial velocity u is inferred from the volumetric flow rate and the flow direction is given by the sign of the provided u.

2.2.4 Lumped rate model with pores (LRMP)

The lumped rate model with pores [Gui+06; FG04] deviates from the general rate model (see Section 2.2.3) by neglecting pore diffusion. The particle phase c^p_j is still there, but no mass transfer happens except for binding and film diffusion. Hence, the model equations are given by

$$\frac{\partial c^l_i}{\partial t} = -u\frac{\partial c^l_i}{\partial z} + D_{\mathrm{ax},i}\frac{\partial^2 c^l_i}{\partial z^2} - \frac{1}{\beta_c}\sum_j d_j \frac{3}{r_{p,j}} k_{f,j,i}\left[c^l_i - c^p_{j,i}\right] + f^l_{\mathrm{react},i}\left(c^l\right), \tag{2.13}$$

$$\begin{aligned}\frac{\partial c^p_{j,i}}{\partial t} + \frac{1-\varepsilon_{p,j}}{F_{\mathrm{acc},j,i}\varepsilon_{p,j}}\frac{\partial}{\partial t}\sum_{m_{j,i}} c^s_{j,i,m_{j,i}} &= \frac{3}{F_{\mathrm{acc},j,i}\varepsilon_{p,j} r_{p,j}} k_{f,j,i}\left[c^l_i - c^p_{j,i}\right] \\ &\quad + f^p_{\mathrm{react},j,i}\left(c^p_j, c^s_j\right) + \frac{1-\varepsilon_{p,j}}{F_{\mathrm{acc},j,i}\varepsilon_{p,j}} f^s_{\mathrm{react},j,i}\left(c^p_j, c^s_j\right)\end{aligned} \tag{2.14}$$

with the same meanings of variables and parameters as in the general rate model. The equations are complemented by Danckwerts boundary conditions [Dan53]

$$\begin{aligned} u c_{\mathrm{in},i}(t) &= u c^l_i(t,0) - D_{\mathrm{ax},i}\frac{\partial c^l_i}{\partial z}(t,0) && \forall t > 0, \\ \frac{\partial c^l_i}{\partial z}(t,L) &= 0 && \forall t > 0. \end{aligned}$$

As for the general rate model, both quasi-stationary and dynamic binding models are supported:

$$\begin{aligned} \text{quasi-stationary:} \quad 0 &= f_{\mathrm{ads},j}\left(c^p_j, c^s_j\right), \\ \text{dynamic:} \quad \frac{\partial c^s_j}{\partial t} &= f_{\mathrm{ads},j}\left(c^p_j, c^s_j\right) + f^s_{\mathrm{react},j}\left(c^p_j, c^s_j\right). \end{aligned}$$

By default, the following initial conditions are applied for all $z \in [0, L]$:

$$c^l_i(0,z) = 0, \qquad c^p_{j,i}(0,z) = 0, \qquad c^s_{j,i,m_{j,i}}(0,z) = 0. \tag{2.15}$$

Multiple particle types and all particle geometries are supported (see Section 2.2.3). This model can also be used to simulate size exclusion chromatography (see Section 2.2.3). For the specification of flow rate and direction, the same holds as for the general rate model (see Section 2.2.3). See Table 3.5.2.

2.2.5 Lumped rate model without pores (LRM)

The lumped rate model without pores [Gui+06; FG04] deviates from the lumped rate model with pores (see Section 2.2.4) by neglecting pores completely. The particle phase c^p is removed and the porosity ε_t is taken as total porosity

$$\varepsilon_t = \varepsilon_c + (1-\varepsilon_c)\,\varepsilon_p. \tag{2.16}$$

The phase ratio is denoted by $\beta_t = \varepsilon_t/(1-\varepsilon_t)$ accordingly. The model equations are given by

$$\frac{\partial c^l_i}{\partial t} + \frac{1}{\beta_t}\frac{\partial}{\partial t}\sum_{m_i} c^s_{i,m_i} = -u\frac{\partial c^l_i}{\partial z} + D_{\mathrm{ax},i}\frac{\partial^2 c^l_i}{\partial z^2} + f^l_{\mathrm{react},i}\left(c^l, c^s\right) + \frac{1}{\beta_t} f^s_{\mathrm{react},i}\left(c^l, c^s\right), \tag{2.17}$$

where $\beta_t = \varepsilon_t/(1-\varepsilon_t)$ denotes the (total) phase ratio. The equations are complemented by Danckwerts boundary conditions [Dan53]

$$\begin{aligned} u c_{\mathrm{in},i}(t) &= u c^l_i(t,0) - D_{\mathrm{ax},i}\frac{\partial c^l_i}{\partial z}(t,0) && \forall t > 0, \\ \frac{\partial c^l_i}{\partial z}(t,L) &= 0 && \forall t > 0. \end{aligned}$$

Both quasi-stationary and dynamic binding models are supported:

$$\begin{aligned}\text{quasi-stationary:}\quad 0 &= f_{\mathrm{ads}}\left(c^l, c^s\right), \\ \text{dynamic:}\quad \frac{\partial q}{\partial t} &= f_{\mathrm{ads}}\left(c^l, c^s\right) + f_{\mathrm{react}}^s\left(c^l, c^s\right).\end{aligned}$$

By default, the following initial conditions are applied for all $z \in [0, L]$:

$$c_i^l(0, z) = 0, \qquad c_{i,m_i}^s(0, z) = 0. \tag{2.18}$$

Note that by setting $\varepsilon_t = 1$, removing all bound states by setting $N_{\mathrm{bnd},i} = 0$ for all components i, and applying no binding model, a dispersive plug flow reactor (DPFR) is obtained.

For the specification of flow rate and direction, the same holds as for the general rate model (see Section 2.2.3). See Table 3.5.2.

2.2.6 Two Dimensional General rate model (GRM2D)

The general rate model as introduced in Section 2.2.3 assumes homogeneity in the cross sections of the column. This allows to consider transport along the axial dimension only. However, due to packing irregularity and inhomogeneous flow at the inlet (i.e., frits), this assumption may be a crude approximation. This model can be improved by introducing a radial coordinate $\rho \in [0, R]$, where R is the column radius, in the interstitial volume Eq. (2.3):

$$\begin{aligned}\varepsilon_c \frac{\partial c_i^l}{\partial t} = &- \varepsilon_c u \frac{\partial c_i^l}{\partial z} + \varepsilon_c D_{\mathrm{ax},i} \frac{\partial^2 c_i^l}{\partial z^2} + \frac{1}{\rho}\frac{\partial}{\partial \rho}\left(\rho D_{\mathrm{rad},i} \frac{\partial}{\partial \rho}\left(\varepsilon_c c_i^l\right)\right) \\ &- (1-\varepsilon_c) \sum_j d_j \frac{3 k_{f,j,i}}{r_{p,j}} \left[c_i^l - c_{j,i}^p(\cdot,\cdot,\cdot,r_{p,j})\right] + \varepsilon_c f_{\mathrm{react},i}^l\left(c^l\right).\end{aligned} \tag{2.19}$$

Here, $c_i^l\colon [0, T_{\mathrm{end}}] \times [0, L] \times [0, R] \to \mathbb{R}^{\geq 0}$, $c_{j,i}^p\colon [0, T_{\mathrm{end}}] \times [0, L] \times [0, R] \times [r_{c,j}, r_{p,j}] \to \mathbb{R}^{\geq 0}$, and $c_{j,i,m_{j,i}}^s\colon [0, T_{\mathrm{end}}] \times [0, L] \times [0, R] \times [r_{c,j}, r_{p,j}] \to \mathbb{R}^{\geq 0}$ depend on ρ. Additionally, the porosity ε_c, axial dispersion coefficient $D_{\mathrm{ax},i}$, radial dispersion coefficient $D_{\mathrm{rad},i}$, and interstitial velocity u may depend on ρ.

The dependence of the parameters on ρ is not arbitrary. For simplicity, it is assumed that the parameters are piecewise constant, that is, the range $[0, R]$ is divided into disjoint zones in which all parameters are constant. These zones are used for radial discretization and can be supplied to the simulator. Continuous dependence of the parameters can be realized by piecewise constant approximation.

The Danckwerts boundary conditions at the column in- and outlet, Eq. (2.4) and (2.5), are modified to account for the radial coordinate:

$$u(\rho) c_{\mathrm{in},i}(t, \rho) = u(\rho) c_i^l(t, 0, \rho) - D_{\mathrm{ax},i}(\rho) \frac{\partial c_i^l}{\partial z}(t, 0, \rho) \qquad \forall t > 0, \rho \in (0, R), \tag{2.20a}$$

$$\frac{\partial c_i^l}{\partial z}(t, L, \rho) = 0 \qquad \forall t > 0, \rho \in (0, R). \tag{2.20b}$$

Conditions for the radial direction are added:

$$\frac{\partial c_i^l}{\partial \rho}(\cdot, \cdot, 0) = 0, \tag{2.21a}$$

$$\frac{\partial c_i^l}{\partial \rho}(\cdot, \cdot, R) = 0. \tag{2.21b}$$

While the inner condition Eq.(2.21a) represents symmetry at the column center, the outer condition Eq. (2.21b) is a no-flux condition.

Using the inlet boundary condition Eq. (2.20a), each radial zone is equipped with its own inlet and outlet port. That is, this unit operation has as many inlet and outlet ports as it has radial zones (parameter `NRAD` in the `discretization` group). This allows each radial zone to have its own inlet profile, which enables modeling of flow distribution in the frits by sending the feed through varying hold-up volumes before injecting it into a radial zone.

See Table 3.5.2.

Specification of flow rate / velocity and direction Since the column radius R and the zones (ρ_k, ρ_{k+1}), $k = 0, \dots, N_{\text{rad}} - 1$, are known, the interstitial velocities u_k are inferred from the volumetric flow rates via

$$u_k = u_{\text{int},k} = \frac{F_{\text{in},k}}{\pi \left(\rho_{k+1}^2 - \rho_k^2\right) \varepsilon_{c,k}},$$

where $F_{\text{in},k}$ denotes the volumetric flow rate into zone k.

The direction of flow inside the radial zone of the unit operation is governed by the sign of the interstitial velocity u_k. A positive sign results in (standard) forward flow, whereas a negative sign reverses the flow direction. Note that in case of reversed flow, the chromatogram is returned at the unit operation's *inlet* port, which may not be returned from simulation by default.

Note that, contrary to the standard general rate model as presented in Section 2.2.3, the interstitial flow rate is always given by the volumetric flow rate. The velocity parameter only determines the flow direction.

2.2.7 Continuous stirred tank reactor model (CSTR)

The continuous stirred tank reactor model is a basic building block in unit operation networks and often used to model holdup volume. When combined with a binding model, it can be used to model batch uptake experiments.

Assuming that the fluid inside the tank is well-mixed and that the volume can vary, the governing equations are given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\left[c_i + \frac{1-\varepsilon}{\varepsilon}\sum_j d_j \sum_{m_{j,i}} c^s_{j,i,m_{j,i}}\right] V\right) = F_{\text{in}} c_{\text{in},i} - F_{\text{out}} c_i + V f^l_{\text{react},i}\left(c\right) + V \frac{1-\varepsilon}{\varepsilon} \sum_j d_j f^s_{\text{react},j,i}\left(c, c^s_j\right),$$

which balances the mass, the binding equation

$$\begin{aligned} \text{quasi-stationary:} \quad & 0 = f_{\text{ads},j}\left(c, c^s_j\right), \\ \text{dynamic:} \quad & \frac{\partial c^s_j}{\partial t} = f_{\text{ads},j}\left(c, c^s_j\right) + f^s_{\text{react},j}\left(c, c^s_j\right), \end{aligned}$$

depending on whether quasi-stationary or dynamic binding is used, and the evolution of volume

$$\frac{\mathrm{d}V}{\mathrm{d}t} = F_{\text{in}} - F_{\text{out}} - F_{\text{filter}}.$$

The porosity ε denotes the ratio of liquid phase volume to total tank volume. Thus, setting $\varepsilon = 1$, removing all bound states by setting $N_{\text{bnd},j,i} = 0$ for all components i and particle types j, and applying no binding model results in a simple tank. The additional parameter F_{filter}, which denotes the flow rate of pure liquid (without any components) out of the tank, can be used to model a filtering unit.

Note that it is the user's duty to make sure that the volume of the CSTR does not fall below $0\,\mathrm{m}^3$. If it does, the simulation may fail to run or may produce unreasonable (e.g., unphysical) results.

See Table 3.5.2.

2.3 Binding models

The following binding models are presented in dynamic binding mode. By replacing all occurrences of $\mathrm{d}q/\mathrm{d}t$ with 0, quasi-stationary (rapid-equlibrium) binding mode is achieved. In quasi-stationary binding it is assumed that ad- and desorption take place on a much faster time scale than the other transport processes such that bead liquid phase $c_{p,i}$ (or bulk liquid phase c_i for certain unit operation models) are always in equilibrium with the solid phase q_i.

Equilibrium constants For the quasi-stationary binding mode, adsorption and desorption rate are no longer separate entities. Instead, the quotient $k_{\mathrm{eq}} = k_a / k_d$ of adsorption and desorption coefficient is the relevant parameter as shown for the linear binding model (see Section 2.3.1):

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} - k_{d,i} q_i \qquad \Rightarrow 0 = k_{a,i} c_{p,i} - k_{d,i} q_i \qquad \Leftrightarrow q_i = \frac{k_{a,i}}{k_{d,i}} c_{p,i} = k_{\mathrm{eq},i} c_{p,i}.$$

The equilibrium constant $k_{\mathrm{eq},i}$ is used in CADET by setting $k_{d,i} = 1$ and $k_{a,i} = k_{\mathrm{eq},i}$.

Correlation of ad- and desorption rates Note that adsorption rate $k_{a,i}$ and desorption rate $k_{d,i}$ are linearly correlated in both binding modes due to the form of the equilibrium constant k_{eq}:

$$k_{a,i} = k_{\mathrm{eq}} k_{d,i}.$$

This correlation can potentially degrade performance of some optimization algorithms. While in quasi-stationary binding mode this is prevented by using the technique above, a dynamic binding model has to be reparameterized in order to decouple parameters:

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} - k_{d,i} q_i = k_{d,i} \left[k_{\mathrm{eq},i} c_{p,i} - q_i \right] = k_{a,i} \left[c_{p,i} - \frac{1}{k_{\mathrm{eq},i}} q_i \right].$$

This can be achieved by a (nonlinear) parameter transform

$$F\left(k_{\mathrm{eq},i}, k_{d,i}\right) = \begin{pmatrix} k_{\mathrm{eq},i} k_{d,i} \\ k_{d,i} \end{pmatrix} \text{ with Jacobian } J_F\left(k_{\mathrm{eq},i}, k_{d,i}\right) = \begin{pmatrix} k_{d,i} & k_{\mathrm{eq},i} \\ 0 & 1 \end{pmatrix}.$$

Dependence on external function A binding model may depend on an external function or profile $T\colon\ [0, T_{\mathrm{end}}] \times [0, L] \to \mathbb{R}$, where L denotes the physical length of the unit operation, or $T\colon\ [0, T_{\mathrm{end}}] \to \mathbb{R}$ if the unit operation model has no axial length. By using an external profile, it is possible to account for effects that are not directly modeled in CADET (e.g., temperature). The dependence of each parameter is modeled by a polynomial of third degree. For example, the adsorption rate k_a is really given by

$$k_a(T) = k_{a,3} T^3 + k_{a,2} T^2 + k_{a,1} T + k_{a,0}.$$

While $k_{a,0}$ is set by the original parameter `XXX_KA` of the file format (`XXX` being a placeholder for the binding model), the parameters $k_{a,3}$, $k_{a,2}$, and $k_{a,1}$ are given by `XXX_KA_TTT`, `XXX_KA_TT`, and `XXX_KA_T`, respectively. The identifier of the externally dependent binding model is constructed from the original identifier by prepending `EXT_` (e.g., `MULTI_COMPONENT_LANGMUIR` is changed into `EXT_MULTI_COMPONENT_LANGMUIR`). This pattern applies to all parameters and supporting binding models (see Table 2.3.1). Note that the parameter units have to be adapted to the unit of the external profile by dividing with an appropriate power.

Each parameter of the externally dependent binding model can depend on a different external source. The 0-based indices of the external source for each parameter is given in the dataset `EXTFUN`. By assigning only one index to `EXTFUN`, all parameters use the same source. The ordering of the parameters in `EXTFUN` is given by the ordering in the file format specification in Section 3.5.5.

Binding model feature matrix A short comparison of the most prominent binding model features is given in Table 2.3.1. The implemented binding models can be divided into two main classes: Single-state and multi-state binding. While single-state models only have one bound state per component (or less), multi-state models provide multiple (possibly different) bound states for each component, which may correspond to different binding orientations or binding site types. The models also differ in whether a mobile phase modifier (e.g., salt) is supported to modulate the binding behavior.

Binding model	Competi-tive	Mobile phase modifier	External function	Multi-state
Linear	✗	✗	✓	✗
Multi component Langmuir	✓	✗	✓	✗
Multi component Anti-Langmuir	✓	✗	✓	✗
Steric mass action	✓	✓	✓	✗
Self association	✓	✓	✓	✗
Mobile phase modulator Langmuir	✓	✓	✓	✗
Extended Mobile phase modulator Langmuir	✓	✓	✓	✗
Kumar-Langmuir	✓	✓	✓	✗
Saska	✗	✗	✓	✗
Multi component Bi-Langmuir	✓	✗	✓	✓
Multi component spreading	✓	✗	✓	✓
Multi-state steric mass action	✓	✓	✓	✓
Simplified multi-state steric mass action	✓	✓	✗	✓
Bi steric mass action	✓	✓	✓	✓

Table 2.3.1: *Supported features of the different binding models*

Reference concentrations Some binding models use reference concentrations c_{ref} and q_{ref} of the mobile phase modulator (e.g., salt) in the particle liquid and solid phase, respectively. The reference values are mainly used for normalizing adsorption and desorption rates, but also for other parameters that appear with those concentrations. They amount to a simple parameter transformation that is exemplified at one equation of the steric mass action binding model

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} \bar{q}_0^{\nu_i} - k_{d,i} q_i c_{p,0}^{\nu_i},$$

where $c_{p,0}$ denotes the mobile phase salt concentration and

$$\bar{q}_0 = \Lambda - \sum_{j=1}^{N_{\mathrm{comp}}-1} \left(\nu_j + \sigma_j\right) q_j$$

is the number of available binding sites which is related to the number of bound salt ions. Using the parameter transformation

$$\begin{aligned} k_{a,i} &= \tilde{k}_{a,i} q_{\mathrm{ref}}^{-\nu_i}, \\ k_{d,i} &= \tilde{k}_{d,i} c_{\mathrm{ref}}^{-\nu_i}, \end{aligned}$$

we obtain the modified model equation

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = \tilde{k}_{a,i} c_{p,i} \left(\frac{\bar{q}_0}{q_{\mathrm{ref}}}\right)^{\nu_i} - \tilde{k}_{d,i} q_i \left(\frac{c_{p,0}}{c_{\mathrm{ref}}}\right)^{\nu_i}.$$

This transformation serves as a (partial) nondimensionalization of the adsorption and desorption rates and, by properly choosing the reference concentrations c_{ref} and q_{ref}, may improve the optimizer performance.

Recommended choices for c_{ref} are the average or maximum inlet concentration of the mobile phase modifier c_0, and for q_{ref} the ionic capacity Λ. Note that setting the reference concentrations to 1.0 each results in the original binding model.

2.3.1 Linear

A linear binding model, which is often employed for low concentrations or in analytic settings [Gui+06].

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} - k_{d,i} q_i \qquad\qquad i = 0, \dots, N_{\text{comp}} - 1.$$

See Table 3.5.5.

2.3.2 Multi Component Langmuir

The Langmuir binding model includes a saturation term and takes into account the capacity of the resin [Lan16; Gui+06]. All components compete for the same binding sites.

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i}\, c_{p,i}\, q_{\text{max},i} \left(1 - \sum_{j=0}^{N_{\text{comp}}-1} \frac{q_j}{q_{\text{max},j}} \right) - k_{d,i} q_i \qquad\qquad i = 0, \dots, N_{\text{comp}} - 1.$$

See Table 3.5.5.

2.3.3 Multi Component Anti-Langmuir

The Anti-Langmuir (or generalized Langmuir) binding model extends the Langmuir model (see Section 2.3.2). The factor $p_j \in \{-1, 1\}$ determines the shape of the isotherm. For $p_j = 1$ (standard Langmuir) the chromatograms have sharp fronts and a dispersed tail (isotherm is concave). In case of the Anti-Langmuir ($p_j = -1$) it is the other way around (isotherm is convex).

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} q_{\text{max},i} \left(1 - \sum_{j=0}^{N_{\text{comp}}-1} p_j \frac{q_j}{q_{\text{max},j}} \right) - k_{d,i} q_i \qquad\qquad i = 0, \dots, N_{\text{comp}} - 1.$$

See Table 3.5.5.

2.3.4 Steric Mass Action

The steric mass action model takes charges of the molecules into account [BC92] and is, thus, often used in ion-exchange chromatography. Each component has a characteristic charge ν that determines the number of available binding sites Λ (ionic capacity) used up by a molecule. Due to the molecule's shape, some additional binding sites (steric shielding factor σ) may be shielded from other molecules and are not available for binding.

