ADE 3D FEM Package, v2.0:
A Practical User's Guide

A Finite Element Method numerical solution package to solve the Advection Dispersion Equation in 3D

Rami Ben-Zvi[1], Nimrod Schwartz[1]

1. Institute of Environmental Sciences, The Hebrew University of Jerusalem, Rehovot, Israel
Email addresses: ytaironigal@gmail.com; nimrod.schwartz@mail.huji.ac.il

24/01/2020

---------- L E G A L N O T I C E ----------

The "ADE 3D FEM Package Software" is produced by Rami Ben-Zvi and Nimrod Schwartz (The Hebrew University of Jerusalem).

*Permission to use " AD**E** 3D FEM Package software" is hereby granted to non-profit educational and research institutions, for educational and research purposes only, provided and as long this Notice appears.*

Distribution of this package in whole or in part, in its current or in any modified form, is strictly forbidden.

THE ADE 3D FEM PACKAGE SOFTWARE IS BEING DEVELOPED AS A TOOL FOR SCIENTIFIC RESEARCH. HENCE IT IS NOT PRESENTED AS ERROR FREE, ACCURATE, COMPLETE, OR USEFUL FOR ANY SPECIFIC APPLICATION, AND THE HEBREW UNIVERSITY OF JERUSALEM MAKE NO WARRANTY OR REPRESENTATION THERETO.

THE HEBREW UNIVERSITY OF JERUSALEM SHALL NOT BE LIABLE FOR ANY CLAIMS, DEMANDS, LIABILITIES, COSTS, LOSSES, DAMAGES OR EXPENSE OF WHATSOEVER KIND OR NATURE CAUSED TO OR SUFFERED BY ANY PERSON OR ENTITY THAT DIRECTLY OR INDIRECTLY ARISE OUT OR RESULT FROM THE USE OF THE CTRW-FEM SOFTWARE OR IN CONNECTION THEREOF.

Trademarks appear throughout this package and documentation without any trademark symbol; they are the property of their trademark owner. There is no intention of infringement; the usage is to the benefit of the trademark owner.

For usage by and for commercial entities please contact Nimrod Schwartz.

If you publish work benefiting from this software package, please cite it as:

Rami Ben-Zvi and Nimrod Schwartz,
ADE 3D FEM Package, v2.0: A Practical User's Guide,
https://github.com/SoilHydrology/ADE3D

Contents

1. Introduction

The present package solves numerically the Advection Dispersion Equation (ADE) in a three- dimensional porous medium (either inert or reactive) by the Finite Element Method (FEM). It was developed from one- and two- dimensional Continuous Time Random Walk (CTRW) codes, described in references [1 – 3] and available in [4**Error! Reference source not found.**]. The ADE is degenerated from CTRW by using a unity memory function.

The first part of this manual is devoted to the 3D chemical transport solver, explaining its I/O file structures and, briefly, also relevant pre- and post- processing tools. After a short theoretical presentation, the I/O files, pre- and post- processing are detailed.

More specifically, Section 1 summarizes the transport equations in a concise manner, including references to more detailed papers that focus specifically on CTRW-FEM applications. Section 2 gives step-by-step instructions for installing all of the packages. Finally, Section **Error! Reference source not found.**33 includes the input and output of each package.

1.1 Software

The codes are available as FORTRAN (f77 and f90) source codes and also as executables (built with Simply Fortran version 3 on a 64-bit Windows 10 machine). Efforts were made to adhere to accepted standards and avoid using extensions, so that ***any*** FORTRAN suite should be appropriate. The source code is commented extensively and straightforward. No special libraries are needed, except the BLAS level 1 function DDOT (double-precision scalar product of two vectors). For completeness, the DDOT source is added to the package. All floating-point variables use 64 bits. Example cases and files for each of these parts are also included.

1.2 Further notes

The software has been tested on a personal computer with installation of Windows 10 – 64 bit, Simply Fortran version 3 and MATLAB R2017b.

