

Technical university of Liberec
Faculty of mechatronics, informatics
and interdisciplinary studies

**FLOW123D - draft, scheme of new
inputs**

version 1.7.0

**Documentation of file formats
and brief user manual.**

Liberec, 2011

Acknowledgement. This work was realized under the state subsidy of the Czech Republic within the research and development project “Advanced Remediation Technologies and Processes Center” 1M0554 – Program of Research Centers PP2-D01 supported by Ministry of Education.

Contents

1	Overview	3
1.1	Basic usage	4
1.1.1	How to run the simulation.	4
1.1.2	Structure of input	5
2	File formats	7
2.1	Flow123D ini file format	7
2.2	Other input files	15
2.2.1	Mesh file format version 2.0	15
2.2.2	Neighbouring file format, version 1.0	15
2.2.3	Material properties file format, version 1.0	17
2.2.4	Boundary conditions file format, version 1.0	20
2.2.5	Transport boundary conditions file format, version 1.0	22
2.2.6	Element data file format, version 1.0	22
2.3	Output files	24
2.3.1	Output data fields of water flow module	24
2.3.2	Output data fields of transport	24
2.3.3	Auxiliary output files	24
3	Mathematical models	26
3.0.4	Advection-Diffusion equation	26
3.0.5	Generalization	28
3.1	Radioactive Decay and First Order Reactions	29
3.2	General Chemical Reactions	31

Chapter 1

Overview

Flow123D is a software for simulation of water flow and reactionary solute transport in a heterogeneous porous and fractured medium. In particular it is suited for simulation of underground processes in a granite rock massive. The program is able to describe explicitly processes in 3D medium, 2D fractures, and 1D channels and exchange between domains of different dimension. The computational mesh is therefore collection of 3D tetrahedrons, 2D triangles and 1D line segments.

The water flow model assumes a saturated medium described by Darcy law. For discretization, we use lumped mixed-hybrid finite element method. We support both steady and unsteady water flow.

The solute transport model can deal with several dissolved substances. It contains non-equilibrium dual porosity model, i.e. exchange between mobile and immobile pores. There is also model for several types of adsorption in both the mobile and immobile zone. The implemented adsorption models are linear adsorption, Freundlich isotherm and Langmuir isotherm. The solute transport model uses finite volume discretization with up-winding in space and explicit Euler discretization in time. The dual porosity and the adsorption are introduced into transport by operator splitting. The dual porosity model use analytic solution and the non-linear adsorption is solved numerically by the Newton method.

Reaction between transported substances can be modeled either by a SEMCHEM module, which is slow, but can describe all sorts of reactions. On the other hand, for reactions of the first order, i.e. linear reactions or decays, we provide our own solver which is much faster. Reactions are coupled with transport by the operator splitting method,

The program provides output of the pressure, the velocity and the concentration fields in two file formats. You can use file format of GMSH mesh generator and post-processor or you can use output into widely supported VTK format. In particular we recommend Paraview software for visualization and post-processing of VTK data.

The program is implemented in C/C++ using essentially PETSC library for linear algebra. The water flow as well as the transport simulation and reactions can be computed in parallel using MPI environment.

The program is distributed under GNU GPL v. 3 license and is available on the project web page: <http://dev.nti.tul.cz/trac/flow123d>

1.1 Basic usage

1.1.1 How to run the simulation.

On the Linux system the program can be started either directly or through a script `flow123d.sh`. When started directly, e.g. by the command

```
> flow123d -s example.ini
```

the program requires one argument after switch `-s` which is the name of the principal input file. Full list of possible command line arguments is as follows.

-s *file*

Set principal INI input file. All relative paths in the INI file are relative against current directory.

-S *file*

Set principal INI input file. All relative paths in the INI file are relative against directory of the INI file. This is equivalent to change directory to the directory of the INI file at the start of the program.

-i *path*

When there is string `${INPUT}` in the any path in the INI file, it will be replaced by given *path*.

-o *path*

Every relative path for any output file will be relative to this *path*.

All other parameters will be passed to the PETSC library. An advanced user can influence lot of parameters of linear solver. In order to get list of supported options use parameter `-help`.

Alternatively, you can use script `flow123d.sh` to start parallel jobs or limit resources used by the program. This script accepts the same parameters as the program itself and further following additional parameters:

-h

Usage overview.

-t *timeout*

Upper estimate for real running time of the calculation. Kill calculation after *timeout* seconds. Can also be used by PBS to choose appropriate job queue.

-np *number of processes*

Specify number of parallel processes for calculation.

-m *memory limit*

Limits total available memory to *memory limit* bytes.

-n *priority*

Change (lower) priority for the calculation. See `nice` command.

-r *out file*

Stdout and stderr will be redirected to *out file*.

On the Windows system we use Cygwin libraries in order to emulate Linux API. Therefore you have to keep the Cygwin libraries within the same directory as the program executable. The Windows package that can be downloaded from project web page contains both the Cygwin libraries and the `mpiexec` command for starting parallel jobs on the Windows workstations.

Then you can start the sequential run by the command:

```
> flow123d.exe -s example.ini
```

or the parallel run by the command:

```
> mpiexec.exe -np 2 flow123d.exe -s example.ini
```

The program accepts the same parameters as the Linux version, but there is no script similar to `flow123d.sh` for the Windows system.

1.1.2 Structure of input

The principal input file of the program is an INI file which contains names of other necessary input files. Those are the file with calculation mesh (`*.msh`), the file with specification of adjacency between dimensions (`*.ngh`), the file with material description (`*.mtr`) and the file with boundary conditions for the water flow problem (`*.bcd`). For unsteady water flow you have to specify file with initial condition for the pressure (key `Input/Initial`) and optionally one can introduce file with water sources (key `Input/Sources`).

In the case of transport simulation one has to specify also the file with transport boundary conditions (`*.tbc`) and the file with transport initial condition for individual substances (`*.tic`).

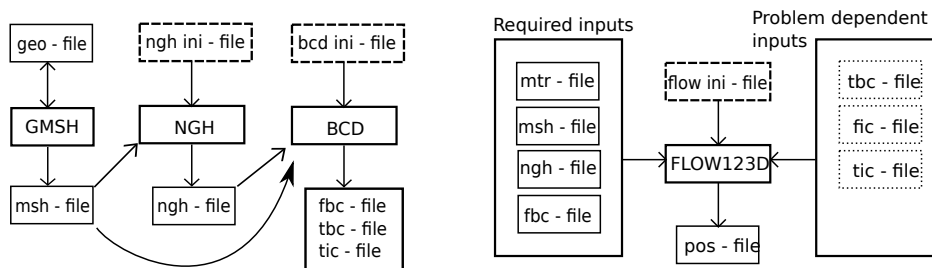


Figure 1.1: Preparation of input files.

For the preparation of input files we use several utilities (see Figure 1.1). We usually begin with a `*.geo` file as a description of the domain geometry. This comes as an input for the GMSH mesh generator, which produces the mesh file. Then we run program `ngh` to produce adjacency file. Finally we can use program `bcd` for the preparation of files with boundary and initial conditions. The file with material properties has to be created

manually, preferably by modifying some of the example problems. The programs `ngh` and `bcd` are distributed together with `flow123d` with their own limited documentation.

