RW3D-Rx

User Guide

Daniel Fernàndez-Garcia and Christopher V. Henri

Grup d'Hidrologia Subterrània
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Universitat Politècnica de Catalunya

Department of Civil and Environmental Engineering
Associated Unit: Hydrogeology Group (UPC-CSIC)
C/ Jordi Girona 1-3, Ed. D2/004 08034 Barcelona, Spain

contact: christopher.henri@upc.edu

last update: *July 12, 2017*

Contents

Related Publications

Conceptual development

Fernàndez-Garcia, D., Illangasekare, T. H., and Rajaram, H. (2005), Differences in the scale-dependence of dispersivity estimated from temporal and spatial moments in physically and chemically heterogeneous porous media, *Advances in Water Resources* (ISSN 0309-1708), 28, 745-759.

Salamon, P., Fernàndez-Garcia, D., and J. J. Gómez-Hernández (2006), A review and numerical assessment of the random walk particle tracking method, *Journal of Contaminant Hydrology* (ISSN 01697722), 86, 277-305.

Salamon, P., Fernàndez-Garcia, D., and J. J. Gómez-Hernández (2006), Modeling mass transfer processes using random walk particle tracking, *Water Resources Research* (ISSN 0043-1397), 42, W11417.

Henri, C.V., and Fernàndez-Garcia, D. (2014), Toward efficiency in heterogeneous multispecies reactive transport modeling: A particle-tracking solution for first-order network reactions, *Water Resources Research*, 50, 7206-7230.

Henri, C.V., and Fernàndez-Garcia, D. (2015), A random walk solution for modeling solute transport with network reactions and multi-rate mass transfer in heterogeneous systems: Impact of biofilms, *Advances in Water Resources*, 86, Part A, 119-132.

Benson, D.A., and Meerschaert, M.M. (2008), Simulation of chemical reaction via particle tracking: Diffusion-limited versus thermodynamic rate-limited regimes, *Water Resources Research*, 44, W12201.

Field applications

Salamon, P., Fernàndez-Garcia, D., J. J. Gómez-Hernández (2007), Modeling tracer transport at the MADE site: The importance of heterogeneity, *Water Resour. Res.* (ISSN 0043-1397), 43, W08404.

Riva, M., A. Guadagnini, D. Fernàndez-Garcia, X. Sánchez-Vila, (2008), Relative importance of geostatistical and transport models in describing heavily tailed breakthrough curves at the Lauswiesen site, *Journal of Contaminant Hydrology* (ISSN 01697722), 101, 1-13, 2008.

Name file	File with names for output files (by default: `rw3d.nam`)
Parameter file	File with parameters (to be defined in the name file)

Those are the only two mandatory input files for RW3D to run. The code reads these files line by line. We describe in the following sections the

A Name File

LINE	VARIABLE	DESCRIPTION
1	Text	
2	Text	
3	Text	
4	Text	
5	File name	Parameter file
6	Text	
7	Text	
8	Text	
9	File name	Output histogram (pdf) of particle arrival times (btcs)
10	File name	Output with cumulative pdf particle arrival times (cbtcs)
11	File name	Output with particle snapshots with time
12	File name	Output with particle paths
13	File name	Output with cartesian spatial moments
14	File name	Output with spatial moments of particle position
15	File name	Output with particle position at control planes
16	File name	Output with dilution index of kitanidis (option disable)
17	File name	Output with radial spatial moments
18	File name	Output with temporal moments of breakthrough curves
19	File name	Output with dispersivities from control planes breakthrus
20	File name	Output with residence times in zonal regions
21	File name	Output with velocity field (for idebug ≥ 1)
22	File name	Output with derivative of BTC in double log

B Parameter File

B.1 General Options

LINE	VARIABLE	DESCRIPTION
1-9	Text	
10	idebug	*idebug*: Integer defining degree of debugging as written in `rw3d_general.dbg` OPTIONS: • idebug > 0 $\rightarrow$ write $\alpha_L, \alpha_{TH}, \alpha_{TV}$ arrays • idebug > 2 $\rightarrow$ write q_x, q_y, q_z in gslib format
11	nspe$_{aq}$,nspe$_{min}$	Number of aqueous and mineral species
12	name$_{aq}$	Name of aqueous species (empty line if nspe$_{aq}$=0)
13	name$_{min}$	Name of mineral species (empty line if nspe$_{min}$=0)
14	tsim	Simulation time