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} c_{p,i} \left(\frac{\bar{q}_0}{q_{\text{ref}}} \right)^{\frac{\nu_i}{\nu_0}} - k_{d,i}\, q_i \left(\frac{c_{p,0}}{c_{\text{ref}}} \right)^{\frac{\nu_i}{\nu_0}} \qquad\qquad i = 1, \dots, N_{\text{comp}} - 1,$$

where $c_{p,0}$ and q_0 denote the salt concentrations in the liquid and solid phase of the beads, respectively. The number of free binding sites

$$\bar{q}_0 = \Lambda - \sum_{j=1}^{N_{\text{comp}}-1} \left(\nu_j + \sigma_j \right) q_j = \nu_0 q_0 - \sum_{j=1}^{N_{\text{comp}}-1} \sigma_j q_j$$

is calculated from the number of bound counter ions q_0 by taking steric shielding into account. In turn, the number of bound counter ions q_0 (electro-neutrality condition) is given by

$$\nu_0 q_0 = \Lambda - \sum_{j=1}^{N_{\text{comp}}-1} \nu_j q_j,$$

which also compensates for the missing equation for $\frac{\mathrm{d}q_0}{\mathrm{d}t}$. See Table 3.5.5.

The concept of reference concentrations (c_{ref} and q_{ref}) is explained in the respective paragraph in Section 2.3.

2.3.5 Self Association

This binding model is similar to the steric mass action model (see Section 2.3.4) but is also capable of describing dimerization [Mol08; Wes+12]. The dimerization, which is the immobilization of protein at some already bound protein, is also termed "self-association". It is modeled by adding a quadratic (in $c_{p,i}$) term to the adsorption part of the equation.

$$\begin{aligned}
\frac{\mathrm{d}q_i}{\mathrm{d}t} &= c_{p,i} \left(\frac{\bar{q}_0}{q_{\mathrm{ref}}}\right)^{\frac{\nu_i}{\nu_0}} \left[k_{a,i,1} + k_{a,i,2} c_{p,i}\right] - k_{d,i}\, q_i \left(\frac{c_{p,0}}{c_{\mathrm{ref}}}\right)^{\frac{\nu_i}{\nu_0}} && i = 1, \dots, N_{\mathrm{comp}} - 1, \\
\nu_0 q_0 &= \Lambda - \sum_{j=1}^{N_{\mathrm{comp}}-1} \nu_j q_j,
\end{aligned}$$

where the number of available binding sites is given by

$$\bar{q}_0 = \Lambda - \sum_{j=1}^{N_{\mathrm{comp}}-1} \left(\nu_j + \sigma_j\right) q_j = \nu_0 q_0 - \sum_{j=1}^{N_{\mathrm{comp}}-1} \sigma_j q_j.$$

See Table 3.5.5.

The concept of reference concentrations (c_{ref} and q_{ref}) is explained in the respective paragraph in Section 2.3.

2.3.6 Mobile Phase Modulator Langmuir

This model is a modified Langmuir model (see Section 2.3.2) which can be used to describe hydrophobic interaction chromatography [MEH89; Kar+04]. A modulator component (termed "salt", $c_{p,0}$ and q_0) influences ad- and desorption processes:

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} e^{\gamma_i c_{p,0}} c_{p,i}\, q_{\mathrm{max},i} \left(1 - \sum_{j=1}^{N_{\mathrm{comp}}-1} \frac{q_j}{q_{\mathrm{max},j}}\right) - k_{d,i}\, c_{p,0}^{\beta_i}\, q_i \qquad i = 1, \dots, N_{\mathrm{comp}} - 1.$$

where $c_{p,0}$ and q_0 denote the salt concentrations in the liquid and solid phase of the beads respectively. Salt is considered to be inert, therefore either

$$\frac{\mathrm{d}q_0}{\mathrm{d}t} = 0$$

is used if salt has one bound state, or salt can be used without a bound state. The parameter γ describes the hydrophobicity and β the ion-exchange characteristics. See Table 3.5.5.

2.3.7 Extended Mobile Phase Modulator Langmuir

This model is an extension of the mobile phase modulator Langmuir model (see Section 2.3.6), which allows linear binding of some selected components. A modifier component $c_{p,\mathrm{mod}}$ is selected and the remaining components are divided into the index sets $\mathcal{I}_{\mathrm{lin}}$ and $\mathcal{I}_{\mathrm{lang}}$.

$$\begin{aligned}
\frac{\mathrm{d}q_i}{\mathrm{d}t} &= k_{a,i} e^{\gamma_i c_{p,\mathrm{mod}}} c_{p,i}\, q_{\mathrm{max},i} \left(1 - \sum_{j=1}^{N_{\mathrm{comp}}-1} \frac{q_j}{q_{\mathrm{max},j}}\right) - k_{d,i}\, c_{p,\mathrm{mod}}^{\beta_i}\, q_i && i \in \mathcal{I}_{\mathrm{lang}}, \\
\frac{\mathrm{d}q_i}{\mathrm{d}t} &= k_{a,i} c_{p,i} - k_{d,i}\, q_i && i \in \mathcal{I}_{\mathrm{lin}}.
\end{aligned}$$

The modifier component is considered to be inert, therefore either

$$\frac{\mathrm{d}q_{\mathrm{mod}}}{\mathrm{d}t} = 0$$

is used if the modifier component has a bound state, or it can be used without a bound state.

The model can also be used without a modifier component. In this case, the equations are given by

$$\begin{aligned}
\frac{\mathrm{d}q_i}{\mathrm{d}t} &= k_{a,i}c_{p,i}\, q_{\mathrm{max},i} \left(1 - \sum_{j=1}^{N_{\mathrm{comp}}-1} \frac{q_j}{q_{\mathrm{max},j}} \right) - k_{d,i}\, q_i && i \in \mathcal{I}_{\mathrm{lang}}, \\
\frac{\mathrm{d}q_i}{\mathrm{d}t} &= k_{a,i}c_{p,i} - k_{d,i}\, q_i && i \in \mathcal{I}_{\mathrm{lin}}.
\end{aligned}$$

See Table 3.5.5.

2.3.8 Kumar-Langmuir

This extension of the Langmuir isotherm (see Section 2.3.2) developed in [Kum+15] was used to model charge variants of monoclonal antibodies in ion-exchange chromatography. A non-binding salt component $c_{p,0}$ is added to modulate the ad- and desorption process.

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = k_{a,i} \exp\left(\frac{k_{\mathrm{act},i}}{T} \right) c_{p,i} q_{\mathrm{max},i} \left(1 - \sum_{j=1}^{N_{\mathrm{comp}}-1} \frac{q_j}{q_{\mathrm{max},j}} \right) - k_{d,i} \left(c_{p,0} \right)^{\nu_i} q_i \qquad i = 1, \dots, N_{\mathrm{comp}} - 1$$

In this model, the true adsorption rate $k_{a,i,\mathrm{true}}$ is governed by the Arrhenius law in order to take temperature into account

$$k_{a,i,\mathrm{true}} = k_{a,i} \exp\left(\frac{k_{\mathrm{act},i}}{T} \right).$$

Here, $k_{a,i}$ is the frequency or pre-exponential factor, $k_{\mathrm{act},i} = E/R$ is the activation temperature (E denotes the activation energy and R the Boltzmann gas constant), and T is the temperature. The characteristic charge ν of the protein is taken into account by the power law. See Table 3.5.5.

2.3.9 Saska

In this binding model an additional quadratic term is added to the linear model [Sas+92]. The quadratic term allows to take interactions of liquid phase components into account.

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = H_i c_{p,i} + \sum_{j=0}^{N_{\mathrm{comp}}-1} k_{ij} c_{p,i} c_{p,j} - q_i \qquad i = 0, \dots, N_{\mathrm{comp}} - 1$$

See Table 3.5.5.

2.3.10 Multi Component Bi-Langmuir

The multi component Bi-Langmuir model [Gui+06] adds $M - 1$ *additional* types of binding sites $q_{i,j}$ ($0 \leq j \leq M - 1$) to the Langmuir model (see Section 2.3.2) without allowing an exchange between the different sites $q_{i,j}$ and $q_{i,k}$ ($k \neq j$). Therefore, there are no competitivity effects between the different types of binding sites and they have independent capacities.

$$\frac{\mathrm{d}q_{i,j}}{\mathrm{d}t} = k_{a,i}^{(j)}\, c_{p,i}\, q_{\mathrm{max},i}^{(j)} \left(1 - \sum_{k=0}^{N_{\mathrm{comp}}-1} \frac{q_{k,j}}{q_{\mathrm{max},k}^{(j)}} \right) - k_{d,i}^{(j)} q_{i,j} \qquad i = 0, \dots, N_{\mathrm{comp}} - 1,\ j = 0, \dots, M - 1.$$

Note that all binding components must have exactly the same number of binding site types $M \geq 1$. See the Langmuir isotherm in Section 2.3.2 and Table 3.5.5.

Originally, the Bi-Langmuir model is limited to two different binding site types. Here, the model has been extended to arbitrary many binding site types.

2.3.11 Multi Component Spreading

The multi component spreading model adds a second bound state $q_{i,2}$ to the Langmuir model (see Section 2.3.2) and allows the exchange between the two bound states $q_{i,1}$ and $q_{i,2}$. In the spreading model a second state of the bound molecule (e.g., a different orientation on the surface or a different folding state) is added. The exchange of molecules between the two states is allowed and, since the molecules can potentially bind in both states at the same binding site, competitivity effects are present. This is different to the Bi-Langmuir model in which another type of binding sites is added and no exchange between the different bound states is considered (see Section 2.3.10). For all components $i = 0, \dots, N_{\text{comp}} - 1$ the equations are given by

$$\begin{aligned}
\frac{\mathrm{d}q_{i,1}}{\mathrm{d}t} &= \left(k_a^A\, c_{p,i} - k_{12} q_{i,1}\right) q_{\max,i}^A \left(1 - \sum_{j=0}^{N_{\text{comp}}-1} \frac{q_j^A}{q_{\max,j}^A} - \sum_{j=0}^{N_{\text{comp}}-1} \frac{q_j^B}{q_{\max,j}^B}\right) - k_d^A q_{i,1} + k_{21} q_{i,2}, \\
\frac{\mathrm{d}q_{i,2}}{\mathrm{d}t} &= \left(k_a^B\, c_{p,i} + k_{12} q_{i,1}\right) q_{\max,i}^A \left(1 - \sum_{j=0}^{N_{\text{comp}}-1} \frac{q_j^A}{q_{\max,j}^A} - \sum_{j=0}^{N_{\text{comp}}-1} \frac{q_j^B}{q_{\max,j}^B}\right) - \left(k_d^B + k_{21}\right) q_{i,2}.
\end{aligned}$$

See Table 3.5.5.

2.3.12 Multi-State Steric Mass Action

The multi-state steric mass action model adds $M_i - 1$ *additional* bound states $q_{i,j}$ $(j = 0, \dots, M_i - 1)$ for each component i to the steric mass action model (see Section 2.3.4) and allows the exchange between the different bound states $q_{i,0}, \dots, q_{i,M-1}$ of each component. In the multi-state SMA model a variable number of states of the bound molecule (e.g., different orientations on the surface, binding strength of tentacle adsorbers) is added which are more and more strongly bound, i.e.,

$$\nu_{i,j} \leq \nu_{i,j+1} \qquad i = 1, \dots, N_{\text{comp}} - 1, \quad j = 0, \dots, M_i - 1.$$

The exchange between the different states of each component is allowed and, since the molecules can potentially bind in all states at the same binding site, competitive effects are present.

$$\begin{aligned}
\frac{\mathrm{d}q_{i,j}}{\mathrm{d}t} = {} & k_{a,i}^{(j)} c_{p,i} \left(\frac{\bar{q}_0}{q_{\text{ref}}}\right)^{\frac{\nu_{i,j}}{\nu_0}} - k_{d,i}^{(j)}\, q_{i,j} \left(\frac{c_{p,0}}{c_{\text{ref}}}\right)^{\frac{\nu_{i,j}}{\nu_0}} \\
& - \underbrace{\sum_{\ell=0}^{j-1} k_{j\ell}^{(i)}\, q_{i,j} \left(\frac{c_{p,0}}{c_{\text{ref}}}\right)^{\frac{\nu_{i,j}-\nu_{i,\ell}}{\nu_0}}}_{\text{to weak state}} - \underbrace{\sum_{\ell=j+1}^{M_i-1} k_{j\ell}^{(i)}\, q_{i,j} \left(\frac{\bar{q}_0}{q_{\text{ref}}}\right)^{\frac{\nu_{i,\ell}-\nu_{i,j}}{\nu_0}}}_{\text{to strong state}} \\
& + \underbrace{\sum_{\ell=0}^{j-1} k_{\ell j}^{(i)}\, q_{i,\ell} \left(\frac{\bar{q}_0}{q_{\text{ref}}}\right)^{\frac{\nu_{i,j}-\nu_{i,\ell}}{\nu_0}}}_{\text{from weak state}} + \underbrace{\sum_{\ell=j+1}^{M_i-1} k_{\ell j}^{(i)}\, q_{i,\ell} \left(\frac{c_{p,0}}{c_{\text{ref}}}\right)^{\frac{\nu_{i,\ell}-\nu_{i,j}}{\nu_0}}}_{\text{from strong state}} \qquad \begin{aligned} & i = 1, \dots, N_{\text{comp}} - 1, \\ & j = 0, \dots, M_i - 1, \end{aligned}
\end{aligned}$$

where $c_{p,0}$ and q_0 denote the salt concentrations in the liquid and solid phase of the beads respectively. The number of available salt ions $\bar{q}_0$ is given by

$$\bar{q}_0 = \Lambda - \sum_{j=1}^{N_{\text{comp}}-1} \sum_{\ell=0}^{M_j-1} \left(\nu_{j,\ell} + \sigma_{j,\ell}\right) q_{j,\ell}.$$

A neutrality condition compensating for the missing equation for $\frac{\mathrm{d}q_0}{\mathrm{d}t}$ is required:

$$\nu_0 q_0 = \Lambda - \sum_{j=1}^{N_{\text{comp}}-1} \sum_{\ell=0}^{M_j-1} \nu_{j,\ell} q_{j,\ell}.$$

See Table 3.5.5.

The concept of reference concentrations (c_{ref} and q_{ref}) is explained in the respective paragraph in Section 2.3.

2.3.13 Simplified Multi-State Steric Mass Action

The simplified multi-state steric mass action is the same as the multi-state SMA model described above (see Section 2.3.12), but with additional assumptions:

- Molecules are only exchanged between two adjacent states, that is, no transfer from state $q_{i,1}$ to state $q_{i,3}$ is allowed.
- Characteristic charge $\nu_{i,j}$ and shielding factor $\sigma_{i,j}$ only depend on the index of the state j.

Thus, the exchange parameters $k_{j\ell}^{(i)}$, the characteristic charge $\nu_{i,j}$, and the shielding $\sigma_{i,j}$ can be parameterized with few degrees of freedom. For all $i = 1, \dots, N_{\text{comp}} - 1$ and $j, \ell = 0, \dots, M_i - 1$ let

$$\begin{aligned}
k_{j\ell}^{(i)} &= \begin{cases} 0, & \text{for } |j-\ell| \neq 1 \\ K_{ws}^{(i)} + jK_{ws,\text{lin}}^{(i)} - K_{ws,\text{quad}}^{(i)} j(j - M_i + 2), & \text{for } \ell = j+1 \\ K_{sw}^{(i)} + \ell K_{sw,\text{lin}}^{(i)} - K_{sw,\text{quad}}^{(i)} \ell(\ell - M_i + 2), & \text{for } \ell = j-1, \end{cases} \\
\nu_{i,j} &= \nu_{\text{min},i} + \frac{j}{M_i - 1}\left(\nu_{\text{max},i} - \nu_{\text{min},i}\right) - \nu_{\text{quad},i} j(j - M_i + 1), \\
\sigma_{i,j} &= \sigma_{\text{min},i} + \frac{j}{M_i - 1}\left(\sigma_{\text{max},i} - \sigma_{\text{min},i}\right) - \sigma_{\text{quad},i} j(j - M_i + 1).
\end{aligned}$$

Note that the characteristic charge $\nu_{i,j}$ has to be monotonically non-decreasing in the second index j and all other rates and the steric factor $\sigma_{i,j}$ have to be non-negative. See Table 3.5.5.

2.3.14 Bi Steric Mass Action

Similar to the Bi-Langmuir model (see Section 2.3.10), the Bi-SMA model adds $M-1$ *additional* types of binding sites $q_{i,j}$ $(0 \leq j \leq M-1)$ to the SMA model (see Section 2.3.4) without allowing an exchange between the different sites $q_{i,j}$ and $q_{i,k}$ $(k \neq j)$. Therefore, there are no competitivity effects between the two types of binding sites and they have independent capacities.

$$\frac{\mathrm{d}q_{i,j}}{\mathrm{d}t} = k_{a,i,j} c_{p,i} \left(\frac{\bar{q}_{0,j}}{q_{\text{ref},j}}\right)^{\frac{\nu_{i,j}}{\nu_{0,j}}} - k_{d,i,j}\, q_{i,j} \left(\frac{c_{p,0}}{c_{\text{ref},j}}\right)^{\frac{\nu_{i,j}}{\nu_{0,j}}} \qquad i = 1, \dots, N_{\text{comp}} - 1, \quad j = 0, \dots, M-1,$$

where $c_{p,0}$ and $q_{0,j}$ $(0 \leq j \leq M-1)$ denote the salt concentrations in the liquid and solid phases of the beads respectively. The number of available salt ions $\bar{q}_{0,j}$ for each binding site type $0 \leq j \leq M-1$ is given by

$$\bar{q}_{0,j} = \Lambda_j - \sum_{k=1}^{N_{\text{comp}}-1} \left(\nu_{k,j} + \sigma_{k,j}\right) q_{k,j}.$$

Electro-neutrality conditions compensating for the missing equations for $\frac{\mathrm{d}q_{0,j}}{\mathrm{d}t}$ are required:

$$\nu_{0,j} q_{0,j} = \Lambda_j - \sum_{k=1}^{N_{\text{comp}}-1} \nu_{k,j} q_{k,j} \qquad j = 0, \dots, M-1.$$

Note that all binding components must have exactly the same number of binding site types $M \geq 1$. See Table 3.5.5.

The reference concentrations $c_{\mathrm{ref},j}$ and $q_{\mathrm{ref},j}$ can be specified for each binding site type $0 \leq j \leq M-1$. The concept of reference concentrations is explained in the respective paragraph in Section 2.3.

Originally, the Bi-SMA model is limited to two different binding site types. Here, the model has been extended to arbitrary many binding site types.

2.3.15 Generalized Ion Exchange

The generalized ion exchange model is based on the steric mass action model [Huu+17; Mol08]. In addition to the first component $c_{p,0}$, which represents salt, the second component $c_{p,1}$ represents another non-binding modifier (e.g., pH). In comparison to the SMA model, the characteristic charge ν and the adsorption and desorption rate constants are modified:

$$\begin{aligned}
\nu_0 q_0 &= \Lambda - \sum_{j=2}^{N_{\mathrm{comp}}-1} \nu_j(c_{p,1}) q_j \\
\frac{\partial q_i}{\partial t} &= k_{a,i}(c_{p,0}, c_{p,1})\, c_{p,i}\, \frac{\left(\bar{q}_0\right)^{\frac{\nu_i(c_{p,1})}{\nu_0}}}{\left(q_{\mathrm{ref}}\right)^{\frac{\nu_{i,\mathrm{base}}}{\nu_0}}} - k_{d,i}(c_{p,0}, c_{p,1})\, q_i\, \frac{\left(c_{p,0}\right)^{\frac{\nu_i(c_{p,1})}{\nu_0}}}{\left(c_{\mathrm{ref}}\right)^{\frac{\nu_{i,\mathrm{base}}}{\nu_0}}} \qquad i = 2, \dots, N_{\mathrm{comp}} - 1,
\end{aligned}$$

where

$$\bar{q}_0 = \Lambda - \sum_{j=2}^{N_{\mathrm{comp}}-1} \left(\nu_j(c_{p,1}) + \sigma_j\right) q_j = \nu_0 q_0 - \sum_{j=2}^{N_{\mathrm{comp}}-1} \sigma_j q_j$$

The dependence of the parameters on $c_{p,0}$ and $c_{p,1}$ is given for $i = 2, \dots, N_{\mathrm{comp}} - 1$ by

$$\begin{aligned}
\nu_i(c_{p,1}) &= \nu_{i,\mathrm{base}} + c_{p,1}\nu_{i,\mathrm{lin}} + c_{p,1}^2 \nu_{i,\mathrm{quad}} \\
k_{a,i}\left(c_{p,0}, c_{p,1}\right) &= k_{a,i,\mathrm{base}} \exp\left(k_{a,i,\mathrm{lin}} c_{p,1} + k_{a,i,\mathrm{quad}} c_{p,1}^2 + k_{a,i,\mathrm{salt}} \frac{c_{p,0}}{c_{\mathrm{ref}}} + k_{a,i,\mathrm{prot}} c_{p,i}\right) \\
k_{d,i}\left(c_{p,0}, c_{p,1}\right) &= k_{d,i,\mathrm{base}} \exp\left(k_{d,i,\mathrm{lin}} c_{p,1} + k_{d,i,\mathrm{quad}} c_{p,1}^2 + k_{d,i,\mathrm{salt}} \frac{c_{p,0}}{c_{\mathrm{ref}}} + k_{d,i,\mathrm{prot}} c_{p,i}\right)
\end{aligned}$$

See Table 3.5.5.