The output files can be either `*.msh` files accepted by the GMSH or one can use VTK format that can be post-processed by Paraview.

In the following chapter, we briefly describe structure of individual input files in particular the main INI file. In the last chapter, we describe mathematical models and numerical methods used in the Flow123d.

Chapter 2

File formats

2.1 Flow123D ini file format

In this section we briefly describe every option you can use in global INI-file. Options marked (NOT IMPLEMENTED) are not implemented in current version, but probably will be reimplemented in near future. Options marked (NOT SUPPORTED) has not been tested since they are obsolete and will be soon replaced by similar functionality.

Section: [Global]

KEY	TYPE	DEFAULT	DESCRIPTION
Problem.type	int	NULL	Type of solved problem. Currently supported: 1 = steady saturated flow 2 = unsteady saturated flow using MH method 4 = unsteady saturated flow using Lumped MH method
Description	string	<i>undefined</i>	Short description of solved problem - any text.
Stop_time	double	1.0	Time interval of the whole problem.[time units]
Time_step	double	1.0	Time step for unsteady water flow solver. [time units]
Save_step	double	1.0	The output with transport is written every Save_step . [time units]

Section: **[Input]**

KEY	TYPE	DEFAULT	DESCRIPTION
Mesh	string	NULL	Name of file containig definition of the mesh for the problem. (see Section 2.2.1)
Material	string	NULL	Name of file with hydraulical properties of the elements. (see Section 2.2.3)
Boundary	string	NULL	Name of file with boundary condition data. (see Section 2.2.4)
Neighbouring	string	NULL	Name of file describing topology of the mesh. (see Section 2.2.2)
Initial	string	NULL	Name of file with initial condition for preasure head [length]. (see Section 2.2.6)
Sources	string	NULL	Name of file with definition of fluid sources. This is optional file, if this key is not defined, calculation runs without sources. (see Section 2.2.6)
sources_formula	string	NULL	Expression for sources as function of space coordinates x, y, z . See documentation of FParser library: http://warp.povusers.org/FunctionParser/fparser.html#literals

Section: **[Run]**

KEY	TYPE	DEFAULT	DESCRIPTION
Screen_verbosity	int	0	Nonzero value turn on a more verbose mode (more messages on screen), however everything is in the log file.
Pause_after_run	YES/NO	NO	Wait for Enter after end of progam in order to keep output screen open.

Section: [Transport]

KEY	TYPE	DEFAULT	DESCRIPTION
Transport_on	YES/NO	NO	If set "YES" program compute transport too.
Sorption	YES/NO	NO	If set "YES" program include sorption too.
Dual_porosity	YES/NO	NO	If set "YES" program include dual porosity too.
Reactions	YES/NO	NO	If set "YES" program include reactions too.
Concentration	string	NULL	Name of file with initial condition for concentrations of individual substances. (see Section 2.2.6)
Transport_BCD	string	NULL	Name of file with boundary condition for transport. (see Section 2.2.5)
Sources	string	<i>undefined</i>	Name of file with sources of species for transport. (see Section 2.2.6)
Transport_out	string	NULL	Name of transport output file.
Transport_out_im	string	NULL	(NOT IMPLEMENTED IN 1.6.5) Name of transport immobile output file.
Transport_out_sorp	string	NULL	(NOT IMPLEMENTED IN 1.6.5) Name of transport sorbed output file.
Transport_out_im_sorp	string	NULL	(NOT IMPLEMENTED IN 1.6.5) Name of transport sorbed immobile output file.
N_substances	int	-1	Number of substances.
Substances	string	<i>undefined</i>	Names of the substances separated by commas.
bc_times	list of doubles	NULL	Times for changing boundary conditions. If you set this variable, you have to prepare a separate file with boundary conditions for every time in the list. Filenames for individual time level are formed from BC filename by appending underscore and three digits of time level number, e.g. <code>transport_bcd_000</code> , <code>transport_bcd_001</code> , etc.

Section: [Solver]

KEY	TYPE	DEFAULT	DESCRIPTION
Use_last_solution	YES/NO	NO	If set to "YES", uses last known solution for chosen solver.
Solver_name	string	petsc	Type of linear solver. Supported solvers are: petsc , petsc.matis (experimental)
Solver_params	string	NULL	PETSc options to override default choice of iterative solver and preconditioner (use with care). In particular to use UMFPACK sequential direct solver set: Solve_params = "-ksp preonly -pc_type lu -pc_factor_mat_solver_package umfpack" To use parallel direct solver MUMPS use: Solve_params = "-ksp preonly -pc_type lu -pc_factor_mat_solver_package mumps -mat_mumps_icntl_14 5"
Keep_solver_files	YES/NO	NO	(NOT SUPPROTED IN 1.6.5) If set to "YES", files for solver are not deleted after the run of the solver.
Manual_solver_run	YES/NO	NO	(NOT SUPPROTED IN 1.6.5) If set to "YES", programm stops after writing input files for solver and lets user to run it.
Use_control_file	YES/NO	NO	(NOT SUPPROTED IN 1.6.5) If set to "YES", programm do not create control file for solver, it uses given file.
Control_file	string	NULL	(NOT SUPPROTED IN 1.6.5) Name of control file for situation, when Use_control_file YES.
NSchurs	int	2	Number of Schur complements to use. Valid values are 0,1,2. The last one should be the fastest.
Solver_accuracy	double	1e-6	When to stop solver run - value of residum of matrix. Useful values from 1e-4 to 1e-10. Bigger number = faster run, less accuracy.
max_it	int	200	Maximum number of iteration of linear solver.

Section: **[Output]**

KEY	TYPE	DEFAULT	DESCRIPTION
Write_output_file	YES/NO	NO	If set to "YES", writes output file.
output_piezo_head	YES/NO	NO	If set to "YES", output also piezometric head into flow output files.
Output_file	string	NULL	Name of the output file for water flow output.
Output_file_2	string	NULL	(NOT IMPLEMENTED IN 1.6.5) Name of the output file (type 2).
Output_digits	int	6	(NOT IMPLEMENTED IN 1.6.5) Number of digits used for floating point numbers in output file.
Output_file_type	int	1	(NOT IMPLEMENTED IN 1.6.5) Type of output file 1 - GMSH like format 2 - Flow data file 3 - both files (two separate names)
Pos_format	string	ASCII	Output file format. One can use: ASCII, BIN, or VTK_SERIAL_ASCII
balance_output	string	NULL	Name of file for output of water boundary fluxes and balance of sources over material subdomains.