B.2 Geometry

LINE	VARIABLE	DESCRIPTION
15	Text	
16	Text	
17	Text	
18	nx, ny, nz	Number of cells in x,y,z directions
19	dx [*read as a real array*]	Cell size in x-direction
20	dy [*read as a real array*]	Cell size in y-direction
21	dz [*read as a real array*]	Cell size in z-direction

COMMENT:
• dz is not allowed to vary arbitrarily in x or y, i.e. same cell thickness on a given layer; 3D cartesian grid will be soon implemented.
• if an input file is used to defined dx, dy or dz, use $ny = 1$ and $nz = 1$ to define the dx gslib array. Same for dy ($nx = 1, nz = 1$) and dz ($nx = 1, ny = 1$).

22	file, const, ivar, flag	*file*: file name with array defining inactive cells (IBOUND variable) *const*: after reading the array, all values are multiplied by *const* *ivar*: column variable in the GSLIB file *flag*: integer defining the way to read the array

OPTIONS:
• flag = 0: The array is not read from file.
• flag = 1: The array is read from file named FILE. This file has GSLIB format such that:
IBOUND $\neq$ 0: active cell
IBOUND = 0: inactive cell $\rightarrow$ particles reflect at cell boundaries
• flag = 2: Option specific for reading IBOUND from MODFLOW external file format.
• flag = 3: Specific for IBOUND array. This reads specific inactive cells from the external file named *file*. The format of this external file is:
1. number of inactive cells [Integer]
for each inactive cell:
2. column, row, layer (in GSLIB format)

23	ibx1, ibx2, iby1, iby2, ibz1, ibz2	Boundary conditions (at left, right, front, back, bottom, top sides of the cuboid model): ib = 0 for flux boundary condition ib = 1 for impermeable boundary condition

B.3 Time Discretization

LINE	VARIABLE	DESCRIPTION
24	Text	
25	Text	
26	Text	
27	string	Method to calculate the time step used to move all particles of the plume at a given time.

OPTIONS:

- string = CONSTANT_DT or CONSTANT_TIME: The time step Δt is fixed: standard random walk

- string = CONSTANT_CU: The time step is estimated from

$$\Delta \text{t} = \text{Cu} \times tc_{\text{ADV}},$$

where Cu is the grid-Curant number (given in next line), and tc_{ADV} is the advective characteristic time, i.e.:

$$tc_{\text{ADV}} = \frac{\Delta s}{\min\limits_{v_1,v_2,v_3} \{v_i/R\}},$$

where Δs is the characteristic size of a cell, v_i is the particle velocity component in the ith direction, and R is the retardation coefficient associated.

- string = CONSTANT_DaMT: The time step is estimated from

$$\Delta \text{t} = \text{Da}_{\text{MT}} \times tc_{\text{MT}}$$

where Da_{MT} is the grid-Damkholer number based on Mass Transfer process (given in next line), and tc_{MT} is the characteristic time for mass transfer, i.e.:

$$tc_{\text{MT}} = \max \frac{1}{\alpha_k \times \beta_k} \quad \forall k = 1, \ldots, N_{im},$$

where α_k and β_k are respectively the mass transfer coefficient and the field capacity coefficient associated to the kth immobile domain, and N_{im} is the number of immobile zones.

- string = CONSTANT_DaDECAY: The time step is estimated from:

$$\Delta \text{t} = \text{Da}_{\text{DECAY}} \times tc_{\text{K}},$$

where Da_{DECAY} is the grid-Damkholer number based on first-order decay process (given in next line), and tc_{K} is the characteristic time for first-order decay, i.e. $tc_{\text{K}} = \frac{R}{k}$ where k is the first-order decay rate.

- string = OPTIMUM_DT: Estimates the time step from grid Curant number, Pecklet number and Damkholer number based on kinetic reaction; and pick the more restrictive one, i.e.

$$\Delta \text{t} = \min\{\ \text{Cu} \times tc_{\text{ADV}}, \text{Pe} \times tc_{\text{DISP}}, \text{Da}_{\text{KINETIC}} \times tc_{\text{KINETIC}}\ \},$$

where T_{esc} is the advective travel time of the particle in the cell.

COMMENT:

- all characteristic times are calculated for each particle of the plume and the more restrictive ones are chosen to compute the time step.