The concept of reference concentrations (c_{ref} and q_{ref}) is explained in the respective paragraph in Section 2.3.

2.4 Reaction models

Reaction models describe the (net) fluxes f_{react} of a reaction mechanism. The most common mechanism is the mass action law.

Correlation of forward- and backward rate constants Note that forward rate constant $k_{\mathrm{fwd},i}$ and backward rate constant $k_{\mathrm{bwd},i}$ of reaction i are linearly correlated due to the form of the equilibrium constant $k_{\mathrm{eq},i}$:

$$k_{\mathrm{fwd},i} = k_{\mathrm{eq},i} k_{\mathrm{bwd},i}.$$

This correlation can potentially degrade performance of some optimization algorithms. The parameters can be decoupled by reparameterization:

$$r_{\mathrm{net},i} = k_{\mathrm{fwd},i} f_{\mathrm{fwd},i} - k_{\mathrm{bwd},i} f_{\mathrm{bwd},i} = k_{\mathrm{bwd},i} \left[k_{\mathrm{eq},i} f_{\mathrm{fwd},i} - f_{\mathrm{bwd},i} \right] = k_{\mathrm{fwd},i} \left[f_{\mathrm{fwd},i} - \frac{1}{k_{\mathrm{eq},i}} f_{\mathrm{bwd},i} \right].$$

This can be achieved by a (nonlinear) parameter transform

$$F\left(k_{\mathrm{eq},i}, k_{\mathrm{bwd},i}\right) = \begin{pmatrix} k_{\mathrm{eq},i} k_{\mathrm{bwd},i} \\ k_{\mathrm{bwd},i} \end{pmatrix} \text{ with Jacobian } J_F\left(k_{\mathrm{eq},i}, k_{\mathrm{bwd},i}\right) = \begin{pmatrix} k_{\mathrm{bwd},i} & k_{\mathrm{eq},i} \\ 0 & 1 \end{pmatrix}.$$

Dependence on external function A reaction model may depend on an external function or profile $T\colon [0, T_{\mathrm{end}}] \times [0, L] \to \mathbb{R}$, where L denotes the physical length of the unit operation, or $T\colon [0, T_{\mathrm{end}}] \to \mathbb{R}$ if the unit operation model has no axial length. By using an external profile, it is possible to account for effects that are not directly modeled in CADET (e.g., temperature). The dependence of each parameter is modeled by a polynomial of third degree. For example, the forward rate constant k_{fwd} is really given by

$$k_{\mathrm{fwd}}(T) = k_{\mathrm{fwd},3} T^3 + k_{\mathrm{fwd},2} T^2 + k_{\mathrm{fwd},1} T + k_{\mathrm{fwd},0}.$$

While $k_{\mathrm{fwd},0}$ is set by the original parameter `XXX_KFWD` of the file format (`XXX` being a placeholder for the reaction model), the parameters $k_{\mathrm{fwd},3}$, $k_{\mathrm{fwd},2}$, and $k_{\mathrm{fwd},1}$ are given by `XXX_KFWD_TTT`, `XXX_KFWD_TT`, and `XXX_KFWD_T`, respectively. The identifier of the externally dependent reaction model is constructed from the original identifier by prepending `EXT_` (e.g., `MASS_ACTION_LAW` is changed into `EXT_MASS_ACTION_LAW`). This pattern applies to all parameters and supporting reaction models. Note that the parameter units have to be adapted to the unit of the external profile by dividing with an appropriate power.

Each parameter of the externally dependent reaction model can depend on a different external source. The 0-based indices of the external source for each parameter is given in the dataset `EXTFUN`. By assigning only one index to `EXTFUN`, all parameters use the same source. The ordering of the parameters in `EXTFUN` is given by the ordering in the file format specification in Section 3.5.6.

2.4.1 Mass action law

The mass action law reaction model is suitable for most reactions. Note that the concentrations are directly used for calculating the fluxes. Hence, the model only holds for dilute solutions under the assumption of a well-stirred reaction vessel. These assumptions can be weakened by passing to the generalized mass action law, which uses chemical activities instead of concentrations.

The mass action law states that the speed of a reaction is proportional to the product of the concentrations of their reactants. The net flux for component i is given by

$$f^l_{\mathrm{react},i}\left(c^l\right) = \sum_{j=0}^{N_{\mathrm{react}}-1} s^l_{i,j} \varphi^l_j\left(c^l\right), \qquad \varphi^l_j(c^l) = k^l_{\mathrm{fwd},j} \prod_{\ell=0}^{N_{\mathrm{comp}}-1} \left(c^l_\ell\right)^{e^l_{\mathrm{fwd},\ell,j}} - k^l_{\mathrm{bwd},j} \prod_{\ell=0}^{N_{\mathrm{comp}}-1} \left(c^l_\ell\right)^{e^l_{\mathrm{bwd},\ell,j}},$$

where $S^l = (s^l_{i,j}) \in \mathbb{R}^{N_{\mathrm{comp}} \times N_{\mathrm{react}}}$ is the stoichiometric matrix, $\varphi^l_j(c)$ is the net flux of reaction j, and $k^l_{\mathrm{fwd},j}$ and $k^l_{\mathrm{bwd},j}$ are the rate constants. The matrices $E^l_{\mathrm{fwd}} = (e^l_{\mathrm{fwd},\ell,j}) \in \mathbb{R}^{N_{\mathrm{comp}} \times N_{\mathrm{react}}}$ and

$E^l_{\text{bwd}} = (e^l_{\text{bwd},\ell,j}) \in \mathbb{R}^{N_{\text{comp}} \times N_{\text{react}}}$ are usually derived by the order of the reaction, that is,

$$e^l_{\text{fwd},\ell,j} = \max(0, -s^l_{\ell,j}), \qquad e^l_{\text{bwd},\ell,j} = \max(0, s^l_{\ell,j}). \tag{2.22}$$

However, these defaults can be changed by providing those matrices.

In situations where both liquid and solid phase are present (e.g., in a bead), the respective other phase may act as a modifier in the net flux equation. For example, consider reactions in the liquid phase of a particle given by

$$f^p_{\text{react},i}\left(c^p, c^s\right) = \sum_{j=0}^{N_{\text{react}}-1} s^p_{i,j} \varphi^p_j\left(c^p, c^s\right),$$

where

$$\begin{aligned}\varphi^p_j(c^p, c^s) = k^p_{\text{fwd},j} &\left[\prod_{\ell=0}^{N_{\text{comp}}-1} \left(c^p_\ell\right)^{e^p_{\text{fwd},\ell,j}}\right] \left[\prod_{m=0}^{\sum_{i=0}^{N_{\text{comp}}-1} N_{\text{bnd},i}-1} \left(c^s_m\right)^{e^{ps}_{\text{fwd},m,j}}\right] \\ - k^p_{\text{bwd},j} &\left[\prod_{\ell=0}^{N_{\text{comp}}-1} \left(c^p_\ell\right)^{e^p_{\text{bwd},\ell,j}}\right] \left[\prod_{m=0}^{\sum_{i=0}^{N_{\text{comp}}-1} N_{\text{bnd},i}-1} \left(c^s_m\right)^{e^{ps}_{\text{bwd},m,j}}\right].\end{aligned}$$

The forward and backward rates of the liquid phase particle reactions can be modified by a power of every bound state in the solid phase of the particle. The exponents of these powers are given by the matrices $E^{ps}_{\text{fwd}} = (e^{ps}_{\text{fwd},m,j})$ and $E^{ps}_{\text{bwd}} = (e^{ps}_{\text{bwd},m,j})$, which are both of size $(\sum_i N_{\text{bnd},i}) \times N_{\text{react}}$. Whereas the exponent matrices $E^p_{\text{fwd}}, E^p_{\text{bwd}} \in \mathbb{R}^{N_{\text{comp}} \times N_{\text{react}}}$ are initialized based on the stoichiometric matrix $S^p \in \mathbb{R}^{N_{\text{comp}} \times N_{\text{react}}}$, see Eq. (2.22), the exponent matrices $E^{ps}_{\text{fwd}}, E^{ps}_{\text{bwd}}$ of the modifier terms default to 0.

Vice versa, the rates of solid phase reactions can be modified by liquid phase concentrations. The corresponding exponent matrices $E^{sp}_{\text{fwd}} = (e^{sp}_{\text{fwd},\ell,j})$ and $E^{sp}_{\text{bwd}} = (e^{sp}_{\text{bwd},\ell,j})$ are both of size $N_{\text{comp}} \times N_{\text{react}}$.

$$f^s_{\text{react},i}\left(c^s, c^p\right) = \sum_{j=0}^{N_{\text{react}}-1} s^s_{i,j} \varphi^s_j\left(c^s, c^p\right),$$

where

$$\begin{aligned}\varphi^s_j(c^s, c^p) = k^s_{\text{fwd},j} &\left[\prod_{m=0}^{\sum_{i=0}^{N_{\text{comp}}-1} N_{\text{bnd},i}-1} \left(c^s_m\right)^{e^s_{\text{fwd},m,j}}\right] \left[\prod_{\ell=0}^{N_{\text{comp}}-1} \left(c^p_\ell\right)^{e^{sp}_{\text{fwd},\ell,j}}\right] \\ - k^p_{\text{bwd},j} &\left[\prod_{m=0}^{\sum_{i=0}^{N_{\text{comp}}-1} N_{\text{bnd},i}-1} \left(c^s_m\right)^{e^s_{\text{bwd},m,j}}\right] \left[\prod_{\ell=0}^{N_{\text{comp}}-1} \left(c^p_\ell\right)^{e^{sp}_{\text{bwd},\ell,j}}\right].\end{aligned}$$

Whereas the exponent matrices $E^s_{\text{fwd}}, E^s_{\text{bwd}} \in \mathbb{R}^{(\sum_i N_{\text{bnd},i}) \times N_{\text{react}}}$ are initialized based on the stoichiometric matrix $S^s \in \mathbb{R}^{(\sum_i N_{\text{bnd},i}) \times N_{\text{react}}}$, see Eq. (2.22), the exponent matrices $E^{sp}_{\text{fwd}}, E^{sp}_{\text{bwd}}$ of the modifier terms default to 0.

The layout of the matrices in the file format is presented in Table 3.5.6.

3 CADET File Format Specifications

The CADET framework is designed to work on a file format structured into groups and datasets. This concept may be implemented by different file formats. At the moment, CADET natively supports HDF5 and XML as file formats. The choice is not limited to those two formats but can be extended as needed. In this section the general layout and structure of the file format is described.

File format versions The file format may change and evolve over time as new features are added to the simulator. This manual describes the most recent file format version that is also set as default value in `/meta/FILE_FORMAT` (see Tab. 3.7). The simulator assumes that the input file uses the most recent format version and does not update old files to the current standard.

3.1 Global structure

The global structure (see Fig. 3.1.1) is divided into three parts: `input`, `output`, and `meta`. Every valid CADET file needs an `input` group (see Fig. 3.1.2) which contains all relevant information for simulating a model. It does not need an `output` (see Fig. 3.1.4) or `meta` (see Fig. 3.1.1) group, since those are created when results are written. Whereas the `output` group is solely used as output and holds the results of the simulation, the `meta` group is used for input and output. Details such as file format version and simulator version are read from and written to the `meta` group.

If not explicitly stated otherwise, all datasets are mandatory. By convention all group names are lowercase, whereas all dataset names are uppercase. Note that this is just a description of the file format and not a detailed explanation of the meaning of the parameters. For the latter, please refer to the corresponding sections in the previous chapter.

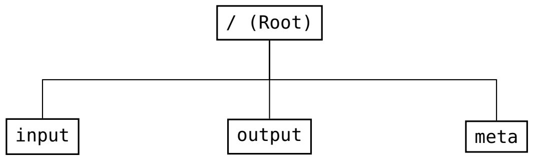


Figure 3.1.1: *Structure of the groups in the root group of the file format*

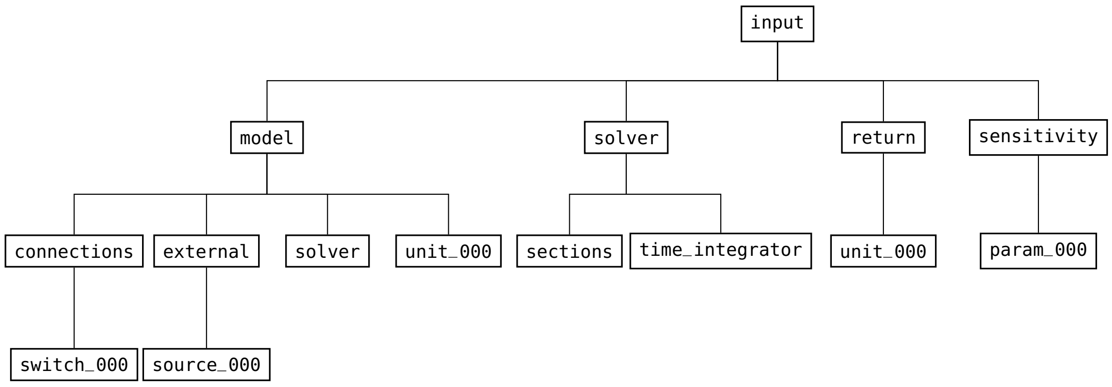


Figure 3.1.2: *High-level structure of the groups in the input part of the file format*

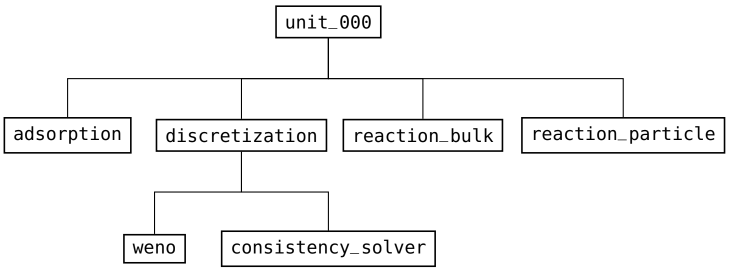


Figure 3.1.3: *Structure of the groups in a column unit operation (`/input/model` group)*

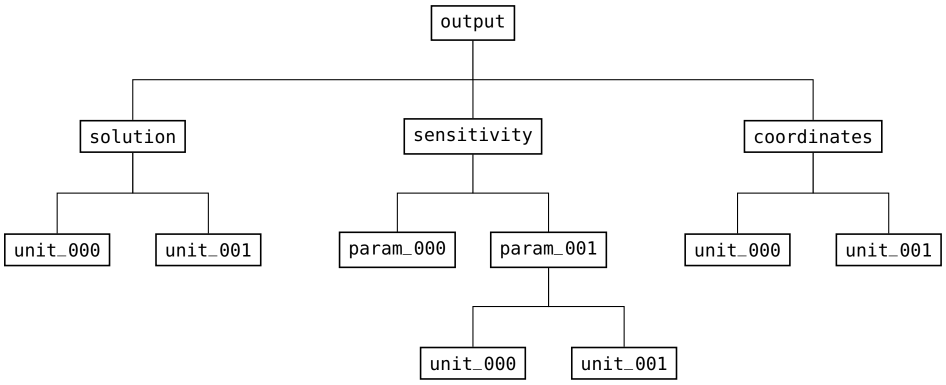


Figure 3.1.4: *Structure of the groups in the output part of the file format*

3.2 Notation and identifiers

Reference volumes are denoted by subscripts:

m^3_{IV}	Interstitial volume
m^3_{MP}	Bead mobile phase volume
m^3_{SP}	Bead solid phase volume

Common notation and identifiers that are used in the subsequent description are listed in Table 3.2.1.

Identifier	Meaning
`NCOMP`	Number of components of a unit operation
`NTOTALCOMP`	Total number of components in the system (sum of all unit operation components)
`NPARTYPE`	Number of particle types of a unit operation
$\texttt{NBND}_{\texttt{i}}$	Number of bound states of component i of the current particle type
`NTOTALBND`	Total number of bound states of the current particle type (sum of all bound states of all components)
`NSTATES`	Maximum of the number of bound states for each component of a particle type
`NREACT`	Number of reactions (in bulk volume or in the current particle type)
`NDOF`	Total number of degrees of freedom of the current unit operation model or system of unit operations
`NSEC`	Number of time integration sections
`PARAM_VALUE`	Value of a generic unspecified parameter

Table 3.2.1: *Common notation and identifiers used in the file format description*

3.3 Ordering of multi dimensional data

Some model parameters, especially in certain binding models, require multi dimensional data. Since CADET only reads one dimensional arrays, the layout of the data has to be specified (i.e., the way how the data is linearized in memory). The term "*xyz*-major" means that the index corresponding to *xyz* changes the slowest.

For instance, suppose a model with 2 components and 3 bound states has a "component-major" dataset. Then, the requested matrix is stored in memory such that all bound states are listed for each component (i.e., the component index changes the slowest and the bound state index the fastest):

```
comp0bnd0, comp0bnd1, comp0bnd2, comp1bnd0, comp1bnd1, comp1bnd2.
```

This linear array can be represented as a 2×3 matrix in "row-major" storage format, or a 3×2 matrix in "column-major" ordering.

3.4 Section dependent model parameters

Some model parameters (see Table 3.4.1) can be assigned different values for each section. For example, the velocity a column is operated with could differ in the load, wash, and elution phases. Section dependency is recognized by specifying the appropriate number of values for the parameters (see *Length* column in the following tables). If a parameter depends on both the component and the section, the ordering is section-major.

For instance, the *Length* field of the parameter `VELOCITY` reads "1 / `NSEC`" which means that it is not recognized as section dependent if only 1 value (scalar) is passed. However, if `NSEC` many values (vector) are present, it will be treated as section dependent.

Note that all components of component dependent datasets have to be section dependent (e.g., you cannot have a section dependency on component 2 only while the other components are not section dependent).

Dataset	Component dependent	Section dependent
`COL_DISPERSION`	✓	✓
`FILM_DIFFUSION`	✓	✓
`PAR_DIFFUSION`	✓	✓
`PAR_SURFDIFFUSION`	✓	✓
`VELOCITY`		✓

Table 3.4.1: *Section dependent datasets in the 1D unit operation models (`/input/model/unit_XXX` group)*

3.5 Input group

3.5.1 System of unit operations

Group `/input/model`

`NUNITS` Number of unit operations in the system

Type: int **Range:** ≥ 1 **Length:** 1

`INIT_STATE_Y` Initial full state vector (optional, unit operation specific initial data is ignored)

Type: int **Range:** ≥ 1 **Length:** 1

`INIT_STATE_YDOT` Initial full time derivative state vector (optional, unit operation specific initial data is ignored)

Type: double **Length:** `NDOF`

`INIT_STATE_SENSY_XXX` Number of unit operations in the system

Type: double **Length:** `NDOF`

`INIT_STATE_SENSYDOT_XXX` Initial full state vector of the `XXX`th sensitivity system (optional, unit operation specific initial data is ignored)

Type: double **Length:** `NDOF`

NUNITS Initial full time derivative state vector of the XXXth sensitivity system (optional, unit operation specific initial data is ignored)

Type: double **Length:** NDOF

Group /input/model/connections

NSWITCHES Number of valve switches

Type: int **Range:** ≥ 1 **Length:** 1

CONNECTIONS_INCLUDE_PORTS Determines whether the CONNECTIONS table includes ports (1) or not (0). Optional, defaults to 0 unless a unit operation model with multiple ports is present.

Type: int **Range:** $\{0, 1\}$ **Length:** 1

CONNECTIONS_INCLUDE_DYNAMIC_FLOW Determines whether the CONNECTIONS table includes linear, quadratic, and cubic flow rate coefficients (1) or not (0). Optional, defaults to 0.

Type: int **Range:** $\{0, 1\}$ **Length:** 1

Group /input/model/connections/switch_XXX

SECTION Index of the section that activates this connection set

Type: int **Range:** ≥ 0 **Length:** 1

CONNECTIONS Matrix with list of connections in row-major storage. Columns are *UnitOpID from*, *UnitOpID to*, *Port from*, *Port to*, *Component from*, *Component to*, *volumetric flow rate*, *linear flow rate coefficient*, *quadratic flow rate coefficient*, *cubic flow rate coefficient*. If both port indices are -1, all ports are connected. If both component indices are -1, all components are connected.

The flow rate is a cubic function of time,

$$Q = Q_0 + Q_1(t - t_s) + Q_2(t - t_s)^2 + Q_3(t - t_s)^3,$$

where t_s is the beginning of the section that activates the switch (i.e., SECTION_TIMES at index SECTION).

The port indices are left out if CONNECTIONS_INCLUDE_PORTS is set to 0 and no unit operation with multiple ports is present in the system. If a unit operation with multiple ports is present, CONNECTIONS_INCLUDE_PORTS is ignored and port indices are mandatory.