Section: [Semchem_module]

KEY	TYPE	DEFAULT	DESCRIPTION
Compute_reactions	Yes/No	"No"	NO = transport without chemical reactions YES = transport influenced by chemical reactions
Output_precision	int	1	Number of decimal places written to output file created by Semchem_module.
Number_of_further_species	int	0	Concentrations of these species are not computed, because they are ment to be unexghaustible.
Temperature	double	0.0	Temperature, one of state variables of the system.
Temperature_Gf	double	0.0	Temperature at which Free Gibbs Energy is specified.
Param_Afi	double	0.0	Parameter of the Debay-Hückel equation for activity coefficients computation.
Param_b	double	0.0	Parameter of the Debay-Hückel equation for activity coefficients computation.
Epsilon	double	0.0	Epsilon specifies relative norm of residuum estimate to stop numerical algorithms used by Semchem_module.
Time_steps	int	1	Number of transport step subdivisions for Semchem_module.
Slow_kinetics_substeps	int	0	Number of substeps performed by Runge-Kutta method used for slow kinetics simulation.
Error_norm_type	string	"Absolute"	Through wich kind of norm the error is measured.
Scalling	boolean	"No"	Type of the problem preconditioning for better convergence of numerical method.

Section: **[Aqueous_species]**

KEY	TYPE	DEFAULT	DESCRIPTION
El_charge	int	0	Electric charge of an Aqueous_specie particleunder consideration.
dGf	double	0.0	Free Gibbs Energy valid for TemperatureGf.
dHf	double	0.0	Enthalpy
Molar_mass	double	0.0	Molar mass of Aqueous_species.

Section: **[Further_species]**

KEY	TYPE	DEFAULT	DESCRIPTION
Specie_name	string	""	Name belonging to Further_specie under consideration.
dGf	double	0.0	Free Gibbs Energy valid for TemperatureGf.
dHf	double	0.0	Enthalpy
Molar_mass	double	0.0	Molar mass of Further_species.
Activity	double	0.0	Activity of Further_species.

Section: **[Reaction_i]**

KEY	TYPE	DEFAULT	DESCRIPTION
Reaction_type	string	"unknown"	Type of considered reaction (Equilibrium, Kinetics, Slow_kinetics).
Stoichiometry	int	0	Stoichiometric coefficients of species taking part in <i>i</i> -th reaction.
Kinetic_constant	double	0.0	Kinetic constant for determination of reaction rate.
Order_of_reaction	int	0	Order of kinetic reaction for participating species.
Equilibrium_constant	double	0.0	Equilibrium constant defining <i>i</i> -th reaction.

Section: **[Reaction_module]**

KEY	TYPE	DEFAULT	DESCRIPTION
Compute_decay	Yes/No	"No"	It enables to switch on simulation of radioactive decay or first order reactions.
Nr_of_decay_chains	int	0	How many decay chains are considered.
Nr_of_FoR	int	0	How many first order reactions are defined.

Section: **[Decay_i]**

KEY	TYPE	DEFAULT	DESCRIPTION
Nr_of_isotopes	int	0	It defines the number of isotopes which does the current section [Decay_i] work with.
Substance_ids	array of int	NULL	Sequence of ids describing the order of isotopes in decay chain.
Half_lives	array of double	NULL	Contain half-lives belonging to isotopes defined by ids.
Bifurcation_on	Yes/No	"No"	It makes it possible to define branches in current [Decay_i].
Bifurcation	array of double	NULL	It defines a percentage, which is the first isotope in current [Decay_i] decaying to products.

Section: **[FoReact_i]**

KEY	TYPE	DEFAULT	DESCRIPTION
Kinetic_constant	double	0.0	It defines kinetic constant which is the corresponding half-life computed from.
Substance_ids	array of int	NULL	It contains a couple of indices of substances which are taking part in first order reactions described in current section FoReact.i.

2.2 Other input files

2.2.1 Mesh file format version 2.0

The only supported format for the computational mesh is MSH ASCII format produced by the GMSH software. You can find its documentation on:

<http://geuz.org/gmsh/doc/texinfo/gmsh.html#MSH-ASCII-file-format>

Comments concerning Flow123d:

- Every inconsistency of the file stops the calculation. These are:
 - Existence of nodes with the same *node-number*.
 - Existence of elements with the same *elm-number*.
 - Reference to non-existing node.
 - Reference to non-existing material (see below).
 - Difference between *number-of-nodes* and actual number of lines in nodes' section.
 - Difference between *number-of-elements* and actual number of lines in elements' section.

- By default Flow123d assumes meshes with *number-of-tags* = 3.

tag1 is number of material (reference to .MTR file) in the element.

tag2 is number of geometry region in which the element lies.

tag3 is partition number (CURRENTLY NOT USED).

In accordance with specification of GMSH mesh format.

- Currently, line (*type* = 1), triangle (*type* = 2) and tetrahedron (*type* = 4) are the only supported types of elements. Existence of an element of different type stops the calculation.
- Wherever possible, we use the file extension .MSH. It is not required, but highly recommended.
- This file format can be used also for storing simple discrete scalar or vector fields. We support output into this format (see Section 2.3)

2.2.2 Neighbouring file format, version 1.0

The file is divided in two sections, header and data. The extension .NGH is highly recommended for files of this type.

```
$NeighbourFormat
1.0 file-type data-size
$EndNeighbourFormat
$Neighbours
```

```

number-of-neighbours
neighbour-number type <type-specific-data>
...
$EndNeighbours

```

where

file-type **int** — is equal 0 for the ASCII file format.

data-size **int** — the size of the floating point numbers used in the file. Usually *data-size* = sizeof(double).

number-of-neighbours **int** — Number of neighbouring defined in the file.

neighbour-number **int** — is the number (index) of the n-th neighbouring. These numbers do not have to be given in a consecutive (or even an ordered) way. Each number has to be given only once, multiple definition are treated as inconsistency of the file and cause stopping the calculation.

type **int** — is type of the neighbouring.

<*type-specific-data*> — format of this list depends on the *type*.

Types of neighbouring and their specific data

type = 10 — “Edge with common nodes”, i.e. sides of elements with common nodes.
(Possible many elements)

type = 11 — “Edge with specified sides”, i.e. sides of the edge are explicitly defined.
(Possible many elements)

type = 20 — “Compatible”, i.e. volume of an element with a side of another element.
(Only two elements)

type = 30 — “Non-compatible” i.e. volume of an element with volume of another element. (Only two elements)

<i>type</i>	<i>type-specific-data</i>	Description
10	<i>n_elm eid1 eid2 ...</i>	number of elements and their ids
11	<i>n_sid eid1 sid1 eid2 sid2 ...</i>	number of sides, their elements and local ids
20	<i>eid1 eid2 sid2 coef</i>	Elm 1 has to have lower dimension
30	<i>eid1 eid2 coef</i>	Elm 1 has to have lower dimension

coef is of the **double** type, other variables are **ints**.

Comments concerning Flow123d:

- Every inconsistency or error in the .NGH file causes stopping the calculation. These are especially:
 - Multiple usage of the same *neighbour-number*.

- Difference between *number-of-neighbours* and actual number of data lines.
- Reference to nonexistent element.
- Nonsense number of side.
- The variables *sid?* must be nonnegative and lower than the number of sides of the particular element.