28	Δt, Cu, Pe, $Da_{KINETIC}$ Da_{DECAY}, Da_{MT}	Δt $\rightarrow$ fixed time step Cu $\rightarrow$ grid-Courant number Pe $\rightarrow$ grid-Pecklet number Da_{DECAY} $\rightarrow$ grid-Damkholer number for first-order decay process $Da_{KINETIC}$ $\rightarrow$ grid-Damkholer number for kinetic reaction Da_{MT} $\rightarrow$ grid-Damkholer number for mass Transfer process

B.4 Advection Package

LINE	VARIABLE	DESCRIPTION
29	Text	
30	Text	
31	Text	
32	Logical Flag	True (T) if package is active
33	method	Character variable specifying computation of the advection displacement of a particle.

OPTIONS:

- method = EULERIAN: Standard Random Walk with Eulerian integration of the velocity:

$$\Delta \mathbf{X}_{p,adv} = \int v(\tau) d\tau \approx v(\mathbf{X}_p, t)\Delta t,$$

where $\mathbf{X}_{p,adv}$ is the advective motion of a particle, and v is the pore velocity.

- method = EXPONENTIAL: Pollock Method to integrate the velocity from finite-difference flow models:

$$\Delta \mathbf{X}_{p,adv} = \int v(\tau) d\tau \approx \frac{v_i(\mathbf{X}_p, t)}{A_i\, R}(\exp(A_i\, \Delta t) - 1), \text{with}$$

$$A_i = \frac{v_{i,face(2)} - v_{i,face(1)}}{\Delta x_i}.$$

34	file, const, ivar, flag	*file*: file name with array of cell-by-cell Darcy velocity in the x-direction, q_x *const*: after reading the array, all values are multiplied by *const* *ivar*: column variable in the GSLIB file *flag*: integer defining the way to read the array

OPTIONS:

- flag = 0: The array is not read from file: q_x is defined by *const*.
- flag = 1: The array is read from file named FILE. This file is in GSLIB format.
- flag = 2: Use Modflow binary cell-by-cell flux file (output of virtually any Modflow versions should be accepted; contact us if you are experiencing issues).

35	file, const, ivar, flag	Darcy velocity in the y-direction, q_y: follow same instructions than for q_x (line 34)
36	file, const, ivar, flag	Darcy velocity in the z-direction, q_z: follow same instructions than for q_x (line 34)
37	poro [*read as a real array*]	Porosity for the Mobile Zone
38	NPER	Number of velocity stress period (as defined in Modflow if binary Modflow output file is used)

for each stress period:		
39	PERLEN, NSTP, TSMULT, SS/TR	PERLEN: length period NSTP: number of time step TSMULT: multiplier SS/TR: specify either SS for steady state or TR for transient
40	Logical Flag	True (T) if the MF fluxes are looped until tsim

B.5 Dispersion Package

LINE	VARIABLE	DESCRIPTION
41	Text	
42	Text	
43	Text	
44	Logical Flag	True (T) if package is active
45	a_L [*read as a real array*]	Longitudinal dispersivity
46	a_{TH} [*read as a real array*]	Transverse dispersivity in horizontal plane
47	a_{TV} [*read as a real array*]	Transverse dispersivity in vertical plane
48	Dm_L [*read as a real array*]	Longitudinal molecular diffusion
49	Dm_{TH} [*read as a real array*]	Transverse molecular diffusion in horizontal plane
50	Dm_{TV} [*read as a real array*]	Transverse molecular diffusion in vertical plane
51	MULTa	Multiplier of dispersivity coefficients for each species (on a single line)
52	MULTD	Multiplier of diffusion coefficients for each species (on a single line)

B.6 Mass Transfer Package

LINE	VARIABLE	DESCRIPTION
53	Text	
54	Text	
55	Logical Flag	True (T) if package is active
56	Model	Type of mass transfer model

OPTIONS:

- model = MULTIRATE: discrete series of mass transfer rates

How to define input parameters?		
57	N_{im}	number of immobile zones
for each zone:		
58	$\phi_{k=1}$ [*read as a real array*]	porosity in the 1^{st} immobile zone
59	$\alpha'_{k=1}$ [*read as a real array*]	first-order mass transfer rate associated with the 1^{st} immobile zone
⋮	ϕ_k [*read as a real array*]	porosity in the kth immobile zone
⋮	α'_k [*read as a real array*]	first-order mass transfer rate associated with the kth immobile zone

- model = SPHERICAL_DIFFUSION or LAYERED_DIFFUSION or CYLINDRICAL_DIFFUSION: diffusion geometry

How to define input parameters?