The last three flow rate coefficients are left out if CONNECTIONS_INCLUDE_DYNAMIC_FLOW is set to 0. Contrary to the constant coefficient, which has the parameter name CONNECTION, the other coefficients are named CONNECTION_LIN, CONNECTION_QUAD, and CONNECTION_CUB, respectively.

For addressing the flow rates as a parameter senstivity, the mapping is as follows:

SENS_UNIT Unused, always set to -1

SENS_BOUNDPHASE *UnitOpID from*

SENS_REACTION *UnitOpID to*

SENS_COMP *Port from*

SENS_PARTYPE *Port to*

SENS_SECTION SECTION that activates the valve switch

Type: double **Range:** ≥ -1 **Length:** $\{5, 7, 8, 10\} \cdot$ NCONNECTIONS

Group /input/model/external/source_XXX – EXTFUN_TYPE = LINEAR_INTERP_DATA

VELOCITY Velocity of the external profile in positive column axial direction

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** 1

DATA Function values T at the data points

Unit: [T] **Type:** double **Range:** $\mathbb{R}$ **Length:** Arbitrary

TIME Time of the data points

Unit: s **Type:** double **Range:** ≥ 0.0 **Length:** Same as DATA

Group /input/model/external/source_XXX – EXTFUN_TYPE = PIECEWISE_CUBIC_POLY

VELOCITY Velocity of the external profile in positive column axial direction

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** 1

CONST_COEFF Constant coefficients of piecewise cubic polynomial

Unit: [T] **Type:** double **Range:** $\mathbb{R}$ **Length:** Arbitrary

LIN_COEFF Linear coefficients of piecewise cubic polynomial

Unit: $[T]\,s^{-1}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** Same as CONST_COEFF

CONST_COEFF Quadratic coefficients of piecewise cubic polynomial

Unit: $[T]\,s^{-2}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** Same as CONST_COEFF

QUAD_COEFF Cubic coefficients of piecewise cubic polynomial

Unit: $[T]\,s^{-3}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** Same as CONST_COEFF

SECTION_TIMES Simulation times at which a new piece begins (breaks of the piecewise polynomial)

Unit: s **Type:** double **Range:** ≥ 0.0 **Length:** CONST_COEFF+1

Group /input/model/solver

GS_TYPE Type of Gram-Schmidt orthogonalization, see IDAS guide Section 4.5.7.3, p. 41f. A value of 0 enables classical Gram-Schmidt, a value of 1 uses modified Gram-Schmidt.

Type: int **Range:** $\{0, 1\}$ **Length:** 1

MAX_KRYLOV Defines the size of the Krylov subspace in the iterative linear GMRES solver (0: MAX_KRYLOV = NDOF)

Type: int **Range:** $\{0, \dots, \texttt{NDOF}\}$ **Length:** 1

MAX_RESTARTS Maximum number of restarts in the GMRES algorithm. If lack of memory is not an issue, better use a larger Krylov space than restarts.

Type: int **Range:** ≥ 0 **Length:** 1

SCHUR_SAFETY Schur safety factor; Influences the tradeoff between linear iterations and nonlinear error control; see IDAS guide Section 2.1 and 5.

Type: double **Range:** ≥ 0 **Length:** 1

LINEAR_SOLUTION_MODE Determines whether the system of models is solved in parallel (1) or sequentially (2). A sequential solution is only possible for systems without cyclic connections. The setting can be chosen automatically (0) based on a heuristic (less than 6 unit operations and acyclic network selects sequential mode). Optional, defaults to automatic (0).

Type: int **Range:** $\{0, 1, 2\}$ **Length:** 1

3.5.2 Unit operation models

Inlet

Group /input/model/unit_XXX – UNIT_TYPE = INLET

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** `INLET` **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

INLET_TYPE Specifies the type of inlet profile

Type: string **Range:** `PIECEWISE_CUBIC_POLY` **Length:** 1

Group /input/model/unit_XXX/sec_XXX

CONST_COEFF Constant coefficients for inlet concentrations

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{IV}}^{-3}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** `NCOMP`

LIN_COEFF Linear coefficients for inlet concentrations

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{IV}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** `NCOMP`

QUAD_COEFF Quadratic coefficients for inlet concentrations

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{IV}}^{-3}\,\mathrm{s}^{-2}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** `NCOMP`

CUBE_COEFF Cubic coefficients for inlet concentrations

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{IV}}^{-3}\,\mathrm{s}^{-3}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** `NCOMP`

Outlet

Group /input/model/unit_XXX – UNIT_TYPE = OUTLET

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** `OUTLET` **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

General rate model

Group /input/model/unit_XXX – UNIT_TYPE = GENERAL_RATE_MODEL

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** `GENERAL_RATE_MODEL` **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

ADSORPTION_MODEL Specifies the type of binding model of each particle type (or of all particle types if length is 1)

Type: string **Range:** See Section 3.5.5 **Length:** 1 / `NPARTYPE`

ADSORPTION_MODEL_MULTIPLEX Multiplexing mode of `ADSORPTION_MODEL`. If set to 0, each particle type has a different binding model and the length of `ADSORPTION_MODEL` is `NPARTYPE`. If set to 1, all particle types share the same binding model and the length of `ADSORPTION_MODEL` is 1.

This field is optional and inferred from the length of `ADSORPTION_MODEL` if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

REACTION_MODEL_BULK Specifies the type of reaction model of the bulk volume. The model is configured in the subgroup `reaction_bulk`.

Type: string **Range:** See Section 3.5.6 **Length:** 1

REACTION_MODEL_PARTICLES Specifies the type of reaction model of each particle type (or of all particle types if length is 1). The model is configured in the subgroup `reaction_particle`, or `reaction_particle_XXX` in case of disabled multiplexing.

Type: string **Range:** See Section 3.5.6 **Length:** 1 / NPARTYPE

REACTION_MODEL_PARTICLES_MULTIPLEX Multiplexing mode of REACTION_MODEL_PARTICLES. If set to 0, each particle type has a different reaction model and the length of REACTION_MODEL_PARTICLES is NPARTYPE. If set to 1, all particle types share the same reaction model and the length of REACTION_MODEL_PARTICLES is 1.

This field is optional and inferred from the length of REACTION_MODEL_PARTICLES if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

INIT_C Initial concentrations for each component in the bulk mobile phase

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

INIT_CP Initial concentrations for each component in the bead liquid phase (optional, INIT_C is used if left out). The length of this field can be NCOMP (same values for each particle type) or NPARTYPE · NCOMP

Values for each particle type can only be given when ADSORPTION_MODEL_MULTIPLEX is 0. The ordering is type-major.

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP / NPARTYPE · NCOMP

INIT_Q Initial concentrations for each bound state of each component in the bead solid phase. If ADSORPTION_MODEL_MULTIPLEX is 0, values for each particle type are required in type-component-major ordering (length is NTOTALBND). If ADSORPTION_MODEL_MULTIPLEX is 1, values for one particle type are required in component-major ordering (length is $\sum_{i=0}^{\mathtt{NCOMP}-1} \mathtt{NBND}_i$).

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND / $\sum_{i=0}^{\mathtt{NCOMP}-1} \mathtt{NBND}_i$

INIT_STATE Full state vector for initialization (optional, INIT_C, INIT_CP, and INIT_Q will be ignored; if length is 2NDOF, then the second half is used for time derivatives)

Unit: various **Type:** double **Range:** $\mathbb{R}$ **Length:** NDOF / 2NDOF

COL_DISPERSION Axial dispersion coefficient

Unit: $\mathrm{m_{IV}^{2}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see COL_DISPERSION_MULTIPLEX

COL_DISPERSION_MULTIPLEX Multiplexing mode of COL_DISPERSION. Determines whether COL_DISPERSION is treated as component- and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of COL_DISPERSION.

Valid modes are:

0 Component-independent, section-independent; length of COL_DISPERSION is 1

1 Component-dependent, section-independent; length of COL_DISPERSION is NCOMP

2 Component-independent, section-dependent; length of COL_DISPERSION is NSEC

3 Component-dependent, section-dependent; length of COL_DISPERSION is NCOMP · NSEC; ordering is section-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

COL_LENGTH Column length

Unit: m **Type:** double **Range:** > 0 **Length:** 1

COL_POROSITY Column porosity

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** 1

FILM_DIFFUSION Film diffusion coefficients for each component of each particle type

Unit: $\mathrm{m\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see FILM_DIFFUSION_MULTIPLEX

FILM_DIFFUSION_MULTIPLEX Multiplexing mode of FILM_DIFFUSION. Determines whether FILM_DIFFUSION is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of FILM_DIFFUSION.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of FILM_DIFFUSION is NCOMP

1 Component-dependent, type-independent, section-dependent; length of FILM_DIFFUSION is NCOMP · NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of FILM_DIFFUSION is NCOMP · NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of FILM_DIFFUSION is NCOMP · NPARTYPE · NSEC; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_POROSITY Particle porosity of all particle types or for each particle type

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** 1 / NPARTYPE

PAR_RADIUS Particle radius of all particle types or for each particle type

Unit: m **Type:** double **Range:** > 0 **Length:** 1 / NPARTYPE

PAR_CORERADIUS Particle core radius of all particle types or for each particle type (optional, defaults to $0\,\mathrm{m}$)

Unit: m **Type:** double **Range:** $[0,$ PAR_RADIUS$)$ **Length:** 1 / NPARTYPE

PORE_ACCESSIBILITY Pore accessibility factor of each component in each particle type (optional, defaults to 1)

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** see PORE_ACCESSIBILITY_MULTIPLEX

PORE_ACCESSIBILITY_MULTIPLEX Multiplexing mode of PORE_ACCESSIBILITY. Determines whether PORE_ACCESSIBILITY is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PORE_ACCESSIBILITY.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of PORE_ACCESSIBILITY is NCOMP

1 Component-dependent, type-independent, section-dependent; length of PORE_ACCESSIBILITY is NCOMP · NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of PORE_ACCESSIBILITY is NCOMP · NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of PORE_ACCESSIBILITY is NCOMP · NPARTYPE · NSEC; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_DIFFUSION Effective particle diffusion coefficients of each component in each particle type

Unit: $\mathrm{m_{MP}^2\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see PAR_DIFFUSION_MULTIPLEX

PAR_DIFFUSION_MULTIPLEX Multiplexing mode of PAR_DIFFUSION. Determines whether PAR_DIFFUSION is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PAR_DIFFUSION.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of `PAR_DIFFUSION` is `NCOMP`

1 Component-dependent, type-independent, section-dependent; length of `PAR_DIFFUSION` is `NCOMP` · `NSEC`; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of `PAR_DIFFUSION` is `NCOMP` · `NPARTYPE`; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of `PAR_DIFFUSION` is `NCOMP` · `NPARTYPE` · `NSEC`; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

`PAR_SURFDIFFUSION` Particle surface diffusion coefficients of each bound state of each component in each particle type (optional, defaults to all $0\,\mathrm{m}_{\mathrm{SP}}^2\,\mathrm{s}^{-1}$)

Unit: $\mathrm{m}_{\mathrm{SP}}^2\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** see `PAR_SURFDIFFUSION_MULTIPL`

`PAR_SURFDIFFUSION_MULTIPLEX` Multiplexing mode of `PAR_SURFDIFFUSION`. Determines whether `PAR_SURFDIFFUSION` is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of `PAR_SURFDIFFUSION`.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of `PAR_SURFDIFFUSION` is `NBND`; ordering is component-major

1 Component-dependent, type-independent, section-dependent; length of `PAR_SURFDIFFUSION` is `NBND` · `NSEC`; ordering is section-component-major

2 Component-dependent, type-dependent, section-independent; length of `PAR_SURFDIFFUSION` is `NTOTALBND`; ordering is type-component-major

3 Component-dependent, type-dependent, section-dependent; length of `PAR_SURFDIFFUSION` is `NTOTALBND` · `NSEC`; ordering is section-type-component-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

`PAR_SURFDIFFUSION_DEP` Parameter dependence of `PAR_SURFDIFFUSION`. Valid dependencies are:

`NONE` Original parameter is used unmodified.

`LIQUID`$_{S}ALT_{E}XPONENTIAL$ Original parameter is modified by exponential law of liquid phase salt concentration.

`LIQUID`$_{S}ALT_{P}OWER$ Original parameter is modified by power law of liquid phase salt concentration.

`LIQUID`$_{S}ALT_{C}OLLOIDAL_{A}FFINITY$ Original parameter is modified by colloidal binding affinity based on liquid phase salt concentration.

Optional: If left out, no parameter dependence is assumed and the original surface diffusion coefficients are used unmodified.

Unit: – **Type:** string **Length:** 1 / `NPARTYPE`

`PAR_SURFDIFFUSION_EXPFACTOR` Factor p_1 in exponential law particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} p_{1,i,m} \exp(p_{2,i,m} c_0^p),$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient. Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_EXPONENTIAL`.

Unit: – **Type:** double **Range:** ≥ 0 **Length:** `NBND`

`PAR_SURFDIFFUSION_EXPARGMULT` Factor p_2 in exponential law particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} p_{1,i,m} \exp(p_{2,i,m} c_0^p),$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient. Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_EXPONENTIAL`.

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** `NBND`

`PAR_SURFDIFFUSION_POWFACTOR` Factor p_1 in power law particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}},$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient. Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_POWER`.

Unit: – **Type:** double **Range:** ≥ 0 **Length:** `NBND`

`PAR_SURFDIFFUSION_POWEXP` Factor p_2 in power law particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}},$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient. Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_POWER`.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** `NBND`

`PAR_SURFDIFFUSION_LOGKEQFACTOR` Factor p_1 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m}\left(c_0^p\right)\right)^{p_{5,i,m}} + p_{6,i,m} \exp\left(p_{7,i,m} k_{i,m}\left(c_0^p\right)\right)\right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_COLLOIDAL_AFFINITY`.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** `NBND`

`PAR_SURFDIFFUSION_LOGKEQEXP` Factor p_2 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m}\left(c_0^p\right)\right)^{p_{5,i,m}} + p_{6,i,m} \exp\left(p_{7,i,m} k_{i,m}\left(c_0^p\right)\right)\right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_COLLOIDAL_AFFINITY`.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** `NBND`

`PAR_SURFDIFFUSION_LOGKEQCONST` Factor p_3 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m}\left(c_0^p\right)\right)^{p_{5,i,m}} + p_{6,i,m} \exp\left(p_{7,i,m} k_{i,m}\left(c_0^p\right)\right)\right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if `PAR_SURFDIFFUSION_DEP` is `LIQUID_SALT_COLLOIDAL_AFFINITY`.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** `NBND`

`PAR_SURFDIFFUSION_POWFACTOR` Factor p_4 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m}\left(c_0^p\right)\right)^{p_{5,i,m}} + p_{6,i,m} \exp\left(p_{7,i,m} k_{i,m}\left(c_0^p\right)\right)\right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p\right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if PAR_SURFDIFFUSION_DEP is LIQUID_SALT_COLLOIDAL_AFFINITY.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** NBND

PAR_SURFDIFFUSION_POWEXP Factor p_5 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m} \left(c_0^p \right) \right)^{p_{5,i,m}} + p_{6,i,m} \exp \left(p_{7,i,m} k_{i,m} \left(c_0^p \right) \right) \right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p \right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if PAR_SURFDIFFUSION_DEP is LIQUID_SALT_COLLOIDAL_AFFINITY.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** NBND

PAR_SURFDIFFUSION_EXPFACTOR Factor p_6 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m} \left(c_0^p \right) \right)^{p_{5,i,m}} + p_{6,i,m} \exp \left(p_{7,i,m} k_{i,m} \left(c_0^p \right) \right) \right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p \right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if PAR_SURFDIFFUSION_DEP is LIQUID_SALT_COLLOIDAL_AFFINITY.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** NBND

PAR_SURFDIFFUSION_EXPARGMULT Factor p_7 in colloidal affinity particle surface diffusion relation

$$D_{s,i,m} = \tilde{D}_{s,i,m} \left[p_{4,i,m} \left(k_{i,m} \left(c_0^p \right) \right)^{p_{5,i,m}} + p_{6,i,m} \exp \left(p_{7,i,m} k_{i,m} \left(c_0^p \right) \right) \right],$$

where $\tilde{D}_{s,i,m}$ is the original surface diffusion coefficient and

$$k_{i,m}(c_0^p) = p_{1,i,m} \left(c_0^p \right)^{p_{2,i,m}} + p_{3,i,m}.$$

Only required if PAR_SURFDIFFUSION_DEP is LIQUID_SALT_COLLOIDAL_AFFINITY.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** NBND

VELOCITY Interstitial velocity of the mobile phase (optional if CROSS_SECTION_AREA is present, see Section 2.2.3)

Unit: $\mathrm{m\,s^{-1}}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** 1 / NSEC

CROSS_SECTION_AREA Cross section area of the column (optional if VELOCITY is present, see Section 2.2.3)

Unit: m^2 **Type:** double **Range:** > 0 **Length:** 1

PAR_TYPE_VOLFRAC Volume fractions of the particle types. The volume fractions can be set for all axial cells together or for each individual axial cell. For each cell, the volume fractions have to sum to 1. In case of a spatially inhomogeneous setting, the data is expected in cell-major ordering and the SENS_SECTION field is used for indexing the axial cell when specifying parameter sensitivities.

This field is optional in case of only one particle type.

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** NPARTYPE / NCOL · NPARTYPE

Group /input/model/unit_XXX/discretization – UNIT_TYPE = GENERAL_RATE_MODEL

NCOL Number of axial column discretization cells

Type: int **Range:** ≥ 1 **Length:** 1

NPARTYPE Number of particle types. Optional, inferred from the length of NPAR or NBOUND if left out.

Type: int **Range:** ≥ 1 **Length:** 1

NPAR Number of particle (radial) discretization cells for each particle type

Type: int **Range:** ≥ 1 **Length:** 1 / `NPARTYPE`

NBOUND Number of bound states for each component in each particle type in type-major ordering

Type: int **Range:** ≥ 0 **Length:** `NCOMP` / `NPARTYPE` $\cdot$ `NCOMP`

PAR_GEOM Specifies the particle geometry for all or each particle type. Valid values are `SPHERE`, `CYLINDER`, and `SLAB`. Optional, defaults to `SPHERE`.

Type: string **Range:** {`SPHERE`, `CYLINDER`, `SLAB`} **Length:** 1 / `NPARTYPE`

PAR_DISC_TYPE Specifies the discretization scheme inside the particles for all or each particle type. Valid values are `EQUIDISTANT_PAR`, `EQUIVOLUME_PAR`, and `USER_DEFINED_PAR`.

Type: string **Length:** 1 / `NPARTYPE`

PAR_DISC_VECTOR Node coordinates for the cell boundaries (ignored if `PAR_DISC_TYPE` $\neq$ `USER_DEFINED_PAR`). The coordinates are relative and have to include the endpoints 0 and 1. They are later linearly mapped to the true radial range $[r_{c,j}, r_{p,j}]$. The coordinates for each particle type are appended to one long vector in type-major ordering.

Unit: – **Type:** double **Range:** $[0,1]$ **Length:** $\sum_i(\texttt{NPAR}_i + 1)$

PAR_BOUNDARY_ORDER Order of accuracy of outer particle boundary condition. Optional, defaults to 2.

Type: int **Range:** $\{1,2\}$ **Length:** 1

USE_ANALYTIC_JACOBIAN Determines whether analytically computed Jacobian matrix (faster) is used (value is 1) instead of Jacobians generated by algorithmic differentiation (slower, value is 0)

Type: int **Range:** $\{0,1\}$ **Length:** 1

RECONSTRUCTION Type of reconstruction method for fluxes

Type: string **Range:** `WENO` **Length:** 1

GS_TYPE Type of Gram-Schmidt orthogonalization, see IDAS guide Section 4.5.7.3, p. 41f. A value of 0 enables classical Gram-Schmidt, a value of 1 uses modified Gram-Schmidt.

Type: int **Range:** $\{0,1\}$ **Length:** 1

MAX_KRYLOV Defines the size of the Krylov subspace in the iterative linear GMRES solver (0: `MAX_KRYLOV` = `NCOL` $\cdot$ `NCOMP` $\cdot$ `NPARTYPE`)

Type: int **Range:** $\{0, \dots, \texttt{NCOL} \cdot \texttt{NCOMP} \cdot \texttt{NPARTYPE}\}$ **Length:** 1

MAX_RESTARTS Maximum number of restarts in the GMRES algorithm. If lack of memory is not an issue, better use a larger Krylov space than restarts.

Type: int **Range:** ≥ 0 **Length:** 1

SCHUR_SAFETY Schur safety factor; Influences the tradeoff between linear iterations and nonlinear error control; see IDAS guide Section 2.1 and 5.

Type: double **Range:** ≥ 0 **Length:** 1

OPTIMIZE_PAR_BANDWIDTH Enables (`OPTIMIZE_PAR_BANDWIDTH` = 1, default) or disables (`OPTIMIZE_PAR_BANDWIDTH` = 0) particle Jacobian bandwidth optimization. If enabled, surface diffusion parameters `PAR_SURFDIFFUSION` that are initially zero must not become non-zero during this or subsequent simulations (i.e., initially zero surface diffusion parameters have to remain zero). This optimization can reduce the bandwidth of the particle Jacobian blocks and increase performance.