2.2.3 Material properties file format, version 1.0

The file is divided in two sections, header and data. The extension .MTR is highly recommended for files of this type.

```

$MaterialFormat
1.0 file-type data-size
$EndMaterialFormat
$Materials
number-of-materials
material-number material-type <material-type-specific-data> [text]
...
$EndMaterials
$Storativity
material-number <storativity-coefficient> [text]
...
$EndStorativity
$Geometry
material-number geometry-type <geometry-type-specific-coefficient> [text]
...
$EndGeometry
$Sorption
material-number substance-id sorption-type <sorption-type-specific-data> [text]
...
$EndSorption
$SorptionFraction
material-number <sorption-fraction-coefficient> [text]
...
$EndSorptionFraction
$DualPorosity
material-number <mobile-porosity-coefficient> <immobile-porosity-coefficient>
<nonequilibrium-coefficient-substance(0)>
...<nonequilibrium-coefficient-substance(n-1)> [text]
...
$EndDualPorosity
$Reactions
reaction-type <reaction-type-specific-coefficient> [text]
...
$EndReactions

```

where:

file-type **int** — is equal 0 for the ASCII file format.

data-size **int** — the size of the floating point numbers used in the file. Usually *data-size* = sizeof(double).

number-of-materials **int** — Number of materials defined in the file.

material-number **int** — is the number (index) of the n-th material. These numbers do not have to be given in a consecutive (or even an ordered) way. Each number has to be given only once, multiple definition are treated as inconsistency of the file and cause stopping the calculation (exception \$Sorption section).

material-type **int** — is type of the material, see table.

<*material-type-specific-data*> — format of this list depends on the *material - type*.

<*storativity-coefficient*> **double** — coefficient of storativity

geometry-type **int** — type of complement dimension parameter (only for 1D and 2D material), for 1D element is supported type 1 - cross-section area, for 2D element is supported type 2 - thickness.

<*geometry-type-specific-coefficient*> **double** — cross-section for 1D element or thickness for 2D element.

substance-id **int** — refers to number of transported substance, numbering starts on 0.

sorption-type **int** — type 1 - linear sorption isotherm, type 2 - Freundlich sorption isotherm, type 3 - Langmuir sorption isotherm.

<*sorption-type-specific-data*> — format of this list depends on the *sorption - type*, see table.

Note: Section \$Sorption is needed for calculation only if *Sorption* is turned on in the *ini* file.

<*sorption-fraction-coefficient*> **double** — ratio of the "mobile" solid surface in the contact with "mobile" water to the total solid surface (this parameter (section) is needed for calculation only if *Dual_porosity* and *Sorption* is together turned on in the ini file).

<*mobile-porosity-coefficient*> **double** — ratio of the mobile pore volume to the total volume (this parameter is needed only if *Transport_on* is turned on in the ini file).

<*immobile-porosity-coefficient*> **double** — ratio of the immobile pore volume to the total pore volume (this parameter is needed only if *Dual_porosity* is turned on in the ini file).

<*nonequilibrium-coefficient-substance(i)*> **double** — nonequilibrium coefficient for substance *i*, $\forall i \in \langle 0, n - 1 \rangle$ where *n* is number of transported substances (this parameter is needed only if *Dual_porosity* is turned on in the ini file).

reaction-type **int** — type 0 - zero order reaction

$\langle \text{reaction-type-specific-data} \rangle$ — format of this list depends on the *reaction - type*, see table.

<i>material-type</i>	<i>material-type-specific-data</i>	Description
11	k	$\mathbf{K} = (k)$
-11	a	$\mathbf{A} = \mathbf{K}^{-1} = (a)$
21	k	$\mathbf{K} = \begin{pmatrix} k & 0 \\ 0 & k \end{pmatrix}$
22	$k_x \quad k_y$	$\mathbf{K} = \begin{pmatrix} k_x & 0 \\ 0 & k_y \end{pmatrix}$
23	$k_x \quad k_y \quad k_{xy}$	$\mathbf{K} = \begin{pmatrix} k_x & k_{xy} \\ k_{xy} & k_y \end{pmatrix}$
-21	a	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix}$
-22	$a_x \quad a_y$	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a_x & 0 \\ 0 & a_y \end{pmatrix}$
-23	$a_x \quad a_y \quad a_{xy}$	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a_x & a_{xy} \\ a_{xy} & a_y \end{pmatrix}$
31	k	$\mathbf{K} = \begin{pmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & k \end{pmatrix}$
33	$k_x \quad k_y \quad k_z$	$\mathbf{K} = \begin{pmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{pmatrix}$
36	$k_x \quad k_y \quad k_z \quad k_{xy} \quad k_{xz} \quad k_{yz}$	$\mathbf{K} = \begin{pmatrix} k_x & k_{xy} & k_{xz} \\ k_{xy} & k_y & k_{yz} \\ k_{xz} & k_{yz} & k_z \end{pmatrix}$
-31	a	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix}$
-33	$a_x \quad a_y \quad a_z$	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a_x & 0 & 0 \\ 0 & a_y & 0 \\ 0 & 0 & a_z \end{pmatrix}$
-36	$a_x \quad a_y \quad a_z \quad a_{xy} \quad a_{xz} \quad a_{yz}$	$\mathbf{A} = \mathbf{K}^{-1} = \begin{pmatrix} a_x & a_{xy} & a_{xz} \\ a_{xy} & a_y & a_{yz} \\ a_{xz} & a_{yz} & a_z \end{pmatrix}$

Note: all variables ($k, k_x, k_y, k_z, k_{xy}, k_{xz}, k_{yz}, a, a_x, a_y, a_z, a_{xy}, a_{xz}, a_{yz}$) are of the **double** type.

<i>sorption-type</i>	<i>sorption-type-specific-data</i>	Description
1	$k_D[1]$	$s = k_D c$
2	$k_F[(L^{-3} \cdot M^1)^{(1-\alpha)}] \quad \alpha[1]$	$s = k_F c^\alpha$
3	$K_L[L^3 \cdot M^{-1}] \quad s^{max}[L^{-3} \cdot M^1]$	$s = \frac{K_L s^{max} c}{1 + K_L c}$

Note: all variables ($k_D, k_F, \alpha, K_L, s^{max}$) are of the **double** type.

<i>reaction-type</i>	<i>reaction-type-specific-data</i>	Description
0	$\text{substance-id}[1] \quad k[M \cdot L^{-3} \cdot T^{-1}]$	$\frac{\partial c_m^{[\text{substance-id}]}}{\partial t} = k$

Where $c_m^{[\text{substance-id}]}$ is mobile concentration of substance with id *substance-id* and

Δt is the internal transport time step defined by CFL condition.

text **char**[] — is a text description of the material, up to 256 chars. This parameter is optional.

Comments concerning Flow123d:

- If *number-of-materials* differs from actual number of material lines in the file, it stops the calculation.