57	N_{im}	number of immobile zones
58	ϕ_{im} [*read as a real array*]	porosity in the immobile domain
59	D_p/a^2 [*read as a real array*]	effective pore diffusion coefficient, related to apparent pore diffusion coefficient D_a/a^2 by $D_a = D_p/R_{im}$. D_p $[L^2/T]$ is the diffusivity coefficient, a^2 $[L^2]$ is the radius of the blocks. The multirate series for diffusion models is given in Table 1.

- model = POWER_LAW: power law memory function

How to define input parameters?

57	N_{im}	number of immobile zones
58	β_{tot} [*read as a real array*]	total capacity ratio of all immobile zones
59	A_{min} [*read as a real array*]	minimum apparent mass transfer coefficient
60	A_{max} [*read as a real array*]	maximum apparent mass transfer coefficient
61	$power$ [*read as a real array*]	exponent of the power law density function of first-order rate coefficient

- model = LOGNORMAL_LAW: lognormal law memory function

How to define input parameters?

57	N_{im}	number of immobile zones
58	β_{tot} [*read as a real array*]	total capacity ratio of all immobile zones
59	$mean$ [*read as a real array*]	mean of the natural log of mass transfer coefficient
60	$stdv$ [*read as a real array*]	standard deviation of the natural log of mass transfer coefficient

- model = COMPOSITE_MEDIA: mixture of geometries

How to define input parameters?

57	$N_{mrate}, N_{sph}, N_{cyl}, N_{lay}$	number of immobile zones for the multirate model (N_{mrate}), the spherical diff. model (N_{sph}), the cylindrical diff. model (N_{cyl}) and the cylindrical diff. model (N_{lay})

for each zone of each mass transfer model:

58	$Fmrate_1$	fraction of the 1st zone associated to the multirate model
...	$Fmrate_k$	fraction of the kth zone associated to the multirate model
59	$Fsph_1$	fraction of the 1st zone associated to the sph. diff. model
...	$Fsph_k$	fraction of the kth zone associated to the sph. diff. model
60	Fcy_1	fraction of the 1st zone associated to the cyl. diff. model
...	Fcy_k	fraction of the kth zone associated to the cyl. diff. model
61	$Flay_1$	fraction of the 1st zone associated to the lay. diff. model
...	$Flay_k$	fraction of the kth zone associated to the lay. diff. model

parameters for the multirate model:

62	$\phi_{k=1}$ [*read as a real array*]	porosity in the 1st imm. zone
63	$\alpha'_{k=1}$ [*read as a real array*]	first-order mass transfer rate associated with the 1st imm. zone
⋮	ϕ_k [*read as a real array*]	porosity in the k^{th} imm. zone
	α'_k [*read as a real array*]	first-order mass transfer rate associated with the k^{th} imm. zone

parameters for the sph. diff model:

64	ϕ_{im} [*read as a real array*]	porosity in the immobile domain
65	D_p/a^2 [*read as a real array*]	effective pore diffusion coefficient

parameters for the cyl. model:

66	ϕ_{im} [*read as a real array*]	porosity in the immobile domain
67	D_p/a^2 [*read as a real array*]	effective pore diffusion coefficient

parameters for the lay. model:

68	ϕ_{im} [*read as a real array*]	porosity in the immobile domain
69	D_p/a^2 [*read as a real array*]	effective pore diffusion coefficient

---

COMMENT:

- it is required to specify a model type and associated parameters even if the logical flag is *False*
- if the logical flag is *False*, define 0 immobile zones to avoid specifying parameters; if the number of immobile zone is larger than 0, define parameters accordingly.