The FORTRAN codes use both f90 and f77 and comply with general standards; it is hoped that the codes will function on any FORTRAN installation.

The Simply Fortran files (*.prj, Makefile) have been included to assist with building the executables within Simply Fortran . Other systems will have to create the relevant building tools.

In the examples presented here the geometry and grid are orthogonal and uniform, but there is no such a limitation in the code: it will accept *any* grid that is admissible (i.e., all elements have positive Jacobian determinant and the elements do not overlap each other). This was verified on a small cluster of arbitrarily-shaped hexahedral elements built manually. Any mesh generator may be used to create grids, but a translating software should be built to convert it to the required input files formats.

The codes are written in both standard FORTRAN77 (*.f files) and standard FORTRAN90 (*. f90 files). If a user wishes to use C/C++ instead, the codes may be converted. Note that some automatic conversion tools are available, e.g.

f2c - a program to convert Fortran 77 to C code (developed by Bell Laboratories): http://www.netlib.org/f2c

fable - Automatic Fortran (fixed-format) to C++ conversion: http://cci.lbl.gov/fable

Fortran 90 to C compiler: http://www.ncsa.illinois.edu/People/mdewing/f90toC

which may help. Even manual conversion should be straightforward (although laborious).

1.3 Practical advice

Having downloaded and installed the zipped files (as detailed in Section 2), the user is advised to proceed to the 3D solver, optionally with its mesh generator, run the example and modify it (and also practice using the freely available general FEM post-processor GMSH [5]).

Troubleshooting: If you get a "Runtime Error", "Missing DLL Error", or other such error notification immediately upon running any of the codes, you may be missing .dll files in your system. This is resolved easily by using this link to install Intel Fortran Redistributables for Windows on Intel 64:
https://software.intel.com/sites/default/files/managed/01/9c/w_cprof_p_11.1.072_redist_intel64.zip
For background, you can read about links to the redistributable installation packages for the Intel Compiler Professional Editions for Windows at:
https://software.intel.com/en-us/articles/redistributable-libraries-of-the-intel-c-and-fortran-compiler-for-windows

1.4 Transport Equation

The 3D solver solves the reactive advection dispersion transport equation

$$\frac{\partial C^s(\boldsymbol{x},t)}{\partial t} + \int_0^t \nabla \cdot \{\boldsymbol{v}(\boldsymbol{x})C^s(\boldsymbol{x},\tau) - [\boldsymbol{D}(\boldsymbol{x})\nabla C^s(\boldsymbol{x},\tau)]\}\, dt = S^s(\boldsymbol{x},t) \tag{1}$$

or in tensor notation

$$\dot{C}^s(x_i,t) + v_i(x_i)C^s(x_i,t) - \left[D_{ij}(x_i)C^s{}_{,j}(x_i,t)\right]_{,i} = S^s(x_i,t) \tag{2}$$

where the bimolecular reaction $A + B \rightarrow C$ is considered, C^s is the concentration of species s (in [mole/m^3]), x_i is the coordinate vector (in [m]), t is time (in [min]), , v_i is the particle (chemical species) velocity vector (in [m/min]), D_{ij} is the generalized dispersion tensor (in [m^2/min]), S^s is the species source term, including a prescribed time-independent (but spatially dependent) volumetric mass source and a reaction source of species s (in [mole/m^3-min]), the superscript s denotes the species number and tensor notation is used, in which lower case subscript indices denote components, a repeated index denotes summation, and comma denotes covariant derivative.

The reaction kinetics are assumed to follow the law of mass action: $S^A = S^B = -S^C = \Gamma C^A C^B$, for this second-order reaction where Γ is the reaction rate coefficient. This reaction is just a generic one. Extending it to more complex reactions (e.g., bidirectional) or to a set of reactions is straightforward and simple (although it may cause stability issues).

The initial condition (IC) and boundary conditions (BCs) and the numerical method used are detailed in [2], and the reactive source term is developed in [3].