2.2.4 Boundary conditions file format, version 1.0

The file is divided in two sections, header and data.

```
$BoundaryFormat
1.0 file-type data-size
$EndBoundaryFormat
$BoundaryConditions
number-of-conditions
condition-number type <type-specific-data> where <where-data> number-of-tags
<tags> [text]
...
$EndBoundaryConditions
```

where

file-type **int** — is equal 0 for the ASCII file format.

data-size **int** — the size of the floating point numbers used in the file. Usually *data-size* = sizeof(double).

number-of-conditions **int** — Number of boundary conditions defined in the file.

condition-number **int** — is the number (index) of the n-th boundary condition. These numbers do not have to be given in a consecutive (or even an ordered) way. Each number has to be given only once, multiple definition are treated as inconsistency of the file and cause stopping the calculation.

type **int** — is type of the boundary condition. See below for definitions of the types.

<*type-specific-data*> — format of this list depends on the *type*. See below for specification of the *type-specific-data* for particular types of the boundary conditions.

where **int** — defines the way, how the place for the condition is prescribed. See below for details.

<*where-data*> — format of this list depends on *where* and actually defines the place for the condition. See below for details.

number-of-tags **int** — number of integer tags of the boundary condition. It can be zero.

`< tags > number-of-tags*int` — list of tags of the boundary condition. Values are separated by spaces or tabs. By default we set *number-of-tags*=1, where *tag1* defines group of boundary conditions, "type of water" in our jargon. This can be used to calculate total fluxes through the boundary group.

`[text] char[]` — arbitrary text, description of the fracture, notes, etc., up to 256 chars. This is an optional parameter.

Types of boundary conditions and their data

type = 1 — Boundary condition of the Dirichlet's type

type = 2 — Boundary condition of the Neumann's type

type = 3 — Boundary condition of the Newton's type

<i>type</i>	<i>type-specific-data</i>	Description
1	<i>scalar</i>	Prescribed value of pressure (in meters [m])
2	<i>flux</i>	Prescribed value of flux through the boundary
3	<i>scalar sigma</i>	Scalar value and the σ coefficient

scalar, *flux* and *sigma* are of the `double` type.

Ways of defining the place for the boundary condition

where = 1 — Condition on a node

where = 2 — Condition on a (generalized) side

where = 3 — Condition on side for element with only one external side.

<i>where</i>	<i><where-data></i>	Description
1	<i>node-id</i>	Node id number, according to <code>.MSH</code> file
2	<i>elm-id sid-id</i>	Elm. id number, local number of side
3	<i>elm-id</i>	Elm. id number

The variables *node-id*, *elm-id*, *sid-id* are of the `int` type.

Comments concerning Flow123d:

- We assume homegemous Neumman's condition as the default one. Therefore we do not need to prescribe conditions on the whole boundary.
- If the condition is given on the inner edge, it is treated as an error and stops calculation.
- Any inconsistence in the file stops calculation. (Bad number of conditions, multiple definition of condition, reference to non-existing node, etc.)

- At least one of the conditions has to be of the Dirichlet's or Newton's type. This is well-known fact from the theory of the PDE's.
- Local numbers of sides for *where* = 2 must be lower than the number of sides of the particular element and greater then or equal to zero.
- The element specified for *where* = 3 must have only one external side, otherwise the program stops.

2.2.5 Transport boundary conditions file format, version 1.0

The file is divided in two sections, header and data.

```
$Transport_BCDFormat
1.0 file-type data-size
$EndTransport_BCDFormat
$Transport_BCD
number-of-conditions
transport-condition-number boundary-condition-number value1 value2 ...
$EndTransport_BCD
```

where

file-type **int** - is equal 0 for the ASCII file format.

data-size **int** - the size of the floating point numbers used in the file. Usually *data-size* = sizeof(double)

number-of-conditions **int** - Number of conditions defined in the file.

transport-condition-number **int** - is the number (index) of the n-th transport condition. These numbers do not have to be given in a consecutive (or even an ordered) way. Each number has to be given only once, multiple definition are treated as inconsistency of the file and cause stopping the calculation.

boundary-condition-number **int** - id number of the boundary-condition where transport boundary condition is prescribed.

valueN **double** - prescribed boundary concentration of substance *N* (should be from interval [0, 1]).

Comments concerning FLOW123d: Number of transport boundary conditions has to be same as number of boundary conditions. Program stops computation in the other case.

2.2.6 Element data file format, version 1.0

Several input data fields are given as constant scalars on every element. In particular this is used for water sources, initial condition of pressure, initial condition for concentrations and substance sources in transport. Common file format of these files is:

```

$FieldName
number-of-lines
eid value1 value2 ...
...
$EndFieldName

```

where

\$FieldName — Unique name of the input field. Since all field data are enclosed by **\$FieldName** and **\$EndFieldName** one can even have different fields in one common file.

number-of-sources int — Number of data lines that has to match number of elements in the mesh.

eid int — is id-number of the element (in the input mesh file).

valueN double — list of field values. Number of values is specific for each particular type of input.

Description of individual input fields.

water sources **FieldName=Sources**, there is only one value per line — the density of water source on the element.

pressure initial condition **FieldName=PressureHead**, there is only one value per line — the initial pressure value on the element.

substance sources **FieldName=TransportSources**, number of values is 3 times number of substances. The density of one substance source is given by formula:

$$f = d + \sigma(c - c_N)$$

where f is total source, the first term is fixed Neuman-like source density d . The second term is Newton-like source density, where σ is transmissivity, c is actual concentration, and c_N is prescribed concentration. For every substance there is triplet of three parameters: d , σ , c_N . The order of substances is same as in the main INI file.

concentration initial conditions **FieldName=Concentrations**, number of values equal to number of transported substances, the order of substances is same as in the main INI file.

Comments concerning Flow123d:

- Every inconsistency or error in the .SRC file causes stopping the calculation. These are especially:
 - Difference between *number-of-lines* and actual number of data lines.
 - Reference to nonexisting element.

2.3 Output files

Flow123d support output of scalar and vector data fields into two formats. First one can use native format of program GMSH (usually with extension `msh`) which contains computational mesh and then various datafields for sequence of time levels. For second we support output into XML version of VTK files. These files can be viewed and postprocessed by several visualization softwares. However, our primal goal is to support data transfer into Paraview visualization software. See key `Pos.format`.

2.3.1 Output data fields of water flow module

Water flow module provides output of four data fields.

pressure on elements Pressure head in length units $[L]$ piecewise constant on every element. This field is directly produced by the MH method and thus contains no postprocessing error.

pressure in nodes Same pressure head field, but interpolated into $P1$ continuous scalar field. Namely you lost discontinuities on fractures.

velocity on elements Vector field of water flux volume unit per time unit $[L^3/T]$. For every element we evaluate discrete flux field in barycenter.

piezometric head on elements Piezometric head in length units $[L]$ piecewise constant on every element. This is just pressure on element plus z-coordinate of the barycenter. This field is produced only on demand (see key `output.piezo.head`).

2.3.2 Output data fields of transport

Transport module provides output only for concentrations (in mobile phase) as a field piecewise constant over elements. There is one field for every substance and names of those fields contain names of substances given by key `Substances`. The physical unit is mass unit over volume unit $[M/L^3]$.