Table 1: Multirate series for diffusion

Diffusion geometry	Multirate series [a] for j=1,…,N_{im}-1 α_j	β_j	for j=N_{im} α_j	β_j
Layered diffusion	$\frac{(2j-1)^2\pi^2}{4}(D_a/a^2)_i$	$\frac{8}{(2j-1)^2\pi^2}\beta_{tot}$	$\frac{3\left(D_a/a^2\right)_i\left[1-\sum_{j=1}^{N_{im}-1}\frac{8}{(2j-1)^2\pi^2}\right]}{1-\sum_{j=1}^{N_{im}-1}\frac{96}{(2j-1)^4\pi^4}}$	$\left[1-\sum_{j=1}^{N_{im}-1}\frac{8}{(2j-1)^2\pi^2}\right]\beta_{tot}$
Cylindrical diffusion[a]	$r_{0,j}^2(D_a/a^2)_i$	$\frac{4}{r_{0,j}^2}\beta_{tot}$	$\frac{8\left(D_a/a^2\right)_i\left[1-\sum_{j=1}^{N_{im}-1}\frac{4}{r_{0,j}^2}\right]}{1-\sum_{j=1}^{N_{im}-1}\frac{32}{r_{0,j}^2}}$	$\left[1-\sum_{j=1}^{N_{im}-1}\frac{4}{r_{0,j}^2}\right]\beta_{tot}$
Spherical diffusion[b]	$j^2\pi^2(D_a/a^2)_i$	$\frac{6}{j^2\pi^2}\beta_{tot}$	$\frac{15\left(D_a/a^2\right)_i\left[1-\sum_{j=1}^{N_{im}-1}\frac{6}{j^2\pi^2}\right]}{1-\sum_{j=1}^{N_{im}-1}\frac{90}{j^4\pi^4}}$	$\left[1-\sum_{j=1}^{N_{im}-1}\frac{6}{j^2\pi^2}\right]\beta_{tot}$

[a] Where $r_{0,j}$ is the jth root of $J_0(x)$ where J_0 is the zero-order Bessel function of the first kind.

[b] Where $(\beta_{tot})_i = \frac{\phi_{im}\,R_i^{im}}{\phi_m\,R_i^m}$ is the capacity ratio for a specie i.

B.7 Reaction Package

LINE	VARIABLE	DESCRIPTION
61	Text	
62	Text	
63	Text	

B.7.1 Sortion

63	Text	
64	Text	
65	Logical Flag	True (T) if package is active
66	model	Type of sorption

OPTIONS:

- model = LINEAR: instantaneous linear sorption isotherm (retardation)

for each species:

67	$R_{i=1}$ [*read as a real array*]	retardation coefficient for the 1^{st} species
…	R_i [*read as a real array*]	retardation coefficient for the i^{th} species

IF MASS TRANSFER:

mass transfer type = MULTIRATE

for each species, for each zone:

68	$Rim_{i=1,k=1}$ [*read as a real array*]	retardation for the 1^{st} species in the 1^{st} immobile zone
…	$Rim_{i=1,k}$ [*read as a real array*]	retardation for the 1^{st} species in the k^{th} immobile zone
69	$Rim_{i,k=1}$ [*read as a real array*]	retardation for the i^{th} species in the 1^{st} immobile zone
…	$Rim_{i,k}$ [*read as a real array*]	retardation for the i^{th} species in the k^{th} immobile zone

mass transfer type = _DIFFUSION or POWER_LAW or LOGNORMAL_LAW:

for each species:

70	$Rim_{i=1}$ [*read as a real array*]	retardation for the 1^{st} species in the immobile domain
…	Rim_i [*read as a real array*]	retardation for the i^{th} species in the immobile domain

- model = CHTM: linear sorption solved by Contineous History Time Method

only available for a single species and no mass transfer

67	bd [*read as a real array*]	bulk density
68	kf [*read as a real array*]	forward mass transfer coefficient
69	kb [*read as a real array*]	backward mass transfer coefficient

B.7.2 First-order decay reaction network

68	Text	
69	Text	
70	Text	
71	Logical Flag	True (T) if package is active
72	model	Type of network reaction

OPTIONS:
- model = SERIAL: for a serial reaction network: A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ . . .
- model = SERIAL_MOMENT: for a serial reaction network with motion solved by calculating the first and second spacial moments
- model = GENERIC: for a generic network reaction

for each species:

73	$k_{i=1}$ [*read as a real array*]	first-order decay rate for the 1st species
…	k_i [*read as a real array*]	first-order decay rate for the i^{th} species
74	$y_{i=1}$ [*read as a real array*]	yield coefficient for the 1st species
…	y_i [*read as a real array*]	yield coefficient for the i^{th} species

IF MASS TRANSFER:
mass transfer type = MULTIRATE
for each species, for each zone:

75	$kim_{i=1,k=1}$ [*read as a real array*]	first-order decay rate for the 1st species in the 1st immobile zone
…	$kim_{i=1,k}$ [*read as a real array*]	first-order decay rate for the 1st species in the kth immobile zone
76	$kim_{i,k=1}$ [*read as a real array*]	first-order decay rate for the ith species in the 1st immobile zone
…	$kim_{i,k}$ [*read as a real array*]	first-order decay rate for the ith species in the kth immobile zone

mass transfer type = _DIFFUSION or POWER_LAW or LOGNORMAL_LAW:
for each species:

75	$kim_{i=1}$ [*read as a real array*]	first-order decay rate for the 1st species in the immobile domain
…	kim_i [*read as a real array*]	first-order decay rate for the ith species in the immobile domain

---

B.7.3 Bimolecular reactions

68	Text	
69	Text	
70	Text	
71	Logical Flag	True (T) if package is active
72	NREACT	Number of chemical reactions

for each reaction:

73	[REAC + REAC + ... –> PROD + PROD + ...]	Description of the reaction

for each reaction:

74	K_f [*read as a real array*]	Reaction rate

COMMENT:
- A reaction is always described between brackets;
- The name of the reactants and products must correspond to the name given in line 12 and 13;

B.8 Control Surfaces Package

LINE	VARIABLE	DESCRIPTION
73	Text	
74	Text	
75	Text	
76	N_{well}	Number of wells
for each well:		
77	WellID, Xwell, Ywell, Rwell, Zbot, Ztop, Flag, Save	*WellID*: Name given to the well *Xwell*, *Ywell*: X, Y well coordinates *Rwell*: well radius *Zbot*: well bottom (z coordinate) *Ztop*: well top (z coordinate) *Flag*: integer defining particle behavior after passing thru the well OPTIONS: • flag = 0 → The particle passes thru the well but does not exit the system • flag = 1 → The particle exits the system when crosses the well *Save*: True (T) is (C)BTC is saved and printed
78	N_{plane}	Number of control plane
for each plane:		

OPTIONS:

- For planes perpendicular or parallel to axes

67	Dist, Type, Flag	*Dist*: Distance of the control plane with respect to the x,y or z coordinate axis *Type*: type of plane which can be: • type = XX → plane parallel to the x coordinate • type = YY → plane parallel to the y coordinate • type = ZZ → plane parallel to the z coordinate *Flag*: Integer that can be: • flag = 0 → The particle passes through the well but does not exit the system • flag = 1 → The particle exits the system when crosses the well

- For planes oriented in any direction. The plane is described by the equation of a plane:

$$Ax + By + Cz + D = 0$$

67	A, B, C, D, Flag	*Flag*: Integer that can be: • flag = 0 → The particle passes thru the well but does not exit the system • flag = 1 → The particle exits the system when crosses the well

B.9 Injections of Particles

LINE	VARIABLE	DESCRIPTION
68	Text	
69	Text	
70	Text	
71	N_{inj}	Number of injections NOTE: This version considers the injections to be independent from one another. Another approach is to use multiple injections to more complicated initial conditions: particles associated with different geometries, zones or species. This can easily be accommodated if requested.
for each injection:		
72	NameInj, TypeInj	Name and type of the injection

OPTIONS FOR NameInj:

- NameInj = POINT

point injection → all particles start the simulation from the same point position.

73	Pmass, Zone, Specie	*Pmass*: Mass of a single particle *Zone*: Zone which the particles belongs initially (0 = mobile) *Specie*: Specie which the particles belongs initially
74	xinj, yinj, zinj	x, y, z point coordinates

- NameInj = LINE

vertical line injection → particles randomly uniformly distributed in a vertical line.

73	Pmass, Zone, Specie	*see point injection*
74	xinj, yinj, zbot, ztop	*xinj*, *yinj*: x, y coordinates vertical line *zbot*: z line bottom vertical position *ztop*: z line top vertical position

- NameInj = BLOCK

block injection → particles uniformly distributed, equidistantly, in a block defined by lower grid-cell and upper grid-cell index.

73	Pmass, Zone, Specie	*see point injection*
74	idwn, jdwn, kdwn, iup, jup, kup	*idwn*, *jdwn*, *kdwn*: lower left cell number in x, y, z direction *iup*, *jup*, *kup*: top right cell number in x, y, z direction

- NameInj = CIRCLE

circle injection → particles uniformly distributed (randomly) within a vertical cylinder.