This field is optional and defaults to 1 (optimization enabled).

Type: int **Range:** $\{0,1\}$ **Length:** 1

Lumped rate model with pores

Group /input/model/unit_XXX – UNIT_TYPE = LUMPED_RATE_MODEL_WITH_PORES

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** LUMPED_RATE_MODEL_WITH_PORES **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

ADSORPTION_MODEL Specifies the type of binding model of each particle type (or of all particle types if length is 1)

Type: string **Range:** See Section 3.5.5 **Length:** 1 / NPARTYPE

ADSORPTION_MODEL_MULTIPLEX Multiplexing mode of ADSORPTION_MODEL. If set to 0, each particle type has a different binding model and the length of ADSORPTION_MODEL is NPARTYPE. If set to 1, all particle types share the same binding model and the length of ADSORPTION_MODEL is 1.

This field is optional and inferred from the length of ADSORPTION_MODEL if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

REACTION_MODEL_BULK Specifies the type of reaction model of the bulk volume. The model is configured in the subgroup reaction_bulk.

Type: string **Range:** See Section 3.5.6 **Length:** 1

REACTION_MODEL_PARTICLES Specifies the type of reaction model of each particle type (or of all particle types if length is 1). The model is configured in the subgroup reaction_particle, or reaction_particle_XXX in case of disabled multiplexing.

Type: string **Range:** See Section 3.5.6 **Length:** 1 / NPARTYPE

REACTION_MODEL_PARTICLES_MULTIPLEX Multiplexing mode of REACTION_MODEL_PARTICLES. If set to 0, each particle type has a different reaction model and the length of REACTION_MODEL_PARTICLES is NPARTYPE. If set to 1, all particle types share the same reaction model and the length of REACTION_MODEL_PARTICLES is 1.

This field is optional and inferred from the length of REACTION_MODEL_PARTICLES if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

INIT_C Initial concentrations for each component in the bulk mobile phase

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

INIT_CP Initial concentrations for each component in the bead liquid phase (optional, INIT_C is used if left out). The length of this field can be NCOMP (same values for each particle type) or NPARTYPE · NCOMP

Values for each particle type can only be given when ADSORPTION_MODEL_MULTIPLEX is 0. The ordering is type-major.

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP / NPARTYPE · NCOMP

INIT_Q Initial concentrations for each bound state of each component in the bead solid phase. If ADSORPTION_MODEL_MULTIPLEX is 0, values for each particle type are required in type-component-major ordering (length is NTOTALBND). If ADSORPTION_MODEL_MULTIPLEX is 1, values for one particle type are required in component-major ordering (length is $\sum_{i=0}^{\mathtt{NCOMP}-1} \mathtt{NBND}_i$).

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND / $\sum_{i=0}^{\mathtt{NCOMP}-1} \mathtt{NBND}_i$

INIT_STATE Full state vector for initialization (optional, INIT_C, INIT_CP, and INIT_Q will be ignored; if length is 2NDOF, then the second half is used for time derivatives)

Unit: various **Type:** double **Range:** $\mathbb{R}$ **Length:** NDOF / 2NDOF

COL_DISPERSION Axial dispersion coefficient

Unit: $\mathrm{m_{IV}^{2}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see COL_DISPERSION_MULTIPLEX

COL_DISPERSION_MULTIPLEX Multiplexing mode of COL_DISPERSION. Determines whether COL_DISPERSION is treated as component- and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of COL_DISPERSION.

Valid modes are:

0 Component-independent, section-independent; length of COL_DISPERSION is 1

1 Component-dependent, section-independent; length of COL_DISPERSION is NCOMP

2 Component-independent, section-dependent; length of COL_DISPERSION is NSEC

3 Component-dependent, section-dependent; length of COL_DISPERSION is NCOMP · NSEC; ordering is section-major

Unit: –	**Type:** int	**Range:** $\{0, \dots, 3\}$	**Length:** 1

COL_LENGTH Column length

Unit: m	**Type:** double	**Range:** > 0	**Length:** 1

COL_POROSITY Column porosity

Unit: –	**Type:** double	**Range:** $(0, 1]$	**Length:** 1

FILM_DIFFUSION Film diffusion coefficients for each component of each particle type

Unit: $\mathrm{m\,s^{-1}}$	**Type:** double	**Range:** ≥ 0	**Length:** see FILM_DIFFUSION_MULTIPLEX

FILM_DIFFUSION_MULTIPLEX Multiplexing mode of FILM_DIFFUSION. Determines whether FILM_DIFFUSION is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of FILM_DIFFUSION.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of FILM_DIFFUSION is NCOMP

1 Component-dependent, type-independent, section-dependent; length of FILM_DIFFUSION is NCOMP · NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of FILM_DIFFUSION is NCOMP · NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of FILM_DIFFUSION is NCOMP · NPARTYPE · NSEC; ordering is section-type-major

Unit: –	**Type:** int	**Range:** $\{0, \dots, 3\}$	**Length:** 1

PAR_POROSITY Particle porosity of all particle types or for each particle type

Unit: –	**Type:** double	**Range:** $(0, 1]$	**Length:** 1 / NPARTYPE

PAR_RADIUS Particle radius of all particle types or for each particle type

Unit: m	**Type:** double	**Range:** > 0	**Length:** 1 / NPARTYPE

PORE_ACCESSIBILITY Pore accessibility factor of each component in each particle type (optional, defaults to 1)

Unit: –	**Type:** double	**Range:** $(0, 1]$	**Length:** see PORE_ACCESSIBILITY_MULTIPLEX

PORE_ACCESSIBILITY_MULTIPLEX Multiplexing mode of PORE_ACCESSIBILITY. Determines whether PORE_ACCESSIBILITY is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PORE_ACCESSIBILITY.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of PORE_ACCESSIBILITY is NCOMP

1 Component-dependent, type-independent, section-dependent; length of `PORE_ACCESSIBILITY` is `NCOMP` · `NSEC`; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of `PORE_ACCESSIBILITY` is `NCOMP` · `NPARTYPE`; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of `PORE_ACCESSIBILITY` is `NCOMP` · `NPARTYPE` · `NSEC`; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

`VELOCITY` Interstitial velocity of the mobile phase (optional if `CROSS_SECTION_AREA` is present, see Section 2.2.3)

Unit: $\mathrm{m\,s^{-1}}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** 1 / `NSEC`

`CROSS_SECTION_AREA` Cross section area of the column (optional if `VELOCITY` is present, see Section 2.2.3)

Unit: $\mathrm{m^2}$ **Type:** double **Range:** > 0 **Length:** 1

`PAR_TYPE_VOLFRAC` Volume fractions of the particle types. The volume fractions can be set for all axial cells together or for each individual axial cell. For each cell, the volume fractions have to sum to 1. In case of a spatially inhomogeneous setting, the data is expected in cell-major ordering and the `SENS_SECTION` field is used for indexing the axial cell when specifying parameter sensitivities.

This field is optional in case of only one particle type.

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** `NPARTYPE` / `NCOL` · `NPARTYPE`

Group /input/model/unit_XXX/discretization – UNIT_TYPE = LUMPED_RATE_MODEL_WITH_PORES

`NCOL` Number of axial column discretization cells

Type: int **Range:** ≥ 1 **Length:** 1

`NPARTYPE` Number of particle types. Optional, inferred from the length of `NBOUND` if left out.

Type: int **Range:** ≥ 1 **Length:** 1

`NBOUND` Number of bound states for each component in each particle type in type-major ordering

Type: int **Range:** ≥ 0 **Length:** `NCOMP` / `NPARTYPE` · `NCOMP`

`PAR_GEOM` Specifies the particle geometry for all or each particle type. Valid values are `SPHERE`, `CYLINDER`, and `SLAB`. Optional, defaults to `SPHERE`.

Type: string **Range:** {`SPHERE`, `CYLINDER`, `SLAB`} **Length:** 1 / `NPARTYPE`

`USE_ANALYTIC_JACOBIAN` Determines whether analytically computed Jacobian matrix (faster) is used (value is 1) instead of Jacobians generated by algorithmic differentiation (slower, value is 0)

Type: int **Range:** $\{0, 1\}$ **Length:** 1

`RECONSTRUCTION` Type of reconstruction method for fluxes

Type: string **Range:** `WENO` **Length:** 1

`GS_TYPE` Type of Gram-Schmidt orthogonalization, see IDAS guide Section 4.5.7.3, p. 41f. A value of 0 enables classical Gram-Schmidt, a value of 1 uses modified Gram-Schmidt.

Type: int **Range:** $\{0, 1\}$ **Length:** 1

`MAX_KRYLOV` Defines the size of the Krylov subspace in the iterative linear GMRES solver (0: `MAX_KRYLOV` = `NCOL` · `NCOMP` · `NPARTYPE`)

Type: int **Range:** {0, . . . , `NCOL` · `NCOMP` · `NPARTYPE`} **Length:** 1

`MAX_RESTARTS` Maximum number of restarts in the GMRES algorithm. If lack of memory is not an issue, better use a larger Krylov space than restarts.

Type: int **Range:** ≥ 0 **Length:** 1

SCHUR_SAFETY Schur safety factor; Influences the tradeoff between linear iterations and nonlinear error control; see IDAS guide Section 2.1 and 5.

Type: double **Range:** ≥ 0 **Length:** 1

Lumped rate model without pores

Group /input/model/unit_XXX – UNIT_TYPE = LUMPED_RATE_MODEL_WITHOUT_PORES

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** LUMPED_RATE_MODEL_WITHOUT_PORES **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

ADSORPTION_MODEL Specifies the type of binding model

Type: string **Range:** See Section 3.5.5 **Length:** 1

REACTION_MODEL Specifies the type of reaction model of the combined bulk and particle volume. The model is configured in the subgroup reaction.

Type: string **Range:** See Section 3.5.6 **Length:** 1

INIT_C Initial concentrations for each component in the bulk mobile phase

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

INIT_Q Initial concentrations for each bound state of each component in the bead solid phase in component-major ordering

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

INIT_STATE Full state vector for initialization (optional, INIT_C and INIT_Q will be ignored; if length is 2NDOF, then the second half is used for time derivatives)

Unit: various **Type:** double **Range:** $\mathbb{R}$ **Length:** NDOF / 2NDOF

COL_DISPERSION Axial dispersion coefficient

Unit: $\mathrm{m_{IV}^{2}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see COL_DISPERSION_MULTIPLEX

COL_DISPERSION_MULTIPLEX Multiplexing mode of COL_DISPERSION. Determines whether COL_DISPERSION is treated as component- and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of COL_DISPERSION.

Valid modes are:

0 Component-independent, section-independent; length of COL_DISPERSION is 1

1 Component-dependent, section-independent; length of COL_DISPERSION is NCOMP

2 Component-independent, section-dependent; length of COL_DISPERSION is NSEC

3 Component-dependent, section-dependent; length of COL_DISPERSION is NCOMP · NSEC; ordering is section-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

COL_LENGTH Column length

Unit: m **Type:** double **Range:** > 0 **Length:** 1

TOTAL_POROSITY Total porosity

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** 1

VELOCITY Interstitial velocity of the mobile phase (optional if CROSS_SECTION_AREA is present, see Section 2.2.3)

Unit: $\mathrm{m\,s^{-1}}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** 1 / NSEC

CROSS_SECTION_AREA Cross section area of the column (optional if VELOCITY is present, see Section 2.2.3)

Unit: m^2 **Type:** double **Range:** > 0 **Length:** 1

Group /input/model/unit_XXX/discretization – UNIT_TYPE = LUMPED_RATE_MODEL_WITHOUT_PORES

NCOL Number of axial column discretization cells

Type: int **Range:** ≥ 1 **Length:** 1

NBOUND Number of bound states for each component

Type: int **Range:** ≥ 0 **Length:** NCOMP

USE_ANALYTIC_JACOBIAN Determines whether analytically computed Jacobian matrix (faster) is used (value is 1) instead of Jacobians generated by algorithmic differentiation (slower, value is 0)

Type: int **Range:** $\{0, 1\}$ **Length:** 1

RECONSTRUCTION Type of reconstruction method for fluxes

Type: string **Range:** WENO **Length:** 1

Two dimensional general rate model

Group /input/model/unit_XXX – UNIT_TYPE = GENERAL_RATE_MODEL_2D

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** GENERAL_RATE_MODEL_2D **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

ADSORPTION_MODEL Specifies the type of binding model of each particle type (or of all particle types if length is 1)

Type: string **Range:** See Section 3.5.5 **Length:** 1 / NPARTYPE

ADSORPTION_MODEL_MULTIPLEX Multiplexing mode of ADSORPTION_MODEL. If set to 0, each particle type has a different binding model and the length of ADSORPTION_MODEL is NPARTYPE. If set to 1, all particle types share the same binding model and the length of ADSORPTION_MODEL is 1.

This field is optional and inferred from the length of ADSORPTION_MODEL if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

REACTION_MODEL_BULK Specifies the type of reaction model of the bulk volume. The model is configured in the subgroup reaction_bulk.

Type: string **Range:** See Section 3.5.6 **Length:** 1

REACTION_MODEL_PARTICLES Specifies the type of reaction model of each particle type (or of all particle types if length is 1). The model is configured in the subgroup reaction_particle, or reaction_particle_XXX in case of disabled multiplexing.

Type: string **Range:** See Section 3.5.6 **Length:** 1 / NPARTYPE

REACTION_MODEL_PARTICLES_MULTIPLEX Multiplexing mode of REACTION_MODEL_PARTICLES. If set to 0, each particle type has a different reaction model and the length of REACTION_MODEL_PARTICLES is NPARTYPE. If set to 1, all particle types share the same reaction model and the length of REACTION_MODEL_PARTICLES is 1.

This field is optional and inferred from the length of REACTION_MODEL_PARTICLES if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

INIT_C Initial concentrations for each component in all radial zones the bulk mobile phase (length NCOMP), or for each component in each radial zone (length NCOMP · NRAD, ordering radial-major)

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP / NCOMP · NRAD

INIT_CP Initial concentrations for each component in the bead liquid phase (optional, INIT_C is used if left out). The length of this field can be NCOMP (same values for each radial zone and particle type), NPARTYPE · NCOMP (same values for each radial zone), RAD · NCOMP (same values for each particle type), or NRAD · NPARTYPE · NCOMP. The ordering is radial-type-major.

Values for each particle type can only be given when ADSORPTION_MODEL_MULTIPLEX is 0. In the radial-inhomogeneous case, the SENS_REACTION field is used for indexing the radial zone when specifying parameter sensitivities.

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double **Range:** ≥ 0

INIT_Q Initial concentrations for each bound state of each component in the bead solid phase. If ADSORPTION_MODEL_MULTIPLEX is 0, values for each particle type are required in type-component-major ordering (length is NTOTALBND). If ADSORPTION_MODEL_MULTIPLEX is 1, values for one particle type are required in component-major ordering (length is $\sum_{i=0}^{\texttt{NCOMP}-1} \texttt{NBND}_i$).

Alternatively, values for each radial zone can be supplied. If ADSORPTION_MODEL_MULTIPLEX is 0, values for each radial zone and each particle type are required in radial-type-component-major ordering (length is NRAD · NTOTALBND). If ADSORPTION_MODEL_MULTIPLEX is 1, values for each radial zone and all particle types are required in radial-component-major ordering (length is NRAD · $\sum_{i=0}^{\texttt{NCOMP}-1} \texttt{NBND}_i$). In the radial-inhomogeneous case, the SENS_REACTION field is used for indexing the radial zone when specifying parameter sensitivities.

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0

INIT_STATE Full state vector for initialization (optional, INIT_C, INIT_CP, and INIT_Q will be ignored; if length is 2NDOF, then the second half is used for time derivatives)

Unit: various **Type:** double **Range:** $\mathbb{R}$ **Length:** NDOF / 2NDOF

COL_DISPERSION Axial dispersion coefficient.

In case of a spatially inhomogeneous setting, the SENS_PARTYPE field is used for indexing the radial zone when specifying parameter sensitivities.

Unit: $\mathrm{m_{IV}^{2}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** see COL_DISPERSION_MULTIPLEX

COL_DISPERSION_MULTIPLEX Multiplexing mode of COL_DISPERSION. Determines whether COL_DISPERSION is treated as component-, radial-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of COL_DISPERSION.

Valid modes are:

0 Component-independent, radial-independent, section-independent; length of COL_DISPERSION is 1

1 Component-independent, radial-dependent, section-independent; length of COL_DISPERSION is NRAD

2 Component-dependent, radial-independent, section-independent; length of COL_DISPERSION is NCOMP

3 Component-dependent, radial-dependent, section-independent; length of COL_DISPERSION is NCOMP · NRAD; ordering is radial-major

4 Component-independent, radial-independent, section-dependent; length of COL_DISPERSION is NSEC

5 Component-independent, radial-dependent, section-dependent; length of COL_DISPERSION is NRAD · NSEC; ordering is section-major

6 Component-dependent, radial-independent, section-independent; length of COL_DISPERSION is NCOMP · NSEC; ordering is section-major

7 Component-dependent, radial-dependent, section-dependent; length of `COL_DISPERSION` is `NCOMP` · `NRAD` · `NSEC`; ordering is section-radial-major

Unit: – **Type:** int **Range:** $\{0, \dots, 7\}$ **Length:** 1

`COL_DISPERSION_RADIAL` Radial dispersion coefficient.

In case of a spatially inhomogeneous setting, the `SENS_PARTYPE` field is used for indexing the radial zone when specifying parameter sensitivities.

Unit: $\mathrm{m}_{\mathrm{IV}}^{2}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** see `COL_DISPERSION_RADIAL_MUL`

`COL_DISPERSION_RADIAL_MULTIPLEX` Multiplexing mode of `COL_DISPERSION_RADIAL`. Determines whether `COL_DISPERSION_RADIAL` is treated as component-, radial-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of `COL_DISPERSION_RADIAL`.

Valid modes are:

0 Component-independent, radial-independent, section-independent; length of `COL_DISPERSION_RADIAL` is 1

1 Component-independent, radial-dependent, section-independent; length of `COL_DISPERSION_RADIAL` is `NRAD`

2 Component-dependent, radial-independent, section-independent; length of `COL_DISPERSION_RADIAL` is `NCOMP`

3 Component-dependent, radial-dependent, section-independent; length of `COL_DISPERSION_RADIAL` is `NCOMP` · `NRAD`; ordering is radial-major

4 Component-independent, radial-independent, section-dependent; length of `COL_DISPERSION_RADIAL` is `NSEC`

5 Component-independent, radial-dependent, section-dependent; length of `COL_DISPERSION_RADIAL` is `NRAD` · `NSEC`; ordering is section-major

6 Component-dependent, radial-independent, section-independent; length of `COL_DISPERSION_RADIAL` is `NCOMP` · `NSEC`; ordering is section-major

7 Component-dependent, radial-dependent, section-dependent; length of `COL_DISPERSION_RADIAL` is `NCOMP` · `NRAD` · `NSEC`; ordering is section-radial-major

Unit: – **Type:** int **Range:** $\{0, \dots, 7\}$ **Length:** 1

`COL_LENGTH` Column length

Unit: m **Type:** double **Range:** > 0 **Length:** 1

`COL_RADIUS` Column radius

Unit: m **Type:** double **Range:** > 0 **Length:** 1

`COL_POROSITY` Column porosity, either constant (length is 1) or for each radial zone (length is `NRAD`).

In case of a spatially inhomogeneous setting, the `SENS_PARTYPE` field is used for indexing the radial zone when specifying parameter sensitivities.