2.3.3 Auxiliary output files

Profiling information

On every run we collect some basic profiling informations. After all computations these data are written into the file `profiler%y%m%d_%H.%M.%S.out` where `%y`, `%m`, `%d`, `%H`, `%M`, `%S` are two digit numbers representing year, month, day, hour, minute, and second of the program start time.

Water flux information

File contains water flow balance, total inflow and outflow over boundary segments. Further there is total water income due to sources (if they are present).

Raw water flow data file

You can force Flow123d to write raw data about results of MH method. The file format is:

```
$FlowField
T=<time>
<number of elements>
<eid> <pressure> <flux x> <flux y> <flux z> <number of sides> <pressures on sides> <fluxes on sides>
...
$EndFlowField
```

where

<time> — is simulation time of the raw output.

<number of elements> — is number of elements in mesh, which is same as number of subsequent lines.

<eid> — element id same as in the input mesh.

<flux x,y,z> — components of water flux interpolated to barycenter of the element

<number of sides> — number of sides of the element, influence number of remaining values

<pressures on sides> — for every side average of the pressure over the side

<fluxes on sides> — for every side total flux through the side

Chapter 3

Mathematical models

3.0.4 Advection-Diffusion equation

Solute transport is governed by advection equation which can be written in the form

$$\frac{\partial c}{\partial t} + \mathbf{v} \frac{\partial c}{\partial x} = 0, \quad (3.1)$$

where c is concentration [$M^3 \cdot L^{-3}$], t is time [T], v is velocity [$L \cdot T^{-1}$], and x is coordinate in cartesian system [L]. Assuming solution which is constant on every element (cell centered finite volume method) and integrating equation (3.1) we get

$$\int_{e_i} \frac{\partial c}{\partial t} dV + \int_{e_i} \mathbf{v} \frac{\partial c}{\partial x} dV = 0.$$

After some rearrangements we obtain on i -th element (e_i)

$$\frac{\partial c_i}{\partial t} V_i + c \int_{\partial e_i} \mathbf{v} \cdot \mathbf{dS} = 0, \quad (3.2)$$

where c_i is average concentration in e_i and V_i its volume, c will be specified later (there are two main possibilities - c_i or concentration from neighbouring element). Term $\frac{\partial c}{\partial t}$ we approximate by explicit Euler difference

$$\frac{\partial c}{\partial t} \approx \frac{c_i^{n+1} - c_i^n}{\Delta t}. \quad (3.3)$$

Where Δt is a time step and upper index at c_i means values in the discrete time steps $n+1$ and n . We assume that all elements have piecewise smooth element boundary ∂e with outwards directed normal. Inside the area Ω we introduce internal flows. With respect to e_i , we define internal flow intake U_{ij}^- (from element e_j) and internal flow drain U_{ij}^+ (to element e_j) as follows

$$\begin{aligned} U_{ij}^- &= \min\left(\int_{\partial e_i \cap \partial e_j, i \neq j} \mathbf{v} \cdot \mathbf{dS}, 0\right), \\ U_{ij}^+ &= \max\left(\int_{\partial e_i \cap \partial e_j, i \neq j} \mathbf{v} \cdot \mathbf{dS}, 0\right). \end{aligned} \quad (3.4)$$

Those flows realizes solute transport in the area Ω . On the $\partial\Omega$ we define external flows which will be important for transport Dirichlet boundary conditions. In the same way as for internal flows we assume (with respect to element e_i) external flow intake U_{ij}^{e-} (from $\partial\Omega$) and external flow drain U_{ij}^{e+} (to $\partial\Omega$).

$$\begin{aligned} U_{ik}^{e-} &= \min\left(\int_{\partial e_i \cap \partial\Omega} \mathbf{v} \cdot \mathbf{dS}, 0\right), \\ U_{ik}^{e+} &= \max\left(\int_{\partial e_i \cap \partial\Omega} \mathbf{v} \cdot \mathbf{dS}, 0\right). \end{aligned} \quad (3.5)$$

Direction of the velocity \mathbf{v} , which affects sign of the U -terms is significant for the construction solution. For the solution stability it is suitable to use an upwind scheme, which can be written for finite difference on simple 1D geometry in the form

$$\begin{aligned} v > 0 : \frac{\partial c}{\partial x} &\approx \frac{c_i^n - c_{i-1}^n}{\Delta x}, \\ v < 0 : \frac{\partial c}{\partial x} &\approx \frac{c_{i+1}^n - c_i^n}{\Delta x}. \end{aligned} \quad (3.6)$$

This scheme can be interpreted as well as in finite volume method - in convection term one can get c value opposite the flow of the quantity \mathbf{v} direction. For every e_i we introduce itemsets $\mathcal{N}_i, \mathcal{B}_i$ which contains indexes of neighbouring elements, local boundary conditions respectively. Assuming upwind scheme, using (3.4), (3.5), and (3.3) we can write solution of the equation (3.2) (relation between two consecutive time steps) on e_i in the form

$$c_i^{n+1} = c_i^n - \frac{\Delta t}{V_i} \left[\sum_{j \in \mathcal{N}_i} [U_{ij}^+ c_i + U_{ij}^- c_j] + \sum_{k \in \mathcal{B}_i} [U_{ik}^{e+} c_i + U_{ik}^{e-} c_{B_{ik}}] \right]. \quad (3.7)$$

Where $c_{B_{ik}}$ are values of Dirichlet boundary conditions which belong to e_i . Formula (3.7) can be rewritten into the matrix notation

$$\mathbf{c}^{n+1} = (\mathbf{I} + \Delta t \mathbf{A}) \cdot \mathbf{c}^n + \Delta t \mathbf{B} \cdot \mathbf{c}_B^n \quad (3.8)$$

Where \mathbf{c} is vector of c_i^{n+1} , \mathbf{A} is a square matrix composed from $\frac{U_{ij}^+}{V_i}$, $\frac{U_{ij}^-}{V_i}$, and $\frac{U_{ij}^{e+}}{V_i}$. \mathbf{B} is in general rectangular matrix composed from $\frac{U_{ij}^{e-}}{V_i}$ and \mathbf{c}_B^n is vector of Dirichlet boundary conditions. There is one stability condition for time step which is called Courant-Friedrich-Levy condition. For the problem without sources/sinks it can be written as

$$\Delta t_{max} = \min_i \left(\frac{V_i}{\sum_j U_{ij}^+ + \sum_k U_{ik}^{e+}} \right) = \min_i \left(\frac{V_i}{\sum_j |U_{ij}^-| + \sum_k |U_{ik}^{e-}|} \right). \quad (3.9)$$

This condition has a physical interpretation, which can be understood as conservation law - volume that intakes/drains to/from element e_i can not be higher then element volume V_i . From algebraical point of view this condition can be seen as a condition which bounds norm of the evolution operator as follows

$$\|\mathbf{I} + \Delta t \mathbf{A} \quad \Delta t \mathbf{B}\| \leq 1. \quad (3.10)$$

3.0.5 Generalization

This approach can be used as well as for more general element connections – for compatible/non-compatible element interconnection, if we know the flow integral values (U_{ij}^+ or U_{ij}^-). The most general case of connection is relation among n elements like