The FEM solver calculates and output the nodal resident and flux-weighted concentrations at preselected times. These can be accessed and manipulated later (e.g., by the available post processing tools) to produce plots of spatial concentration profiles and temporal concentration breakthrough curves.

The solver allows for high order shape functions. The order of each element is input by the user. The local element residual (discretization error) is also calculated and reported. Details of this addition will be published in a forthcoming paper.

1.5 Versions log

02/12/20 version 1.0.

2. Installation of the packages

All of the software resides in two zip files freely accessible at:
https://github.com/SoilHydrology/ADE3D
The following zip files are available:

ADE-3D: 3D transport solver, including source files, executable, and a 3D example directory.

MeshGenRect: 3D simple orthogonal parallelepiped and uniform mesh generator, including source files, executable, and a 3D mesh generation example directory for ADE-3D. Output from this subpackage can be used as input to ADE-3D.

2.1 ADE-3D: 3D solver

Unzip all files in ADE-3D.zip into a convenient location (e.g., <top-dir\ADE-3D>). When completed, this folder includes:

1. The 3D solver executable, ADE-3D.exe.
2. An example I/O subdirectory, Warrick, (a 40×20×20×elements comparison to an analytic solution from [6] pp 318-321).
3. The FORTRAN source items, *.f90 and *.f.
4. Files used by Simply Fortran to build the executable ((*.prj, Makefile).

Note: Items 1-2 in the above list are sufficient to use the code "as is". Items 3-4 are intended for those wishing to study the code, build it on different platforms, and/or extend/modify it for their specific needs.

2.2 MeshGenRect: 3D simple rectangular and uniform mesh generator

Unzip all files in MeshGenRect.zip in a convenient location (e.g., <top-dir\MeshGenRect>). When completed, this folder includes:

1. The 3D mesh generator executable, MeshGenRect_3D.exe.
2. An example I/O subdirectory, MeshGenRect-example.
3. The FORTRAN source items, *.f90.
4. Files used by Simply Fortran to build the executable ((*.prj, Makefile).

Note: Items 1-2 in the above list are sufficient to use the code as-is. Items 3-4 are intended for those wishing to study the code, build it on different platforms, and/or extend/modify it for their specific needs.

3. Input/Output (I/O)

The codes may use any consistent units system. Usually SI is preferred here, but occasionally, for convenience, other units (e.g., minutes rather than seconds) may be used. In the latter case, ensure that all input data are consistent..

3.1 Input files (for ADE-3D package)

All I/O files are ASCII, and all input files are also free-formatted. The respective 1D and 2D input files appear in the FEM-1D and FEM-2D packages.

3.1.1 The main input file, **finp.txt**

The main input file consists of the following data.

Parameters

Ne	total number of elements
Nn	total number of nodes
Nb	total number of BC faces
Nm	total number of materials
Nd	maximum number of elements per node
Ns	total number of species
Npol	max. polynomial order for any element (must be between 1 and 7)
Nf	total number of element faces

Properties

Kappa	reaction kinetic coefficient
tc	characteristic time for the reaction
por	porosity

Time

tmax	end time
dt	time step
dto	time step for output

Temporal scheme

theta	implicitness factor (0 – explicit, 0.5 – Crank-Nicholson, 1 – fully implicit)

bCGstab input

The input for par(2:3), ipar(6), fpar(1:2), fpar(11) and the suggested values – for a description see comments below (from bCGstab.f)

ipar(2) = 0	no preconditioning
ipar(3) = 0	default stopping criteria
ipar(6) =-1	maximum $[A]\{v\}$ operations (unlimited if <0)
fpar(1)	relative tolerance
fpar(2)	absolute tolerance
fpar(11) = 0	initial number of floating-point operations from matrix-vector multiplications and preconditioners.