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** 1 / `NRAD`

`FILM_DIFFUSION` Film diffusion coefficients for each component of each particle type

Unit: $\mathrm{m}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** see `FILM_DIFFUSION_MULTIPLEX`

`FILM_DIFFUSION_MULTIPLEX` Multiplexing mode of `FILM_DIFFUSION`. Determines whether `FILM_DIFFUSION` is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of `FILM_DIFFUSION`.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of `FILM_DIFFUSION` is `NCOMP`

1 Component-dependent, type-independent, section-dependent; length of FILM_DIFFUSION is NCOMP $\cdot$ NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of FILM_DIFFUSION is NCOMP $\cdot$ NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of FILM_DIFFUSION is NCOMP $\cdot$ NPARTYPE $\cdot$ NSEC; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_POROSITY Particle porosity of all particle types or for each particle type

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** 1 / NPARTYPE

PAR_RADIUS Particle radius of all particle types or for each particle type

Unit: m **Type:** double **Range:** > 0 **Length:** 1 / NPARTYPE

PAR_CORERADIUS Particle core radius of all particle types or for each particle type (optional, defaults to 0 m)

Unit: m **Type:** double **Range:** $[0,$ PAR_RADIUS$)$ **Length:** 1 / NPARTYPE

PORE_ACCESSIBILITY Pore accessibility factor of each component in each particle type (optional, defaults to 1)

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** see PORE_ACCESSIBILITY_MULTIPLEX

PORE_ACCESSIBILITY_MULTIPLEX Multiplexing mode of PORE_ACCESSIBILITY. Determines whether PORE_ACCESSIBILITY is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PORE_ACCESSIBILITY.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of PORE_ACCESSIBILITY is NCOMP

1 Component-dependent, type-independent, section-dependent; length of PORE_ACCESSIBILITY is NCOMP $\cdot$ NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of PORE_ACCESSIBILITY is NCOMP $\cdot$ NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of PORE_ACCESSIBILITY is NCOMP $\cdot$ NPARTYPE $\cdot$ NSEC; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_DIFFUSION Effective particle diffusion coefficients of each component in each particle type

Unit: $\mathrm{m}_{\mathrm{MP}}^2\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** see PAR_DIFFUSION_MULTIPLEX

PAR_DIFFUSION_MULTIPLEX Multiplexing mode of PAR_DIFFUSION. Determines whether PAR_DIFFUSION is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PAR_DIFFUSION.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of PAR_DIFFUSION is NCOMP

1 Component-dependent, type-independent, section-dependent; length of PAR_DIFFUSION is NCOMP $\cdot$ NSEC; ordering is section-major

2 Component-dependent, type-dependent, section-independent; length of PAR_DIFFUSION is NCOMP $\cdot$ NPARTYPE; ordering is type-major

3 Component-dependent, type-dependent, section-dependent; length of PAR_DIFFUSION is NCOMP $\cdot$ NPARTYPE $\cdot$ NSEC; ordering is section-type-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_SURFDIFFUSION Particle surface diffusion coefficients of each bound state of each component in each particle type (optional, defaults to all $0\,\mathrm{m}_{\mathrm{SP}}^2\,\mathrm{s}^{-1}$)

Unit: $\mathrm{m}_{\mathrm{SP}}^2\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** see PAR_SURFDIFFUSION_MULTIPL

PAR_SURFDIFFUSION_MULTIPLEX Multiplexing mode of PAR_SURFDIFFUSION. Determines whether PAR_SURFDIFFUSION is treated as component-, type-, and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PAR_SURFDIFFUSION.

Valid modes are:

0 Component-dependent, type-independent, section-independent; length of PAR_SURFDIFFUSION is NBND; ordering is component-major

1 Component-dependent, type-independent, section-dependent; length of PAR_SURFDIFFUSION is NBND · NSEC; ordering is section-component-major

2 Component-dependent, type-dependent, section-independent; length of PAR_SURFDIFFUSION is NTOTALBND; ordering is type-component-major

3 Component-dependent, type-dependent, section-dependent; length of PAR_SURFDIFFUSION is NTOTALBND · NSEC; ordering is section-type-component-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

VELOCITY Indicates flow direction in each radial zone (forward if value is positive, backward if value is negative), see Section 2.2.6).

In case of a spatially inhomogeneous setting, the SENS_PARTYPE field is used for indexing the radial cell when specifying parameter sensitivities.

Unit: – **Type:** double **Range:** $\mathbb{R}$ **Length:** see VELOCITY_MULTIPLEX

VELOCITY_MULTIPLEX Multiplexing mode of VELOCITY. Determines whether VELOCITY is treated as radial- and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of VELOCITY.

Valid modes are:

0 Radial-independent, section-independent; length of VELOCITY is 1

1 Radial-dependent, section-independent; length of VELOCITY is NRAD

2 Section-dependent; length of VELOCITY is NSEC

3 Radial-dependent, section-dependent; length of VELOCITY is NRAD · NSEC; ordering is section-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

PAR_TYPE_VOLFRAC Volume fractions of the particle types. The volume fractions can be set homogeneous or individually along both axes. For each cell, the volume fractions have to sum to 1.

In case of a spatially inhomogeneous setting, the SENS_SECTION field is used for indexing the axial cell and the SENS_REACTION field is used for indexing the radial cell when specifying parameter sensitivities.

This field is optional in case of only one particle type.

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** see PAR_TYPE_VOLFRAC_MULTIPLE

PAR_TYPE_VOLFRAC_MULTIPLEX Multiplexing mode of PAR_TYPE_VOLFRAC. Determines whether PAR_TYPE_VOLFRAC is treated as radial- and/or section-independent.

This field is optional. When left out, multiplexing behavior is inferred from the length of PAR_TYPE_VOLFRAC.

Valid modes are:

0 Radial-independent, axial-independent; length of PAR_TYPE_VOLFRAC is NPARTYPE

1 Radial-dependent, axial-independent; length of `PAR_TYPE_VOLFRAC` is `NRAD` $\cdot$ `NPARTYPE`; ordering is radial-major

2 Axial-dependent; length of `PAR_TYPE_VOLFRAC` is `NCOL` $\cdot$ `NPARTYPE`; ordering is axial-major

3 Radial-dependent, axial-dependent; length of `PAR_TYPE_VOLFRAC` is `NCOL` $\cdot$ `NRAD` $\cdot$ `NPARTYPE`; ordering is axial-radial-major

Unit: – **Type:** int **Range:** $\{0, \dots, 3\}$ **Length:** 1

Group /input/model/unit_XXX/discretization – UNIT_TYPE = GENERAL_RATE_MODEL_2D

NCOL Number of axial column discretization cells

Type: int **Range:** ≥ 1 **Length:** 1

NRAD Number of radial column discretization cells

Type: int **Range:** ≥ 1 **Length:** 1

NPARTYPE Number of particle types. Optional, inferred from the length of `NPAR` or `NBOUND` if left out.

Type: int **Range:** ≥ 1 **Length:** 1

NPAR Number of particle (radial) discretization cells for each particle type

Type: int **Range:** ≥ 1 **Length:** 1 / `NPARTYPE`

NBOUND Number of bound states for each component in each particle type in type-major ordering

Type: int **Range:** ≥ 0 **Length:** `NCOMP` / `NPARTYPE` $\cdot$ `NCOMP`

PAR_GEOM Specifies the particle geometry for all or each particle type. Valid values are `SPHERE`, `CYLINDER`, and `SLAB`. Optional, defaults to `SPHERE`.

Type: string **Range:** {`SPHERE`, `CYLINDER`, `SLAB`} **Length:** 1 / `NPARTYPE`

RADIAL_DISC_TYPE Specifies the radial discretization scheme. Valid values are `EQUIDISTANT`, `EQUIVOLUME`, and `USER_DEFINED`.

Type: string **Length:** 1

RADIAL_COMPARTMENTS Coordinates for the radial compartment boundaries (ignored if `RADIAL_DISC_TYPE` $\neq$ `USER_DEFINED`). The coordinates are absolute and have to include the endpoints 0 and `COLUMN_RADIUS`. The values are expected in ascending order (i.e., 0 is the first and `COLUMN_RADIUS` the last value in the array).

Unit: m **Type:** double **Range:** [0, `COLUMN_RADIUS`] **Length:** `NRAD` + 1

PAR_DISC_TYPE Specifies the discretization scheme inside the particles for all or each particle type. Valid values are `EQUIDISTANT_PAR`, `EQUIVOLUME_PAR`, and `USER_DEFINED_PAR`.

Type: string **Length:** 1 / `NPARTYPE`

PAR_DISC_VECTOR Node coordinates for the cell boundaries (ignored if `PAR_DISC_TYPE` $\neq$ `USER_DEFINED_PAR`). The coordinates are relative and have to include the endpoints 0 and 1. They are later linearly mapped to the true radial range $[r_{c,j}, r_{p,j}]$. The coordinates for each particle type are appended to one long vector in type-major ordering.

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** $\sum_i(\texttt{NPAR}_i + 1)$

PAR_BOUNDARY_ORDER Order of accuracy of outer particle boundary condition. Optional, defaults to 2.

Type: int **Range:** $\{1, 2\}$ **Length:** 1

USE_ANALYTIC_JACOBIAN Determines whether analytically computed Jacobian matrix (faster) is used (value is 1) instead of Jacobians generated by algorithmic differentiation (slower, value is 0)

Type: int **Range:** $\{0, 1\}$ **Length:** 1

LINEAR_SOLVER_BULK Linear solver used for the sparse column bulk block. This field is optional, the best available method is selected (i.e., sparse direct solver if possible).

Valid values are:

DENSE Converts the sparse matrix into a banded matrix and uses regular LAPACK. Slow and memory intensive, but always available.

UMFPACK Uses the UMFPACK sparse direct solver (LU decomposition) from SuiteSparse. Fast, but has to be enabled when compiling and requires UMFPACK library.

SUPERLU Uses the SuperLU sparse direct solver (LU decomposition). Fast, but has to be enabled when compiling and requires SuperLU library.

Type: string **Range:** {DENSE, UMFPACK, SUPERLU} **Length:** 1

RECONSTRUCTION Type of reconstruction method for fluxes

Type: string **Range:** WENO **Length:** 1

GS_TYPE Type of Gram-Schmidt orthogonalization, see IDAS guide Section 4.5.7.3, p. 41f. A value of 0 enables classical Gram-Schmidt, a value of 1 uses modified Gram-Schmidt.

Type: int **Range:** $\{0, 1\}$ **Length:** 1

MAX_KRYLOV Defines the size of the Krylov subspace in the iterative linear GMRES solver (0: MAX_KRYLOV $=$ NCOL $\cdot$ NRAD $\cdot$ NCOMP $\cdot$ NPARTYPE)

Type: int **Range:** $\{0, \dots,$ NCOL $\cdot$ NRAD $\cdot$ NCOMP $\cdot$ NPARTYPE$\}$ **Length:** 1

MAX_RESTARTS Maximum number of restarts in the GMRES algorithm. If lack of memory is not an issue, better use a larger Krylov space than restarts.

Type: int **Range:** ≥ 0 **Length:** 1

SCHUR_SAFETY Schur safety factor; Influences the tradeoff between linear iterations and nonlinear error control; see IDAS guide Section 2.1 and 5.

Type: double **Range:** ≥ 0 **Length:** 1

Continuous stirred tank reactor model

Group /input/model/unit_XXX – UNIT_TYPE = CSTR

UNIT_TYPE Specifies the type of unit operation model

Type: string **Range:** CSTR **Length:** 1

NCOMP Number of chemical components in the chromatographic medium

Type: int **Range:** ≥ 1 **Length:** 1

NBOUND Number of bound states for each component in each particle type in type-major ordering (optional, defaults to all 0)

Type: int **Range:** ≥ 0 **Length:** NPARTYPE $\cdot$ NCOMP

USE_ANALYTIC_JACOBIAN Determines whether analytically computed Jacobian matrix (faster) is used (value is 1) instead of Jacobians generated by algorithmic differentiation (slower, value is 0)

Type: int **Range:** $\{0, 1\}$ **Length:** 1

ADSORPTION_MODEL Specifies the type of binding model of each particle type

Type: string **Range:** See Section 3.5.5 **Length:** NPARTYPE

REACTION_MODEL_BULK Specifies the type of reaction model of the bulk volume. The model is configured in the subgroup reaction_bulk.

Type: string **Range:** See Section 3.5.6 **Length:** 1

REACTION_MODEL_PARTICLES Specifies the type of reaction model of each particle type (or of all particle types if length is 1). The model is configured in the subgroup `reaction_particle`, or `reaction_particle_XXX` in case of disabled multiplexing.

Type: string **Range:** See Section 3.5.6 **Length:** 1 / NPARTYPE

REACTION_MODEL_PARTICLES_MULTIPLEX Multiplexing mode of REACTION_MODEL_PARTICLES. If set to 0, each particle type has a different reaction model and the length of REACTION_MODEL_PARTICLES is NPARTYPE. If set to 1, all particle types share the same reaction model and the length of REACTION_MODEL_PARTICLES is 1.

This field is optional and inferred from the length of REACTION_MODEL_PARTICLES if left out.

Unit: – **Type:** int **Range:** $\{0, 1\}$ **Length:** 1

INIT_C Initial concentrations for each component in the mobile phase

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

INIT_VOLUME Initial tank volume

Unit: $\mathrm{m^3}$ **Type:** double **Range:** ≥ 0 **Length:** 1

INIT_Q Initial concentrations for each bound state of each component in the bead solid phase of each particle type in type-component-major ordering. This field is optional and defaults to all 0.

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

INIT_STATE Full state vector for initialization (optional, INIT_C, INIT_Q, and INIT_VOLUME will be ignored; if length is 2NDOF, then the second half is used for time derivatives)

Unit: various **Type:** double **Range:** $\mathbb{R}$ **Length:** NDOF / 2NDOF

POROSITY Porosity ε (defaults to 1)

Unit: – **Type:** double **Range:** $(0, 1]$ **Length:** 1

FLOWRATE_FILTER Flow rate of pure liquid without components (optional, defaults to $0\,\mathrm{m^3\,s^{-1}}$)

Unit: $\mathrm{m^3\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** 1 / NSEC

PAR_TYPE_VOLFRAC Volume fractions of the particle types, have to sum to 1

Unit: – **Type:** double **Range:** $[0, 1]$ **Length:** NPARTYPE

3.5.3 Flux reconstruction methods

Group /input/model/unit_XXX/discretization/weno – WENO parameters

BOUNDARY_MODEL Boundary model type:

0 Lower WENO order (stable)

1 Zero weights (unstable for small D_{ax})

2 Zero weights for $p \neq 0$ (stable?)

3 Large ghost points

Type: int **Range:** $\{0, 1, 2, 3\}$ **Length:** 1

WENO_EPS WENO ε

Type: double **Range:** ≥ 0 **Length:** 1

WENO_ORDER WENO order, also called WENO k:

1 Standard upwind scheme (order 1)

2 WENO 2 (order 3)

3 WENO 3 (order 5)

Type: int **Range:** $\{1, 2, 3\}$ **Length:** 1

3.5.4 Nonlinear solver for consistent initialization

Group /input/model/unit_XXX/discretization/consistency_solver – Nonlinear consistency solver parameters

SOLVER_NAME Name of the solver. Available solvers are LEVMAR, ATRN_RES, ATRN_ERR, and COMPOSITE.
 Type: string **Length:** 1

INIT_DAMPING Initial damping factor (default is 0.01)
 Type: double **Range:** ≥ 0 **Length:** 1

MIN_DAMPING Minimal damping factor (default is 0.0001; ignored by LEVMAR)
 Type: double **Range:** ≥ 0 **Length:** 1

SUBSOLVERS Vector with names of solvers for the composite solver (only required for composite solver). See SOLVER_NAME for available solvers.
 Type: string **Length:** > 1

3.5.5 Binding models

Externally dependent binding models Some binding models have a variant that can use external sources as specified in Section 3.5.1 (also see Section 2.3 and Table 2.3.1 on which binding models support this feature). For the sake of brevity, only the standard variant of those binding models is specified below. In order to obtain the format for the externally dependent variant, first replace the binding model name XXX by EXT_XXX. Each parameter p (except for reference concentrations XXX_REFC0 and XXX_REFQ) depends on a (possibly distinct) external source in a polynomial way:

$$p(T) = p_{\mathrm{TTT}} T^3 + p_{\mathrm{TT}} T^2 + p_{\mathrm{T}} T + p.$$

Thus, a parameter XXX_YYY of the standard binding model variant is replaced by the four parameters EXT_XXX_YYY, EXT_XXX_YYY_T, EXT_XXX_YYY_TT, and EXT_XXX_YYY_TTT. Since each parameter can depend on a different external source, the dataset EXTFUN (not listed in the standard variants below) should contain a vector of 0-based integer indices of the external source of each parameter. The ordering of the parameters in EXTFUN is given by the ordering in the standard variant. However, if only one index is passed in EXTFUN, this external source is used for all parameters.

Note that parameter sensitivities with respect to column radius, column length, particle core radius, and particle radius may be wrong when using externally dependent binding models. This is caused by not taking into account the derivative of the external profile with respect to column position.

Non-binding components For binding models that do not support multiple bound states, many parameters can vary per component and their length is taken as NCOMP. However, these models still support non-binding components. In this case, the entries in their parameters that correspond to non-binding components are simply ignored.

Multiple particle types The group that contains the parameters of a binding model in unit operation with index XXX reads /input/model/unit_XXX/adsorption. This is valid for models with a single particle type. If a model has multiple particle types, it may have a different binding model in each type. The parameters are then placed in the group /input/model/unit_XXX/adsorption_YYY instead, where YYY denotes the index of the particle type.

Note that, in any case, /input/model/unit_XXX/adsorption_000 contains the parameters of the first (and possibly sole) particle type. This group also takes precedence over a possibly existing /input/model/unit_XXX/adsorption group.

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = LINEAR

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / `NTOTALBND`

`LIN_KA` Adsorption rate constants for each component
Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

`LIN_KD` Desorption rate constants for each component
Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MULTI_COMPONENT_LANGMUIR

`IS_KINETIC` Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.
Type: int **Range:** $\{0, 1\}$ **Length:** 1 / `NTOTALBND`

`MCL_KA` Adsorption rate constants
Type: double **Range:** ≥ 0 **Length:** `NCOMP`

`MCL_KD` Desorption rate constants
Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

`MCL_QMAX` Maximum adsorption capacities
Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** `NCOMP`

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MULTI_COMPONENT_ANTILANGMUIR

`IS_KINETIC` Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.
Type: int **Range:** $\{0, 1\}$ **Length:** 1 / `NTOTALBND`

`MCAL_KA` Adsorption rate constants
Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

`MCAL_KD` Desorption rate constants
Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

`MCAL_QMAX` Maximum adsorption capacities
Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** `NCOMP`

`MCAL_ANTILANGMUIR` Anti-Langmuir coefficients (optional)
Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** $\{-1, 1\}$ **Length:** `NCOMP`

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MOBILE_PHASE_MODULATOR

`IS_KINETIC` Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.
Type: int **Range:** $\{0, 1\}$ **Length:** 1 / `NTOTALBND`

`MPM_KA` Adsorption rate constants
Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

MPM_KD Desorption rate constants

Unit: $m_{MP}^{3\beta}\,mol^{-\beta}\,s^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

MPM_QMAX Maximum adsorption capacities

Unit: $mol\,m_{SP}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

MPM_BETA Parameters describing the ion-exchange characteristics (IEX)

Unit: $mol\,m_{SP}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

MPM_GAMMA Parameters describing the hydrophobicity (HIC)

Unit: $m_{MP}^{3}\,mol^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = EXTENDED_MOBILE_PHASE_MODULATOR

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

EMPM_COMP_MODE Determines the mode of each component (0 denotes the modifier component, 1 is linear binding, 2 is modified Langmuir binding). At most one modifier component is allowed, that is, a modifier is not required.

Note that this field has the same name for the externally dependent variant of the model.

Type: int **Range:** $\{0, 1, 2\}$ **Length:** NCOMP

EMPM_KA Adsorption rate constants

Unit: $m_{MP}^{3}\,mol^{-1}\,s^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

EMPM_KD Desorption rate constants

Unit: $m_{MP}^{3\beta}\,mol^{-\beta}\,s^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

EMPM_QMAX Maximum adsorption capacities

Unit: $mol\,m_{SP}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

EMPM_BETA Parameters describing the ion-exchange characteristics (IEX)

Unit: $mol\,m_{SP}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

EMPM_GAMMA Parameters describing the hydrophobicity (HIC)

Unit: $m_{MP}^{3}\,mol^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = STERIC_MASS_ACTION

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

SMA_KA Adsorption rate constants

Unit: $m_{MP}^{3}\,m_{SP}^{-3}\,s^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SMA_KD Desorption rate constants

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SMA_NU Characteristic charges of the proteins; The number of sites ν that the protein interacts with on the resin surface. The first entry denotes the valence of the salt ions (defaults to 1 if ≤ 0).

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMA_SIGMA Steric factors of the proteins; The number of sites σ on the surface that are shielded by the protein and prevented from exchange with the salt counterions in solution

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMA_LAMBDA Stationary phase capacity (monovalent salt counterions); The total number of binding sites available on the resin surface

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** 1

SMA_REFC0 Reference liquid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double **Range:** > 0 **Length:** 1

SMA_REFQ Reference solid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** > 0 **Length:** 1

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = SELF_ASSOCIATION

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

SAI_KA1 Adsorption rate constants

Unit: $\mathrm{m_{MP}^{3}\,m_{SP}^{-3}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SAI_KA2 Adsorption rate constants

Unit: $\mathrm{m_{MP}^{6}\,m_{SP}^{-6}\,s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SAI_KD Desorption rate constants

Unit: $\mathrm{s^{-1}}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SAI_NU Characteristic charges ν of the proteins. The first entry denotes the valence of the salt ions (defaults to 1 if ≤ 0).