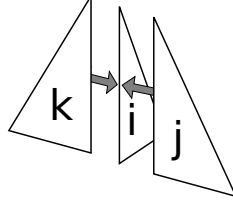


Figure 3.1: Edge with 3 elements

in figure (3.1). For this case we define edge element indexset \mathcal{G}_l that contains all the indexes of elements which sides make l -th edge (g_l), so that $\mathcal{G}_l = \{i, j, k\}$. For \mathcal{G}_l we introduce its subsets \mathcal{G}_{ij} , \mathcal{G}_{ji} , \mathcal{G}_{ik} , \mathcal{G}_{ki} , \mathcal{G}_{kj} , and \mathcal{G}_{jk} , where $\mathcal{G}_{ij} = \mathcal{G}_{ik} = \mathcal{G}_l \setminus i = \{j, k\}$, $\mathcal{G}_{ji} = \mathcal{G}_{jk} = \mathcal{G}_l \setminus j = \{i, k\}$, and $\mathcal{G}_{ki} = \mathcal{G}_{kj} = \mathcal{G}_l \setminus k = \{i, j\}$. It can be written in the same way for any edge g with more than 3 elements, it is hold $|\mathcal{G}_g| - 1 = |\mathcal{G}_{ab}|; \forall a, b \in \mathcal{G}_g$. For l -th edge (g_l) we can define total edge flow U_{g_l} eg. as

$$\begin{aligned} U_{g_l} &= \sum_{m \in \mathcal{G}_{ji}} \left[U_{mj}^+ + \frac{U_{jm}^+}{|\mathcal{G}_{ji}|} \right] = \sum_{m \in \mathcal{G}_{jk}} \left[U_{mj}^+ + \frac{U_{jm}^+}{|\mathcal{G}_{jk}|} \right] \\ &= \sum_{m \in \mathcal{G}_{ij}} \left[U_{mi}^+ + \frac{U_{im}^+}{|\mathcal{G}_{ij}|} \right] = \sum_{m \in \mathcal{G}_{ik}} \left[U_{mi}^+ + \frac{U_{im}^+}{|\mathcal{G}_{ik}|} \right] \\ &= \sum_{m \in \mathcal{G}_{ki}} \left[U_{mk}^+ + \frac{U_{km}^+}{|\mathcal{G}_{ki}|} \right] = \sum_{m \in \mathcal{G}_{kj}} \left[U_{mk}^+ + \frac{U_{km}^+}{|\mathcal{G}_{kj}|} \right], \end{aligned} \quad (3.11)$$

U_{g_l} with respect to any e_m ; $m \in \mathcal{G}_l$ has to have the same value because continuity equation, for assumed incompressible flow, has to be fulfilled in every edge. Edges with more than two elements and two and more nonzero intakes to edge realize an ideal mixing (to an average concentration) with weights which will be specified later. This fact modifies equation (3.7) on the general mesh into the form

$$c_i^{n+1} = c_i^n - \frac{\Delta t}{V_i} \left[\sum_{j \in \mathcal{N}_i} \left[U_{ij}^+ c_i + \frac{U_{ij}^-}{\sum_{k \in \mathcal{G}_{ij}} \left[U_{ki}^+ + \frac{U_{ik}^+}{|\mathcal{G}_{ij}|} \right]} \sum_{k \in \mathcal{G}_{ij}} U_{ki}^+ c_k \right] + \sum_{k \in \mathcal{B}_i} [U_{ik}^{e+} c_i + U_{ik}^{e-} c_{B_{ik}}] \right]. \quad (3.12)$$

The edges with total edge flow $U_{g_l} = 0$ can occur breakdown in the equation (3.12) via term $\sum_{k \in \mathcal{G}_{ij}} \left[U_{ki}^+ + \frac{U_{ik}^+}{|\mathcal{G}_{ij}|} \right] = 0$. This fact implies as well as numerator $U_{ij}^- = 0$. In order to avoid dividing by zero we have to assume computation only for nonzero flows. Concentrations c_k , $k \in \mathcal{G}_{ij}$ that may intakes into element e_i are weighted with weights

$$\alpha_k = \frac{U_{ki}^+}{\sum_{k \in \mathcal{G}_{ij}} \left[U_{ki}^+ + \frac{U_{ik}^+}{|\mathcal{G}_{ij}|} \right]}, \quad (3.13)$$

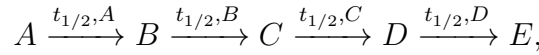
so that the ideal mixing in this edge leads to the average concentration

$$c_{av} = \frac{\sum_{k \in \mathcal{G}_{ij}} U_{ki}^+ c_k}{\sum_{k \in \mathcal{G}_{ij}} \left[U_{ki}^+ + \frac{U_{ik}^+}{|\mathcal{G}_{ij}|} \right]}. \quad (3.14)$$

Matrix notation is the same as in (3.8). Finally ...

3.1 Radioactive Decay and First Order Reactions

Lets consider to have a narrow decay chain without branches. This kind of decay chain can be described by following equation



where letters $\{A, \dots, E\}$ denotes isotopes containet in considered decay chain and $t_{1/2,i}$, $i \in \{A, \dots, E\}$ is a symbol for a halflife of i -th isotope. For a simulation of radioactive decay and first order reactions matrix multiplication based approach has been developed. It has been based on an arrangement of all the data to matrices. The matrix \mathbf{C}^k contains the information about concentrations of all species (s) in all observed elements (e). The upper index k denotes k -th time step. The matrix \mathbf{C}^k has the dimension $e \times s$ (*rows* \times *columns*). The transport simulation is realized by matrix multiplication

$$\mathbf{T} \cdot \mathbf{C}^k = \mathbf{C}^{k+1},$$

where \mathbf{T} is a square, block-diagonal matrix, representing a set of algebraic equations constructed from a set of partial differential equations. When the simulation of the radioactive decay or first order reaction is switched on, one step of simulation changes to

$$\mathbf{T} \cdot \mathbf{C}^k \cdot \mathbf{R} = \mathbf{C}^{k+1},$$

where \mathbf{R} is a square matrix with the dimension $(s \times s)$. It is much easier to construct and to use \mathbf{R} , than to include chemical influence to the transport matrix \mathbf{T} , because the matrix \mathbf{R} has usually a simple structure and s is much smaller than e . In the most simple case, when the order of identification numbers of isotopes in considered decay chain is the same as the order of identifiers of species transported by groundwater, then just two diagonals are engaged and the matrix \mathbf{R} looks as follows:

$$\mathbf{R} = \begin{pmatrix} \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,1}}} & 1 - \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,1}}} & 0 & \dots & \dots & 0 \\ 0 & \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,2}}} & 1 - \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,2}}} & 0 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & 0 & \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,n-2}}} & 1 - \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,n-2}}} & 0 \\ 0 & \dots & \dots & 0 & \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,n-1}}} & 1 - \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,n-1}}} \\ 0 & \dots & \dots & 0 & 0 & 1 \end{pmatrix}$$

Every single j -th column, except the first one, includes the contribution $1 - \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,j}}}$, $j \in \{A, \dots, E\}$ from $(j-1)$ -th isotope with its half-life $t_{1/2,j-1}$. The term $\left(\frac{1}{2}\right)^{\frac{\Delta t}{t_{1/2,j}}}$ describes concentration decrease caused by the radioactive decay of j -th isotope itself. In general

cases the matrix \mathbf{R} can have much more complicated structure, especially when the considered decay chain has more branches. The implementation of the radioactive decay in Flow123D does not firmly include standard natural decay chain. Instead of that it is possible for a user to define his/her decay chain.