73	Pmass, Zone, Specie	*see point injection*
74	x0, y0, zbot, ztop, rcyr	*x0*, *y0*: coordinates origin cylinder *zbot*: z bottom position cylinder *ztop*: z top position cylinder *rcyr*: cylinder radius

- NameInj = RADIAL

radial injection → particles uniformly distributed (randomly) on the surface of a vertical cylinder.

73	Pmass, Zone, Specie	*see point injection*
74	xinj, yinj, zbot, ztop, rcp	*xinj*, *yinj*: coordinates origin cylinder *zbot*: z bottom position cylinder *ztop*: z top position cylinder *rcp*: cylinder radius

- NameInj = PLANE

plane injection → particles uniformly distributed in a vertical plane perpendicular to x direction (not random, equidistant points).

73	Pmass, Zone, Specie	*see point injection*
74	xdist, width, height	*xdist*: x position of the vertical plane *width*: width of the plane in the y direction *height*: height of the plane in the z direction

- NameInj = PLANE_RANDOM

plane injection random → particles uniformly distributed (randomly) in a vertical plane perpendicular to x direction.

73	Pmass, Zone, Specie	*see point injection*
74	xdist, width, height	*see plane injection*

- NameInj = PLANE_FLUX_WEIGHTED

plane injection random → particles in a vertical plane perpendicular to x direction, and proportionally to the Darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	xdist, width, height	*see plane injection*

- NameInj = LINE_BY_POINTS

line injection by points → particles distributed uniformly (not random) in a line specified by points.

73	Pmass, Zone, Specie	*see point injection*
74	x1, y1, z1, x2, y2, z2	*x1*, *y1*, *z1*: x,y,z coordinates of the first point *x2*, *y2*, *z2*: x,y,z coordinates of the second point

- NameInj = LINE_FLUX_WEIGHTED

line flux weighted → particles are distributed proportional to the darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	x1, y1, z1, x2, y2, z2	*see line_by_points injection*

- NameInj = LINE_BY_POINT_RANDOM

line by point (random) → particles are distributed randomly.

73	Pmass, Zone, Specie	*see point injection*
74	x1, y1, z1, x2, y2, z2	*see line_by_points injection*

- NameInj = VERTICAL_LINE_FLUX_WEIGHTED

vertical line flux weighted → particles distributed on a vertical line and proportionally to the Darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	idwn,jdwn,kdwn,kup	*idwn*, *jdwn*: cell x- and y-index (the injection line is located at the center of the cell) *kdwn*, *kup*: bottom cell and upper cell z-index

- NameInj = READ_PARTICLE_FILE

read particle file → particles location is read from an external file.

73	filename	*filename*: name of the ascii file containing the particles location

COMMENT:
- Only for Dirac injection;
- File format:

(I1) Header
(I2) Number of particles
for each particle:
(I3) x,y,z,Pmass,Zone,Specie: particle location (x,y,z), see point injection for basic particle properties

- NameInj = READ_CONCENTRATION_FILE

read concentration file → cell-by-cell concentrations are read from an external file.

73	Pmass, Zone, Specie	*see point injection*
74	filename, const	*filename*: name of the ascii file containing the cell-by-cell concentration *const*: multiplier of concentration read in *filename*

COMMENT:
- Only for Dirac injection;
- Particle location generate randomly within a given grid-cell;
- File format:

(I1) Header
(I2 to ny*nz+1) concentration matrix in MODFLOW format (reversed y and z axis).

- NameInj = VERTICAL_BLOCK_FLUX_WEIGHTED

vertical block flux weighted → particles injected in a given cells of a vertical block (single i and j cell index), and proportionally to the Darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	idwn,jdwn,kdwn,kup	*idwn*, *jdwn*: cell x- and y- index *kdwn*, *kup*: bottom cell and upper cell z-index

- NameInj = CELLS_FILE

cells file → particles randomly injected in given cells.

73	Pmass, Zone, Specie	*see point injection*
74	filename	*filename*: name of the ascii file containing the cells where particles are injected.

COMMENT:
- Particle location generate randomly within a given grid-cell;
- File format:

(I1) nblock: number of grid-cells
(I2) ivar: number of variable to be read (3 for ix,iy,iz)
(I3) text: "ix"
(I4) text: "iy"
(I5) text: "iz"
for each cell:
(I6) ix,iy,iz

- NameInj = CELL_FILE_FLUX_WEIGHTED

cell file flux weighted → particles injected in given cells proportionally to their Darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	filename	*filename*: name of the ascii file containing the cells where particles are injected.