Implicit BC and source convergence parameters

maxit	maximum number of iterations
Ctol	allowable maximum absolute tolerance for C^s

3.1.2 The nodal data file, **nodes.txt**

The nodal data file consists of Nn nodal coordinates and initial conditions (ICs) data, one line per node.

n	node number (**NOTE** that this number is not used, and is overridden)
x, y, z	nodal x, y and z coordinates
$C_s(x,0)$	nodal IC (Ns values, one for each species, *s*)

3.1.3 The elements data file, **elements.txt**

The elements data file consists of Ne nodes and material data, one line per element.

e	element number (**NOTE** that this number is not used, and is overridden)
n1	element node number 1
n2	element node number 2
n3	element node number 3
n4	element node number 4
n5	element node number 5
n6	element node number 6
n7	element node number 7
n8	element node number 8
mat	element material number

3.1.4 The velocity data file, **v.txt**

The Darcian velocity (or the advective flux, i.e., the superficial velocity, being the physical velocity times the porosity) and dispersion of Ne elements data, one line per element.

e	the element number (**NOTE** that this number is not used, and is overridden)
Vx	Darcian velocity *x* component
Vy	Darcian velocity *y* component
Vz	Darcian velocity *z* component

NOTE that immediately after v.txt reading, Vx, Vy, Vz are converted to the physical velocity, which is also written to the results.

3.1.5 The dispersion data file, **d.txt**

The dispersion of Ne elements data, one line per element.

e	element number (**NOTE** that this number is not used, and is overridden)
Dxx	dispersion *xx* component
Dxy	dispersion *xy* component
Dxz	dispersion *xz* component
Dyx	dispersion *yx* component
Dyy	dispersion *yy* component
Dyz	dispersion *yz* component
Dzx	dispersion *zx* component
Dzy	dispersion *zy* component
Dzz	dispersion *zz* component

3.1.6 The BC data file, **bcs.txt**

The boundary conditions (BCs) data file consists of Nb BC faces data, one line per face.

f	element local boundary face number (**NOTE** that this number is not used, and is overridden)
BCe	BC element number
BCtype	'R', 'N' or 'D' for Robin, Neumann or Dirichlet, respectively
BCvalue	prescribed $\bar{j}$, $\bar{q}$ or $\overline{C}$ nodal values used for the for R, N or D BC, respectively (Ns values, one for each species, 4 sets - one for each element face node) for 'R' and 'N' these are NORMAL fluxes, positive for outflow and negative for inflow

3.1.7 The time-independent source data file, **source.txt**

The souce data file consists of Ne elements data, one line per element.

e	element number (**NOTE** that this number is not used, and is overridden)
s	species number (**NOTE** that this number is not used, and is overridden)
Sxy	eight prescribed time-independent source nodal values, 1 line for each species. (**NOTE** that for cases that do not include such time-independent source, zeros mush be given)

3.1.8 The elements faces data file, **faces.txt**

The elements data file consists of Nf faces data, one line per element.

global_face	element global face number (**NOTE** that this number is not used, and is overridden)
element	element number
local_face	element local face number (1 for west, 2 for south, 3 for bottom, 4 for east, 5 for north, 6 for top)
node1	face global node number 1
node2	face global node number 2

node3	face global node number 3
node4	face global node number 4
neighbor_element	neighbor element number
neighbor_element	neighbor element global face number

3.1.9 Legendre polynomial data file, **legendre.txt**

The Legendre polynomial coefficients for orders 0 – 7 are given, one line for each polynomial

NOTE this file should be modified ***only*** if orders higher than 7 are desired. In this case, the code itself should also be modified in few parts by modifying 7 with the new value. The file is read in main.f90, and the coefficients are defined there immediately after reading.

Dn(i)	denominator for order i
Ain(i,0:7)	8 values for the coefficients

3.1.10 The elements order data file, **porder.txt**

The elements order data file consists of Ne lines, one line per element.

element	element number
order	polynomial order for the element

3.2 Output files (for ADE-3D package)

All output files are ASCII files.