It is also possible to simulate decay chains with branches as picture 3.2 shows.

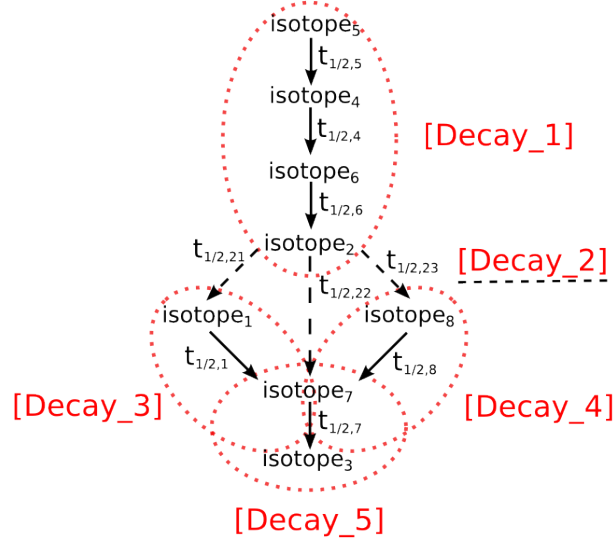
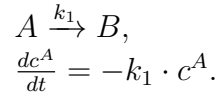


Figure 3.2: Decay chain with branches.

When it comes to a simulation of first order reactions, the kinetic constant is given as an input. The description of a kinetic chemical reaction has obviously two following forms



The first one description is a standard chemical one. The second equation describes temporal decrease in amount of concentrations of the specie c^A . The constant k_1 is so called kinetic constant and for the case of a first order reactions it is equal to so called reaction rate. The order of reaction with just one reactant is equal to the power of c^A in partial differential reaction.

For an inclusion of first order reaction into a reaction matrix a half-life need to be computed from the corresponding kinetic constant k_i . The derivation follows

$$A \xrightarrow{k} B$$

$$\frac{dc^A}{d\tau} = -k \cdot c^A$$

$$\frac{dc^A}{c^A} = -k \cdot d\tau$$

$$\int_{c_0^A/2}^{c_0^A} \frac{dc^A}{c^A} = -k \cdot \int_{t_{1/2,A}}^0 d\tau$$

$$[\ln c^A]_{c_0^A/2}^{c_0^A} = -[k\tau]_{t_{1/2,A}}^0$$

$$\ln c_0^A - \ln \frac{c_0^A}{2} = k \cdot t_{1/2,A}$$

$$\ln 2 = k \cdot t_{1/2,A}$$

$$t_{1/2,A} = \frac{\ln 2}{k_1}$$

The matrix \mathbf{R} is constructed in the same way as for the radioactive decay.

3.2 General Chemical Reactions

For a simulation of general chemical reactions as a part of reactive transport simulation, an application Semchem has been merged together with Flow123D. It enables to simulate following types of reactions:

- Chemical equilibrium (solved using iterative Newtons method)

$$\text{mathematical description } K^{(r)} = \prod_i c_i^{\alpha_i^{(r)}},$$

- Slow evolving chemical kinetics (solved using Runge-Kutta method)

$$\text{mathematical description } \frac{dc_i}{dt} = -k^{(r)} \prod_j c_j^{\beta_j^{(r)}},$$

- Fast evolving chemical kinetics (discretized using implicit Eulerova method and solved using Newtons method)

$$\text{mathematical description } \frac{c_i^{(T+1)} - c_i^T}{\Delta t} = -k^{(r)} \prod_j c_j^{\beta_j^{(r)}, (T+1)},$$

- Radioactive decay can be simulated as a special case of first order reaction.

Mathematical description of general chemical reactions contains often nonlinear terms. That brings the need to solve set of nonlinear algebraic equation in every single time step. It can be very time demanding and that is the reason to look for ways how to reduce number of variables and nonlinear equations somehow. Semchem uses transformation of variables into so called ‘reaction rate’ (ζ^r). ‘Reaction rate’ is an molar concentration (m_i [mol/m³]) of an exhausted reactant divided by appropriate stochiometric coefficient a_i [1].

$$[m_i] = \frac{[n_i]}{[V]} = \frac{[m]}{[M_m][V]} = \frac{[\rho][V]}{[M_m][V]} = \frac{[\rho]}{[M_m]} \cdot [c_i],$$

where m without subscription denotes weight.

Thanks to this approach it is possible to decrease the number of variables to the number of considered chemical reactions.

Further is described how to decrease number of variables in descriptions of kinetic reactions (3.15) and chemical equilibrium (3.21).

$$\frac{dm_j}{dt} = \sum_r k^{(r)} a_j^{(r)} \prod_i m_i^{a_i^{(r)}} \quad (3.15)$$

Using implicit time discretisation of (3.15) we get

$$m_j^{T+1} = m_j^T + \Delta t \sum_r a_j^{(r)} k^{(r)} \prod_i (m_i^{T+1})^{a_i^{(r)}}. \quad (3.16)$$

Through replacement in (3.16) we obtain an equation (3.17).

$$m_j^{T+1} = m_j^T + \sum_r a_j^{(r)} \zeta^{(r)} \quad (3.17)$$

By a substitution of (3.17) into (3.16), an equation (3.18) is created.

$$m_j^T + \sum_s a_j^{(s)} \zeta^{(s)} = m_j^T + \Delta t \sum_r a_j^{(r)} k^{(r)} \prod_i \left(m_i^T + \sum_R a_i^{(R)} \zeta^{(R)} \right)^{a_i^{(r)}} \quad (3.18)$$

Substraction of m_j^T from both sides of previous equation leads to a set of nonlinear equations (3.19) with just as many variables as considered chemical reactions identified by indeces (r) .

$$\zeta^{(s)} = \Delta t \cdot k^{(r)} \prod_i \left(m_i^T + \sum_R a_i^{(R)} \zeta^{(R)} \right)^{a_i^{(r)}} \quad (3.19)$$

(3.19) can be modiflicated to (3.20)

$$\zeta^{(s)} \prod_i \left(m_i^T + \sum_r a_i^{(r)} \zeta^{(r)} \right)^{-a_i^{(r)}} = \Delta t \cdot k^{(r)} \quad (3.20)$$

which needs to be solved.

Consider to have N species taking part in chemical reactions. For chemical equilibriums the equation (3.21) is obviously used for description. Letter X_i denotes an activity of i -th chemical specie.

$$K^{(r)} = \prod_{i=1}^N X_i^{(r)} \quad (3.21)$$