Type: double **Range:** ≥ 0 **Length:** NCOMP

SAI_SIGMA Steric factors σ of the proteins

Type: double **Range:** ≥ 0 **Length:** NCOMP

SAI_LAMBDA Stationary phase capacity (monovalent salt counterions); The total number of binding sites available on the resin surface

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** ≥ 0 **Length:** 1

SAI_REFC0 Reference liquid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double **Range:** > 0 **Length:** 1

SAI_REFQ Reference solid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double **Range:** > 0 **Length:** 1

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = SASKA

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

SASKA_H Henry coefficient

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP

SASKA_K Quadratic factors

Unit: $\mathrm{m}_{\mathrm{MP}}^{6}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP^{2}

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MULTI_COMPONENT_BILANGMUIR

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

MCBL_KA Adsorption rate constants in state-major ordering

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NSTATES · NCOMP

MCBL_KD Desorption rate constants in state-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NSTATES · NCOMP

MCBL_QMAX Maximum adsorption capacities in state-major ordering

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0.0 **Length:** NSTATES · NCOMP

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = KUMAR_MULTI_COMPONENT_LANGMUIR

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

KMCL_KA Adsorption pre-exponential factors

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

KMCL_KD Desorption rate

Unit: $\mathrm{m}_{\mathrm{MP}}^{3\nu_i}\,\mathrm{mol}^{-\nu_i}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

KMCL_KACT Activation temperatures

Unit: K **Type:** double **Range:** ≥ 0 **Length:** NCOMP

KMCL_QMAX Maximum adsorption capacities

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** NCOMP

KMCL_NU Salt exponents / characteristic charges

Type: double **Range:** > 0 **Length:** NCOMP

KMCL_TEMP Temperature

Unit: K **Type:** double **Range:** ≥ 0 **Length:** 1

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MULTI_COMPONENT_SPREADING

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

MCSPR_KA Adsorption rate constants in state-major ordering

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

MCSPR_KD Desorption rate constants in state-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

MCSPR_QMAX Maximum adsorption capacities in state-major ordering

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** NTOTALBND

MCSPR_K12 Exchange rates from the first to the second bound state

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NCOMP

MCSPR_K21 Exchange rates from the second to the first bound state

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NCOMP

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = MULTISTATE_STERIC_MASS_ACTION

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

MSSMA_KA Adsorption rate constants of the components to the different bound states in component-major ordering

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

MSSMA_KD Desorption rate constants of the components in the different bound states in component-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

MSSMA_NU Characteristic charges of the components in the different bound states in component-major ordering. The first entry denotes the valence of the salt ions (defaults to 1 if ≤ 0).

Type: double **Range:** ≥ 0 **Length:** NTOTALBND

MSSMA_SIGMA Steric factors of the components in the different bound states in component-major ordering

Type: double **Range:** ≥ 0 **Length:** NTOTALBND

MSSMA_RATES Conversion rates between different bound states in component-row-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** $\sum_{i=0}^{\texttt{NCOMP}-1} \texttt{NBND}_i^2$

MSSMA_LAMBDA Stationary phase capacity (monovalent salt counterions); The total number of binding sites available on the resin surface

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** 1

MSSMA_REFC0 Reference liquid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{MP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

MSSMA_REFQ Reference solid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = SIMPLE_MULTISTATE_STERIC_MASS_ACTION

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: 1 = kinetic, 0 = quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

SMSSMA_LAMBDA Stationary phase capacity (monovalent salt counterions); The total number of binding sites available on the resin surface

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** 1

SMSSMA_KA Adsorption rate constants of the components to the different bound states in component-major ordering

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

SMSSMA_KD Desorption rate constants of the components to the different bound states in component-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NTOTALBND

SMSSMA_NU_MIN Characteristic charges of the components in the first (weakest) bound state

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_NU_MAX Characteristic charges of the components in the last (strongest) bound state

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_NU_QUAD Quadratic modifiers of the characteristic charges of the different components depending on the index of the bound state

Type: double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_SIGMA_MIN Steric factors of the components in the first (weakest) bound state

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_SIGMA_MAX Steric factors of the components in the last (strongest) bound state

Type: double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_SIGMA_QUAD Quadratic modifiers of steric factors of the different components depending on the index of the bound state

Type: double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_KWS Exchange rates from a weakly bound state to the next stronger bound state

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_KWS_LIN Linear exchange rate coefficients from a weakly bound state to the next stronger bound state

Unit: s^{-1} **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_KWS_QUAD Quadratic exchange rate coefficients from a weakly bound state to the next stronger bound state

Unit: s^{-1} **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_KSW Exchange rates from a strongly bound state to the next weaker bound state

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** NCOMP

SMSSMA_KSW_LIN Linear exchange rate coefficients from a strongly bound state to the next weaker bound state

Unit: s^{-1} **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_KSW_QUAD Quadratic exchange rate coefficients from a strongly bound state to the next weaker bound state

Unit: s^{-1} **Type:** double **Range:** $\mathbb{R}$ **Length:** NCOMP

SMSSMA_REFC0 Reference liquid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{MP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

SMSSMA_REFQ Reference solid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

SMSSMA_NU_SALT Denotes the valence of the salt ions (optional, defaults to 1)

Type: double **Range:** ≥ 0 **Length:** NCOMP

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = BI_STERIC_MASS_ACTION

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: $1 =$ kinetic, $0 =$ quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

BISMA_KA Adsorption rate constants in state-major ordering

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0.0 **Length:** NSTATES $\cdot$ NCOMP

BISMA_KD Desorption rate constants in state-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0.0 **Length:** NSTATES $\cdot$ NCOMP

BISMA_NU Characteristic charges $\nu_{i,j}$ of the ith protein with respect to the jth binding site type in state-major ordering. The first entry of each binding site type denotes the valence of the salt ions (defaults to 1 if ≤ 0).

Unit: s^{-1} **Type:** double **Range:** ≥ 0.0 **Length:** NSTATES $\cdot$ NCOMP

BISMA_SIGMA Steric factors $\sigma_{i,j}$ of the ith protein with respect to the jth binding site type in state-major ordering

Unit: s^{-1} **Type:** double **Range:** ≥ 0.0 **Length:** NSTATES $\cdot$ NCOMP

BISMA_LAMBDA Stationary phase capacity (monovalent salt counterions) of the different binding site types λ_j

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** ≥ 0.0 **Length:** NSTATES

BISMA_REFC0 Reference liquid phase concentration for each binding site type or one value for all types (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{MP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** $\{1, \texttt{NSTATES}\}$

BISMA_REFQ Reference solid phase concentration for each binding site type or one value for all types (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** $\{1, \texttt{NSTATES}\}$

Group /input/model/unit_XXX/adsorption – ADSORPTION_MODEL = GENERALIZED_ION_EXCHANGE

IS_KINETIC Selects kinetic or quasi-stationary adsorption mode: $1 =$ kinetic, $0 =$ quasi-stationary. If a single value is given, the mode is set for all bound states. Otherwise, the adsorption mode is set for each bound state separately.

Type: int **Range:** $\{0, 1\}$ **Length:** 1 / NTOTALBND

GIEX_KA Base value of adsorption rate constant

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{m}_{\mathrm{SP}}^{-3}\,\mathrm{s}^{-1}$ **Type:** double **Range:** ≥ 0 **Length:** NCOMP

`GIEX_KA_LIN` Coefficient of linear dependence of adsorption rate constant on modifier component

Unit: $[\mathrm{Mod}]^{-1}$ **Type:** double **Length:** `NCOMP`

`GIEX_KA_QUAD` Coefficient of quadratic dependence of adsorption rate constant on modifier component

Unit: $[\mathrm{Mod}]^{-2}$ **Type:** double **Length:** `NCOMP`

`GIEX_KA_SALT` Salt coefficient of adsorption rate constants; difference of water-protein and salt-protein interactions

Type: double **Length:** `NCOMP`

`GIEX_KA_PROT` Protein coefficient of adsorption rate constants; difference of water-protein and protein-protein interactions

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}$ **Type:** double **Length:** `NCOMP`

`GIEX_KD` Base value of desorption rate constant

Unit: s^{-1} **Type:** double **Range:** ≥ 0 **Length:** `NCOMP`

`GIEX_KD_LIN` Coefficient of linear dependence of desorption rate constant on modifier component

Unit: $[\mathrm{Mod}]^{-1}$ **Type:** double **Length:** `NCOMP`

`GIEX_KD_QUAD` Coefficient of quadratic dependence of desorption rate constant on modifier component

Unit: $[\mathrm{Mod}]^{-2}$ **Type:** double **Length:** `NCOMP`

`GIEX_KD_SALT` Salt coefficient of desorption rate constants; difference of water-protein and salt-protein interactions

Type: double **Length:** `NCOMP`

`GIEX_KD_PROT` Protein coefficient of desorption rate constants; difference of water-protein and protein-protein interactions

Unit: $\mathrm{m}_{\mathrm{MP}}^{3}\,\mathrm{mol}^{-1}$ **Type:** double **Length:** `NCOMP`

`GIEX_NU` Base value for characteristic charges of the protein; The number of sites ν that the protein interacts with on the resin surface. The first entry denotes the valence of the salt ions (defaults to 1 if ≤ 0).

Type: double **Length:** `NCOMP`

`GIEX_NU_LIN` Coefficient of linear dependence of characteristic charge on modifier component

Unit: $[\mathrm{Mod}]^{-1}$ **Type:** double **Length:** `NCOMP`

`GIEX_NU_QUAD` Coefficient of quadratic dependence of characteristic charge on modifier component

Unit: $[\mathrm{Mod}]^{-2}$ **Type:** double **Length:** `NCOMP`

`GIEX_SIGMA` Steric factors of the protein; The number of sites σ on the surface that are shielded by the protein and prevented from exchange with the salt counterions in solution

Type: double **Range:** ≥ 0 **Length:** `NCOMP`

`GIEX_LAMBDA` Stationary phase capacity (monovalent salt counterions); The total number of binding sites available on the resin surface

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** ≥ 0 **Length:** 1

`GIEX_REFC0` Reference liquid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{MP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

`GIEX_REFQ` Reference solid phase concentration (optional, defaults to 1.0)

Unit: $\mathrm{mol}\,\mathrm{m}_{\mathrm{SP}}^{-3}$ **Type:** double **Range:** > 0 **Length:** 1

3.5.6 Reaction models

Externally dependent reaction models Some reaction models have a variant that can use external sources as specified in Section 3.5.1 (also see Section 2.4). For the sake of brevity, only the standard variant of those reaction models is specified below. In order to obtain the format for the externally dependent variant, first replace the reaction model name `XXX` by `EXT_XXX`. Each parameter p (except for stoichiometric and exponent matrices) depends on a (possibly distinct) external source in a polynomial way:

$$p(T) = p_{\texttt{TTT}} T^3 + p_{\texttt{TT}} T^2 + p_{\texttt{T}} T + p.$$

Thus, a parameter `XXX_YYY` of the standard reaction model variant is replaced by the four parameters `EXT_XXX_YYY`, `EXT_XXX_YYY_T`, `EXT_XXX_YYY_TT`, and `EXT_XXX_YYY_TTT`. Since each parameter can depend on a different external source, the dataset `EXTFUN` (not listed in the standard variants below) should contain a vector of 0-based integer indices of the external source of each parameter. The ordering of the parameters in `EXTFUN` is given by the ordering in the standard variant. However, if only one index is passed in `EXTFUN`, this external source is used for all parameters.

Note that parameter sensitivities with respect to column radius, column length, particle core radius, and particle radius may be wrong when using externally dependent reaction models. This is caused by not taking into account the derivative of the external profile with respect to column position.

Multiple particle types The group that contains the parameters of a reaction model in unit operation with index `XXX` reads `/input/model/unit_XXX/reaction_particle`. This is valid for models with a single particle type. If a model has multiple particle types, it may have a different reaction model in each type. The parameters are then placed in the group `/input/model/unit_XXX/reaction_particle_YYY` instead, where `YYY` denotes the index of the particle type.

Note that, in any case, `/input/model/unit_XXX/reaction_particle_000` contains the parameters of the first (and possibly sole) particle type. This group also takes precedence over a possibly existing `/input/model/unit_XXX/adsorption_particle` group.

Group /input/model/unit_XXX/reaction – REACTION_MODEL = MASS_ACTION_LAW

`MAL_KFWD_BULK` Forward rate constants for bulk volume reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_KBWD_BULK` Backward rate constants for bulk volume reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_KFWD_LIQUID` Forward rate constants for particle liquid phase reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_KBWD_LIQUID` Backward rate constants for particle liquid phase reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_KFWD_SOLID` Forward rate constants for particle solid phase reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_KBWD_SOLID` Backward rate constants for particle solid phase reactions (available for external functions)

Type: double **Range:** ≥ 0 **Length:** `NREACT`

`MAL_STOICHIOMETRY_BULK` Stoichiometric matrix of bulk volume reactions as `NCOMP` $\times$ `NREACT` matrix in row-major storage

Type: double **Length:** `NCOMP` $\cdot$ `NREACT`

MAL_EXPONENTS_BULK_FWD Forward exponent matrix of bulk volume reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_BULK by default)

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_EXPONENTS_BULK_BWD Backward exponent matrix of bulk volume reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_BULK by default)

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_STOICHIOMETRY_LIQUID Stoichiometric matrix of particle liquid phase reactions as NCOMP $\times$ NREACT matrix in row-major storage

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_EXPONENTS_LIQUID_FWD Forward exponent matrix of particle liquid phase reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_LIQUID by default)

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_EXPONENTS_LIQUID_BWD Backward exponent matrix of particle liquid phase reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_LIQUID by default)

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_EXPONENTS_LIQUID_FWD_MODSOLID Forward solid phase modifier exponent matrix of particle liquid phase reactions as NTOTALBND $\times$ NREACT matrix in row-major storage (optional, defaults to all 0)

Type: double **Length:** NTOTALBND $\cdot$ NREACT

MAL_EXPONENTS_LIQUID_BWD_MODSOLID Backward solid phase modifier exponent matrix of particle liquid phase reactions as NTOTALBND $\times$ NREACT matrix in row-major storage (optional, defaults to all 0)

Type: double **Length:** NTOTALBND $\cdot$ NREACT

MAL_STOICHIOMETRY_SOLID Stoichiometric matrix of particle solid phase reactions as NTOTALBND $\times$ NREACT matrix in row-major storage

Type: double **Length:** NTOTALBND $\cdot$ NREACT

MAL_EXPONENTS_SOLID_FWD Forward exponent matrix of particle solid phase reactions as NTOTALBND $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_SOLID by default)

Type: double **Length:** NTOTALBND $\cdot$ NREACT

MAL_EXPONENTS_SOLID_BWD Backward exponent matrix of particle solid phase reactions as NTOTALBND $\times$ NREACT matrix in row-major storage (optional, calculated from MAL_STOICHIOMETRY_SOLID by default)

Type: double **Length:** NTOTALBND $\cdot$ NREACT

MAL_EXPONENTS_SOLID_FWD_MODLIQUID Forward liquid phase modifier exponent matrix of particle solid phase reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, defaults to all 0)

Type: double **Length:** NCOMP $\cdot$ NREACT

MAL_EXPONENTS_SOLID_BWD_MODLIQUID Backward liquid phase modifier exponent matrix of particle solid phase reactions as NCOMP $\times$ NREACT matrix in row-major storage (optional, defaults to all 0)

Type: double **Length:** NCOMP $\cdot$ NREACT

3.5.7 Return data

Group /input/return

WRITE_SOLUTION_TIMES Write times at which a solution was produced (optional, defaults to 1)

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_LAST Write full solution state vector at last time point (optional, defaults to 0)

Type: int **Range:** $\{0, 1\}$

WRITE_SENS_LAST Write full sensitivity state vectors at last time point (optional, defaults to 0)

Type: int **Range:** $\{0, 1\}$

SPLIT_COMPONENTS_DATA Determines whether a joint dataset (matrix or tensor) for all components is created or if each component is put in a separate dataset (`XXX_COMP_000`, `XXX_COMP_001`, etc.) (optional, defaults to 1)

Type: int **Range:** $\{0, 1\}$

SPLIT_PORTS_DATA Determines whether a joint dataset (matrix or tensor) for all inlet/outlet ports is created or if each port is put in a separate dataset (`XXX_PORT_000`, `XXX_PORT_001`, etc.) (optional, defaults to 1)

Type: int **Range:** $\{0, 1\}$

SINGLE_AS_MULTI_PORT Determines whether single port unit operations are treated as multi port unit operations in the output naming scheme (i.e., `_PORT_XYZ_` is added to the name) (optional, defaults to 0)

Type: int **Range:** $\{0, 1\}$

Group /input/return/unit_XXX

WRITE_COORDINATES Write coordinates of discretization nodes

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_INLET Write solutions at unit operation inlet $c^l_i(t, 0)$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_OUTLET Write solutions at unit operation outlet (chromatograms) $c^l_i(t, L)$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_BULK Write solutions of the bulk volume c^l_i

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_PARTICLE Write solutions of the particle mobile phase $c^p_{j,i}$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_SOLID Write solutions of the solid phase $c^s_{j,i,m_{j,i}}$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_FLUX Write solutions of the bead fluxes $j_{f,i}$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLUTION_VOLUME Write solutions of the volume V

Type: int **Range:** $\{0, 1\}$

WRITE_SOLDOT_INLET Write solution time derivatives at unit operation inlet $\partial c^l_i(t, 0)/\partial t$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLDOT_OUTLET Write solution time derivatives at unit operation outlet (chromatograms) $\partial c^l_i(t, L)/\partial t$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLDOT_BULK Write solution time derivatives of the bulk volume $\partial c^l_i/\partial t$

Type: int **Range:** $\{0, 1\}$

WRITE_SOLDOT_PARTICLE Write solution time derivatives of the particle mobile phase $\partial c^{p}_{j,i}/\partial t$

Type: int **Range:** $\{0,1\}$

WRITE_SOLDOT_SOLID Write solution time derivatives of the solid phase $\partial c^{s}_{j,i,m_{j,i}}/\partial t$

Type: int **Range:** $\{0,1\}$

WRITE_SOLDOT_FLUX Write solution time derivatives of the bead fluxes $\partial j_{f,i}/\partial t$

Type: int **Range:** $\{0,1\}$

WRITE_SOLDOT_VOLUME Write solution time derivatives of the volume $\partial V/\partial t$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_INLET Write sensitivities at unit operation inlet $\partial c^{l}_{i}(t,0)/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_OUTLET Write sensitivities at unit operation outlet (chromatograms) $\partial c^{l}_{i}(t,L)/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_BULK Write sensitivities of the bulk volume $\partial c^{l}_{i}/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_PARTICLE Write sensitivities of the particle mobile phase $\partial c^{p}_{j,i}/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_SOLID Write sensitivities of the solid phase $\partial c^{s}_{j,i,m_{j,i}}/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_FLUX Write sensitivities of the bead fluxes $\partial j_{f,i}/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENS_VOLUME Write sensitivities of the volume $\partial V/\partial p$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_INLET Write sensitivity time derivatives at unit operation inlet $\partial^2 c^{l}_{i}(t,0)/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_OUTLET Write sensitivity time derivatives at unit operation outlet (chromatograms) $\partial^2 c^{l}_{i}(t,L)/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_BULK Write sensitivity time derivatives of the bulk volume $\partial^2 c^{l}_{i}/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_PARTICLE Write sensitivity time derivatives of the particle mobile phase $\partial^2 c^{p}_{j,i}/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_SOLID Write sensitivity time derivatives of the solid phase $\partial^2 c^{s}_{j,i,m_{j,i}}/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_FLUX Write sensitivity time derivatives of the bead fluxes $\partial^2 j_{f,i}/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

WRITE_SENSDOT_VOLUME Write sensitivity time derivatives of the volume $\partial^2 V/(\partial p,\partial t)$

Type: int **Range:** $\{0,1\}$

3.5.8 Parameter sensitivities

Group `/input/sensitivity`

`NSENS` Number of sensitivities to be computed

Type: int **Range:** ≥ 0 **Length:** 1

`SENS_METHOD` Method used for computation of sensitivities (algorithmic differentiation)

Type: string **Range:** `ad1` **Length:** 1

Group `/input/sensitivity/param_XXX`

`SENS_UNIT` Unit operation index

Type: int **Range:** ≥ 0 **Length:** ≥ 1

`SENS_NAME` Name of the parameter

Type: string **Range:** See **Length:** ≥ 1

`SENS_COMP` Component index (-1 if parameter is independent of components)

Type: int **Range:** ≥ -1 **Length:** ≥ 1

`SENS_PARTYPE` Particle type index (-1 if parameter is independent of particle types)

Type: int **Range:** ≥ -1 **Length:** ≥ 1

`SENS_REACTION` Reaction index (-1 if parameter is independent of reactions)

Type: int **Range:** ≥ -1 **Length:** ≥ 1

`SENS_BOUNDPHASE` Bound phase index (-1 if parameter is independent of bound phases)

Type: int **Range:** ≥ -1 **Length:** ≥ 1

`SENS_SECTION` Section index (-1 if parameter is independent of sections)

Type: int **Range:** ≥ -1 **Length:** ≥ 1

`SENS_ABSTOL` Absolute tolerance used in the computation of the sensitivities (optional). Rule of thumb: `ABSTOL/PARAM_VALUE`

Type: double **Range:** ≥ 0.0 **Length:** ≥ 1

`SENS_FACTOR` Linear factor of the combined sensitivity (optional, taken as 1.0 if left out)

Type: double **Range:** $\mathbb{R}$ **Length:** ≥ 1

3.5.9 Solver configuration

Group `/input/solver`

`NTHREADS` Number of used threads

Type: int **Range:** ≥ 1 **Length:** 1

`USER_SOLUTION_TIMES` Vector with timepoints at which the solution is evaluated

Unit: s **Type:** double **Range:** ≥ 0 **Length:** Arbitrary

`CONSISTENT_INIT_MODE` Consistent initialization mode (optional, defaults to 1). Valid values are:

0 None

1 Full

2 Once, full

3 Lean

4 Once, lean

5 Full once, then lean

6 None once, then full

7 None once, then lean

Type: int **Range:** $\{0, \ldots, 7\}$ **Length:** 1

CONSISTENT_INIT_MODE_SENS Consistent initialization mode for parameter sensitivities (optional, defaults to 1). Valid values are:

0 None

1 Full

2 Once, full

3 Lean

4 Once, lean

5 Full once, then lean

6 None once, then full

7 None once, then lean

Type: int **Range:** $\{0, \ldots, 7\}$ **Length:** 1

Group /input/solver/time_integrator

ABSTOL Absolute tolerance in the solution of the original system

Type: double **Range:** > 0 **Length:** 1

RELTOL Relative tolerance in the solution of the original system

Type: double **Range:** ≥ 0 **Length:** 1

ALGTOL Tolerance in the solution of the nonlinear consistency equations

Type: double **Range:** > 0 **Length:** 1

RELTOL_SENS Relative tolerance in the solution of the sensitivity systems

Type: double **Range:** ≥ 0 **Length:** 1

INIT_STEP_SIZE Initial time integrator step size for each section or one value for all sections (0.0: IDAS default value), see IDAS guide 4.5, p. 36f.