COMMENT:
- Particle location in x and y generate randomly within a given grid-cell;
- File format:

(I1) nblock: number of grid-cells
(I2) ivar: number of variable to be read (3 for ix,iy,iz)
(I3) text: "ix"
(I4) text: "iy"
(I5) text: "iz"
for each cell:
(I6) ix,iy,iz

- NameInj = HORIZONTAL_PLANE_FLUX_WEIGHTED

horizontal plane flux weighted → particles injected in a horizontal plane proportionally to the Darcy velocity.

73	Pmass, Zone, Specie	*see point injection*
74	x1, y1, x2, y2, z1	*x1*, *y1*, *z1*: x,y,z coordinates of the plane corner *x2*, *y2*: x,y,z coordinates of the second plane corner (same z)

OPTIONS FOR TypeInj:
- TypeInj = DIRAC

Dirac → Instantaneous injection.

75	TimeStartInj	Time at which particles are injected
76	Np	Number of particles to inject (delete the line if NameInj = READ_PARTICLE_FILE)

- TypeInj = GENERAL

General → Injection time and mass fluxes specified in a TimeFunction file.

75	Filename, const	*Filename*: Name of the ascii TimeFunction file; *const*: multiplier (of mass fluxes specified in TimeFunction file)

COMMENT:
- TimeFunction file format:

(I1) Header
(I2) text: TimeFunction name
(I3) Number of time steps
for each time step:
(I4) Time, Mass_flux
Time: Start time to apply given mass flux
Mass_flux: Mass flux applied until next time step

B.10 Recirculation

LINE	VARIABLE	DESCRIPTION
77	Text	
78	Text	
79	Text	
80	Logical Flag	True (T) if package is active
81	Nconnect	Number of connections between groups of wells
for each connection:		
82	[W1$_{out}$ W2$_{out}$ ... -> W1$_{in}$ W2$_{in}$ ...]	Description of the connection between groups of wells (given by their *WellID* assigned in the Control Surfaces Package)
83	Filename, const	*Filename*: Name of the ascii TimeFunction file; *const*: multiplier (of mass fluxes specified in TimeFunction file)

COMMENT:
- A connection is always described between brackets;
- Particles leaving a given well given at the left of the arrow are split and transferred into wells given at the right of the arrow;
- The name of the wells correspond to the name given in Control Surfaces Package.
- The TimeFunction is controlling when the recirculation framework is active.

B.11 Post-Processing and Output Options

LINE	VARIABLE	DESCRIPTION
84	Text	
85	Text	
86	Text	
plume snapshots parameters		
87	Text	
88	Flag	*Flag*: 1 if Print Cartesian Spatial Moments at Snapshots defined bellow (0 otherwise)
89	Flag	*Flag*: 1 if Print Particle Cloud at Snapshots defined bellow (0 otherwise)
90	Tlen, Ntstep, Tmult	*Tlen*: Total elapsed time *Ntstep*: Total number of shots *Tmult*: Multiplier $\rightarrow$ time shots are calculated as $dt(i+1) = Tmult \times dt(i)$
breakthru curve parameters		
91	Text	
92	Flag	*Flag*: 1 if Print temporal moments of BTCs (0 otherwise)

93	Flag, ngrid, method, bw, min,max	*Flag*: 1 if Print BreakThrough Curves (BTCs) (0 otherwise) *ngrid*: size of the grid used for pdf reconstruction *method*: options $\rightarrow$ BOX, TRIANGLE, GAUSS, PLUGING

NOTE: The method PLUGIN optimizes the bandwidth with an iterative algorithm that minimizes the mean integrated squared error of the density function. In this case the resulting bandwidth is the standard deviation of the Gaussian density function. For most conditions works quite well.

...		*bw*: half bandwidth support for histogram evaluation • bw $< 0 \rightarrow$ bw estimated by the program *min*: minimum value of the histogram bin • min $< 0 \rightarrow$ min estimated by the program *max*: maximum value of the histogram bin • max $< 0 \rightarrow$ max estimated by the program
92	Flag, Frequency	*Flag*: 1 if Print Cumulative BTCs (0 otherwise) *Frequency*: Frequency of printing particles = moves/prints
89	Flag, Frequency, Particle	*Flag*: 1 if Print particle path (0 otherwise) *Frequency*: Frequency of printing particles = moves/prints *Particle*: Number of particle to print. If < 0 all particles are printed