Unit: s **Type:** double **Range:** ≥ 0 **Length:** 1 / NSEC

MAX_STEPS Maximum number of timesteps taken by IDAS (0: IDAS default = 500), see IDAS guide Sec. 4.5

Type: int **Range:** ≥ 0 **Length:** 1

MAX_STEP_SIZE Maximum size of timesteps taken by IDAS (optional, defaults to unlimited 0.0), see IDAS guide Sec. 4.5

Unit: s **Type:** double **Range:** ≥ 0 **Length:** 1

ERRORTEST_SENS Determines whether (forward) sensitivities take part in local error test (optional, defaults to 1)

Type: int **Range:** $\{0, 1\}$ **Length:** 1

MAX_NEWTON_ITER Maximum number of Newton iterations in time step (optional, defaults to 3)

Type: int **Range:** ≥ 0 **Length:** 1

MAX_ERRTEST_FAIL Maximum number of local error test failures in time step (optional, defaults to 7)

Type: int **Range:** ≥ 0 **Length:** 1

MAX_CONVTEST_FAIL Maximum number of Newton convergence test failures (optional, defaults to 10)

Type: int **Range:** ≥ 0 **Length:** 1

MAX_NEWTON_ITER_SENS Maximum number of Newton iterations in forward sensitivity time step (optional, defaults to 3)

Type: int **Range:** ≥ 0 **Length:** 1

Group `/input/solver/sections`

NSEC Number of sections

Type: int **Range:** ≥ 1 **Length:** 1

SECTION_TIMES Simulation times at which the model changes or behaves discontinously; including start and end times

Unit: s **Type:** double **Range:** ≥ 0 **Length:** `NSEC` $+ 1$

SECTION_CONTINUITY Continuity indicator for each section transition: 0 (discontinuous) or 1 (continuous).

Type: int **Range:** $\{0, 1\}$ **Length:** `NSEC` $- 1$

3.6 Output group

Group `/output/solution`

LAST_STATE Full state vector at the last time point of the time integrator if `WRITE_SOLUTION_LAST` in `/input/return` is enabled

Type: double

LAST_STATE_YDOT Full time derivative state vector at the last time point of the time integrator if `WRITE_SOLUTION_LAST` in `/input/return` is enabled

Type: double

LAST_STATE_SENSY_XXX Full state vector of the `XXX`th sensitivity system at the last time point of the time integrator if `WRITE_SENS_LAST` in `/input/return` is enabled

Type: double

LAST_STATE_SENSYDOT_XXX Full time derivative state vector of the `XXX`th sensitivity system at the last time point of the time integrator if `WRITE_SENS_LAST` in `/input/return` is enabled

Type: double

Group `/output/solution`

SOLUTION_TIMES Time points at which the solution is written if `WRITE_SOLUTION_TIMES` in `/input/return` is enabled

Unit: s **Type:** double

Group `/output/solution/unit_XXX`

SOLUTION_BULK Interstitial solution as $n_{\text{Time}} \times$ `UNITOPORDERING` tensor in row-major storage

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}$ **Type:** double

SOLUTION_PARTICLE Mobile phase solution inside the particles as $n_{\text{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\text{mol}\,\text{m}_{\text{MP}}^{-3}$ **Type:** double

SOLUTION_PARTICLE_PARTYPE_XXX Mobile phase solution inside the particles of type XXX as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double

SOLUTION_SOLID Solid phase solution inside the particles as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}}$ **Type:** double

SOLUTION_SOLID_PARTYPE_XXX Solid phase solution inside the particles of type XXX as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{SP}^{-3}}$ **Type:** double

SOLUTION_FLUX Flux solution as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage

Unit: $\mathrm{mol\,m^{-2}\,s^{-1}}$ **Type:** double

SOLUTION_VOLUME Volume solution

Unit: $\mathrm{m^3}$ **Type:** double

SOLUTION_OUTLET Tensor of solutions at the unit operation outlet with components as columns in time-port-major storage. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are both disabled. If the unit operation only has a single port, the port-dimension is removed if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_INLET Tensor of solutions at the unit operation inlet with components as columns in time-port-major storage. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are both disabled. If the unit operation only has a single port, the port-dimension is removed if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_OUTLET_COMP_XXX Component XXX of the solution at all outlet ports of the unit operation as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is enabled and SPLIT_PORTS_DATA is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_INLET_COMP_XXX Component XXX of the solution at all inlet ports of the unit operation inlet as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is enabled and SPLIT_PORTS_DATA is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_OUTLET_PORT_XXX All components at outlet port XXX of the solution of the unit operation as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_INLET_PORT_XXX All components at inlet port XXX of the solution of the unit operation inlet as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

SOLUTION_OUTLET_PORT_XXX_COMP_YYY Component YYY at outlet port XXX of the solution of the unit operation. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are enabled, and the unit operation has multiple outlet ports. If the unit operation only has a single port, the field is created if SINGLE_AS_MULTI_PORT is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

`SOLUTION_INLET_PORT_XXX_COMP_YYY` Component `YYY` at inlet port `XXX` of the solution of the unit operation. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are enabled, and the unit operation has multiple inlet ports. If the unit operation only has a single port, the field is created if `SINGLE_AS_MULTI_PORT` is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ **Type:** double

`SOLDOT_BULK` Interstitial solution time derivative as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_PARTICLE` Mobile phase solution time derivative inside the particles as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_PARTICLE_PARTYPE_XXX` Mobile phase solution time derivative inside the particles of type `XXX` as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_SOLID` Solid phase solution time derivative inside the particles as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_SOLID_PARTYPE_XXX` Solid phase solution time derivative inside the particles of type `XXX` as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{SP}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_FLUX` Flux solution time derivative as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage

Unit: $\mathrm{mol\,m^{-2}\,s^{-2}}$ **Type:** double

`SOLDOT_VOLUME` Volume solution time derivative

Unit: $\mathrm{m^{3}\,s^{-1}}$ **Type:** double

`SOLDOT_OUTLET` Tensor of solution time derivatives at the unit operation outlet with components as columns in time-port-major storage. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are both disabled. If the unit operation only has a single port, the port-dimension is removed if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_INLET` Tensor of solution time derivatives at the unit operation inlet with components as columns in time-port-major storage. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are both disabled. If the unit operation only has a single port, the port-dimension is removed if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_OUTLET_COMP_XXX` Component `XXX` of the solution time derivative at all outlet ports of the unit operation as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is enabled and `SPLIT_PORTS_DATA` is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

`SOLDOT_INLET_COMP_XXX` Component `XXX` of the solution time derivative at all inlet ports of the unit operation inlet as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is enabled and `SPLIT_PORTS_DATA` is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

SOLDOT_OUTLET_PORT_XXX All components at outlet port XXX of the solution time derivative of the unit operation as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

SOLDOT_INLET_PORT_XXX All components at inlet port XXX of the solution time derivative of the unit operation inlet as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

SOLDOT_OUTLET_PORT_XXX_COMP_YYY Component YYY at outlet port XXX of the solution time derivative of the unit operation. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are enabled, and the unit operation has multiple outlet ports. If the unit operation only has a single port, the field is created if SINGLE_AS_MULTI_PORT is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

SOLDOT_INLET_PORT_XXX_COMP_YYY Component YYY at inlet port XXX of the solution time derivative of the unit operation. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are enabled, and the unit operation has multiple inlet ports. If the unit operation only has a single port, the field is created if SINGLE_AS_MULTI_PORT is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ **Type:** double

Group /output/sensitivity/param_XXX/unit_YYY

SENS_BULK Interstitial sensitivity as $n_{\mathrm{Time}} \times$ UNITOPORDERING tensor in row-major storage

Unit: $\mathrm{mol\,m_{IV}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_PARTICLE Mobile phase sensitivity inside the particles as $n_{\mathrm{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_PARTICLE_PARTYPE_XXX Mobile phase sensitivity inside the particles of type XXX as $n_{\mathrm{Time}} \times$UNITOPORDERING tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_SOLID Solid phase sensitivity inside the particles as $n_{\mathrm{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_SOLID_PARTYPE_XXX Solid phase sensitivity inside the particles of type XXX as $n_{\mathrm{Time}} \times$UNITOPORDERING tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{SP}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_FLUX Flux sensitivity as $n_{\mathrm{Time}} \times$ UNITOPORDERING tensor in row-major storage

Unit: $\mathrm{mol\,m^{-2}\,s^{-1}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_VOLUME Volume sensitivity

Unit: $\mathrm{m^{3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

SENS_OUTLET Tensor of sensitivities at the unit operation outlet with components as columns in time-port-major storage. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are both disabled. If the unit operation only has a single port, the port-dimension is removed if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}\,[\mathrm{Param}]^{-1}$ **Type:** double

`SENS_INLET` Tensor of sensitivities at the unit operation inlet with components as columns in time-port-major storage. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are both disabled. If the unit operation only has a single port, the port-dimension is removed if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_OUTLET_COMP_XXX` Component `XXX` of the sensitivity at all outlet ports of the unit operation as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is enabled and `SPLIT_PORTS_DATA` is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_INLET_COMP_XXX` Component `XXX` of the sensitivity at all inlet ports of the unit operation inlet as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is enabled and `SPLIT_PORTS_DATA` is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if `SINGLE_AS_MULTI_PORT` is disabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_OUTLET_PORT_XXX` All components at outlet port `XXX` of the sensitivity of the unit operation as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is disabled and `SPLIT_PORTS_DATA` is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_INLET_PORT_XXX` All components at inlet port `XXX` of the sensitivity of the unit operation inlet as matrix in time-major storage. Only present if `SPLIT_COMPONENTS_DATA` is disabled and `SPLIT_PORTS_DATA` is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_OUTLET_PORT_XXX_COMP_YYY` Component `YYY` at outlet port `XXX` of the sensitivity of the unit operation. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are enabled, and the unit operation has multiple outlet ports. If the unit operation only has a single port, the field is created if `SINGLE_AS_MULTI_PORT` is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENS_INLET_PORT_XXX_COMP_YYY` Component `YYY` at inlet port `XXX` of the sensitivity of the unit operation. Only present if `SPLIT_COMPONENTS_DATA` and `SPLIT_PORTS_DATA` are enabled, and the unit operation has multiple inlet ports. If the unit operation only has a single port, the field is created if `SINGLE_AS_MULTI_PORT` is enabled.

Unit: $\mathrm{mol\,m_{IV}^{-3}}$ [Param]$^{-1}$ **Type:** double

`SENSDOT_BULK` Interstitial sensitivity time derivative as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage

Unit: $\mathrm{mol\,m_{IV}^{-3}\,s^{-1}}$ [Param]$^{-1}$ **Type:** double

`SENSDOT_PARTICLE` Mobile phase sensitivity time derivative inside the particles as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ [Param]$^{-1}$ **Type:** double

`SENSDOT_PARTICLE_PARTYPE_XXX` Mobile phase sensitivity time derivative inside the particles of type `XXX` as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ [Param]$^{-1}$ **Type:** double

`SENSDOT_SOLID` Solid phase sensitivity time derivative inside the particles as $n_{\mathrm{Time}} \times$ `UNITOPORDERING` tensor in row-major storage. Only present if just one particle type is defined.

Unit: $\mathrm{mol\,m_{MP}^{-3}\,s^{-1}}$ [Param]$^{-1}$ **Type:** double

SENSDOT_SOLID_PARTYPE_XXX Solid phase sensitivity time derivative inside the particles of type XXX as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage. Only present if more than one particle type is defined.

Unit: $\text{mol}\,\text{m}_{\text{SP}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_FLUX Flux sensitivity time derivative as $n_{\text{Time}} \times$ UNITOPORDERING tensor in row-major storage

Unit: $\text{mol}\,\text{m}^{-2}\,\text{s}^{-2}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_VOLUME Volume sensitivity time derivative

Unit: $\text{s}^{-3}\,\text{m}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_OUTLET Tensor of sensitivity time derivatives at the unit operation outlet with components as columns in time-port-major storage. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are both disabled. If the unit operation only has a single port, the port-dimension is removed if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_INLET Tensor of sensitivity time derivatives at the unit operation inlet with components as columns in time-port-major storage. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are both disabled. If the unit operation only has a single port, the port-dimension is removed if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_OUTLET_COMP_XXX Component XXX of the sensitivity time derivative at all outlet ports of the unit operation as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is enabled and SPLIT_PORTS_DATA is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_INLET_COMP_XXX Component XXX of the sensitivity time derivative at all inlet ports of the unit operation inlet as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is enabled and SPLIT_PORTS_DATA is disabled. If the unit operation only has a single port, a vector (1D array) is returned instead of a matrix if SINGLE_AS_MULTI_PORT is disabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_OUTLET_PORT_XXX All components at outlet port XXX of the sensitivity time derivative of the unit operation as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_INLET_PORT_XXX All components at inlet port XXX of the sensitivity time derivative of the unit operation inlet as matrix in time-major storage. Only present if SPLIT_COMPONENTS_DATA is disabled and SPLIT_PORTS_DATA is enabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_OUTLET_PORT_XXX_COMP_YYY Component YYY at outlet port XXX of the sensitivity time derivative of the unit operation. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are enabled, and the unit operation has multiple outlet ports. If the unit operation only has a single port, the field is created if SINGLE_AS_MULTI_PORT is enabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

SENSDOT_INLET_PORT_XXX_COMP_YYY Component YYY at inlet port XXX of the sensitivity time derivative of the unit operation. Only present if SPLIT_COMPONENTS_DATA and SPLIT_PORTS_DATA are enabled, and the unit operation has multiple inlet ports. If the unit operation only has a single port, the field is created if SINGLE_AS_MULTI_PORT is enabled.

Unit: $\text{mol}\,\text{m}_{\text{IV}}^{-3}\,\text{s}^{-1}\,[\text{Param}]^{-1}$ **Type:** double

Group `/output/coordinates/unit_XXX`

`AXIAL_COORDINATES` Axial coordinates of the bulk discretization nodes

Unit: m **Type:** double **Length:** `NCOL`

`RADIAL_COORDINATES` Radial coordinates of the bulk discretization nodes (only for 2D unit operations)

Unit: m **Type:** double **Length:** `NRAD`

`PARTICLE_COORDINATES_XXX` Coordinates of the particle discretization nodes in particles of type `XXX`

Unit: m **Type:** double **Length:** `NPAR`

3.7 Meta group

Group `/meta`

`FILE_FORMAT` Version of the file format (defaults to 040000 = 4.0.0 if omitted) with two digits per part (Major.Minor.Patch)

In/out: In **Type:** int

`CADET_VERSION` Version of the executed CADET simulator

In/out: Out **Type:** string

`CADET_COMMIT` Git commit SHA1 from which the CADET simulator was built

In/out: Out **Type:** string

`CADET_BRANCH` Git branch from which the CADET simulator was built

In/out: Out **Type:** string

`TIME_SIM` Time that the time integration took (excluding any preparations and postprocessing)

Unit: s **In/out:** Out **Type:** double

Bibliography

[BC92] Clayton A. Brooks and Steven M. Cramer. “Steric mass-action ion exchange: Displacement profiles and induced salt gradients”. In: *AIChE Journal* 38.12 (Dec. 1992), pp. 1969–1978. DOI: 10.1002/aic.690381212.

[Dan53] P.V. Danckwerts. “Continuous flow systems: Distribution of residence times”. In: *Chemical Engineering Science* 2.1 (Feb. 1953), pp. 1–13. DOI: 10.1016/0009-2509(53)80001-1.

[FG04] A. Felinger and G. Guiochon. “Comparison of the Kinetic Models of Linear Chromatography”. In: *Chromatographia* 60.S1 (May 2004), pp. 175–180. DOI: 10.1365/s10337-004-0288-7.

[FTB97] William F. Feehery, John E. Tolsma, and Paul I. Barton. “Efficient sensitivity analysis of large-scale differential-algebraic systems”. In: *Applied Numerical Mathematics* 25.1 (Oct. 1997), pp. 41–54. DOI: 10.1016/S0168-9274(97)00050-0.

[Gu95] Tingyue Gu. *Mathematical Modeling and Scale-up of Liquid Chromatography.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. DOI: 10.1007/978-3-642-79541-1.

[Gui+06] Georges Guiochon et al. *Fundamentals of Preparative and Nonlinear Chromatography.* 2nd. Amsterdam: Elsevier Academic Press, 2006, p. 990.

[Hin+05] Alan C. Hindmarsh et al. “SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers”. In: *ACM Transactions on Mathematical Software* 31.3 (Sept. 2005), pp. 363–396. DOI: 10.1145/1089014.1089020.

[Huu+17] Thiemo C. Huuk et al. “Modeling of complex antibody elution behavior under high protein load densities in ion exchange chromatography using an asymmetric activity coefficient”. In: *Biotechnology Journal* 12.3 (Mar. 2017), p. 1600336. DOI: 10.1002/biot.201600336.

[Kar+04] David Karlsson et al. “Model-based optimization of a preparative ion-exchange step for antibody purification”. In: *Journal of Chromatography A* 1055.1-2 (Nov. 2004), pp. 29–39. DOI: 10.1016/j.chroma.2004.08.151.

[Kuč65] Eugene Kučera. “Contribution to the theory of chromatography”. In: *Journal of Chromatography* 19 (Jan. 1965), pp. 237–248. DOI: 10.1016/S0021-9673(01)99457-9.

[Kum+15] Vijesh Kumar et al. “Mechanistic modeling of ion-exchange process chromatography of charge variants of monoclonal antibody products”. In: *Journal of Chromatography A* 1426 (Dec. 2015), pp. 140–153. DOI: 10.1016/j.chroma.2015.11.062.

[Lan16] Irving Langmuir. “The Constitution and Fundamental Properties of Solids and Liquids. Part I. Solids”. In: *Journal of the American Chemical Society* 38.11 (Nov. 1916), pp. 2221–2295. DOI: 10.1021/ja02268a002.

[MEH89] Wayne R. Melander, Ziad El Rassi, and Csaba Horváth. “Interplay of hydrophobic and electrostatic interactions in biopolymer chromatography”. In: *Journal of Chromatography A* 469 (Jan. 1989), pp. 3–27. DOI: 10.1016/S0021-9673(01)96437-4.

[Miy07] Kanji Miyabe. “Surface diffusion in reversed-phase liquid chromatography using silica gel stationary phases of different C1 and C18 ligand densities.” In: *Journal of Chromatography A* 1167.2 (Oct. 2007), pp. 161–70. DOI: 10.1016/j.chroma.2007.08.045.

[Mol08] Jørgen M. Mollerup. “A Review of the Thermodynamics of Protein Association to Ligands, Protein Adsorption, and Adsorption Isotherms”. In: *Chemical Engineering & Technology* 31.6 (June 2008), pp. 864–874. DOI: 10.1002/ceat.200800082.

[MWW96] Z. Ma, R. D. Whitley, and N.-H. L. Wang. “Pore and surface diffusion in multicomponent adsorption and liquid chromatography systems”. In: *AIChE Journal* 42.5 (May 1996), pp. 1244–1262. DOI: 10.1002/aic.690420507.

[Püt+16] Andreas Püttmann et al. “Utilizing algorithmic differentiation to efficiently compute chromatograms and parameter sensitivities”. In: *Chemical Engineering Science* 139 (Jan. 2016), pp. 152–162. DOI: `10.1016/j.ces.2015.08.050`.

[Sas+92] M. Saska et al. “Continuous Separation of Sugarcane Molasses with a Simulated Moving-Bed Adsorber. Adsorption Equilibria, Kinetics, and Application”. In: *Separation Science and Technology* 27.13 (Oct. 1992), pp. 1711–1732. DOI: `10.1080/01496399208019442`.

[SS68] Petr Schneider and J. M. Smith. “Chromatographic study of surface diffusion”. In: *AIChE Journal* 14.6 (Nov. 1968), pp. 886–895. DOI: `10.1002/aic.690140613`.

[Wes+12] K. Westerberg et al. “Model-Based Process Challenge of an Industrial Ion-Exchange Chromatography Step”. In: *Chemical Engineering & Technology* 35.1 (Jan. 2012), pp. 183–190. DOI: `10.1002/ceat.201000560`.