3.2.1 3D solver output

The 3D solver produces the following output files that may be read by the 3D post-processor (see Sec. 3.4):

fout.txt is the general output, including the version identification, a partial input echo, the calculated CFL and Peclet numbers, the calculated SUPG parameter, the initialization and the total CPU run time. In case the run aborts prematurely, the error(s) (if identified) are also reported.

fdbg.txt is the debugging output. As-is, it just prints one line for each entry to the main program, MAIN, and to the output routine, OUT. If there are input errors or if the linear equations solver, BCGSTAB, fails to converge it contains useful data that may help to identify the problem.

resi.txt is the element residual file. It consists of a value (one line per element) for each element and each time step.

jump.txt is the nodal jumps file. It consists of a value (one line per node) for each node and each time step.

post.msh is the output file that may be read by the general purpose GMSH post-processor (see Sec. 3.4):

3.3 3D pre-processor for ADE-3D

While it is possible to enter all of the input data manually, it may be easier to use a pre-processor. For 1D, the larger data (the nodes and elements) may be generated by simple tools, i.e., a simple ad hoc EXCEL spreadsheet or MATLAB scripts. For 3D, a simple FORTRAN pre-processor was implemented for an (i) orthogonal parallelepiped domain, (ii) uniform velocity and dispersion coefficient, (iii) constant, uniform or $\delta(x,y,z)$ ICs, (iv) time-independent BCs (assumed to be piecewise-continuous *x*-, *y*- or *z*-dependent functions) for each boundary face for each chemical species, and (v) time-independent volumetric mass source term (assumed to be the product of piecewise-continuous *x*-, *y*- or *z*-*y*-dependent functions).

The pre-processor reads an input file, **meshinp.txt**, and produces the following files: **nodes.txt**, **elements.txt**, **v.txt, d.txt source.txt,** and **bcs.txt**. In addition, some of the parameters of **finp.tx**t are written to **param.txt** (the rest of **finp.inp** should be entered manually). **Legendre.txt** file need not be modified (see comments above). The **porder.txt** file may be created by a spreadsheet, MATLAB etc.

3.3.1 The 3D pre-processor input file, **meshinp.txt**

The 3D pre-processor input refers to files in the MeshGenRect package.

Parameters

Darcy	F for ADE-3D
Xmin, Xmax, Ymin, Ymax, Zmin, Zmax	domain geometric range
Nx, Ny, Nz, Ns	number of *x* and *y* intervals and number of species
Vx, Vy, Vz	velocity vector components

Properties

Dxx, Dxy, Dxz, Dyx, Dyy, Dyz, Dzx, Dzy, Dzz	dispersion coefficient tensor components

Time-independent source function [*S(x,y) = f(x) g(y) h(z)*]

Nxx, Nyy, Nzz	number of points defining *f(x), g(y)* and *h(z)*
(xx(k), k=1,Nxx)	coordinate *x* for *f(x)*
(fx(s,k), k=1,Nxx)	*f(x)* values (one line for each s=1, ... , Ns)
(yy(k), k=1,Nyy)	coordinate *y* for *g(y)*
(gy(s,k), k=1,Nxx)	*g(y)* values (one line for each s=1, ... , Ns)
(zz(k), k=1,Nzz)	coordinate *z* for *h(z)*
(hz(s,k), k=1,Nzz)	*h(z)* values (one line for each s=1, ... , Ns)

IC

delta (TRUE or FALSE) and its coordinates Xdelta, Ydelta, Zdelta	
(C(s), s=1,Ns)	IC (constant for all nodes)

BCs function [*f(x), g(y)* and *h(z)*]

(BCt(f), f=1,6)	BC type for (west, south, bottom, east, north, top) faces. One of: 'D' for Dirichlet BC 'N' for Neumann BC 'R' for Robin BC
(Nxy1(f), f=1,6)	number of points defining *f(x), g(y)* or *h(z)* for (west, south, bottom, east, north, top) faces, 1st coordinate
(Nxy2(f), f=1,6)	number of points defining *f(x), g(y)* or *h(z)* for (west, south, bottom, east, north, top) faces, 2nd coordinate
(xy1(1.i), i=1, Nxy(1))	*y* (1st) coordinates for the west face (f=1)
(vl1(1,s,i), i=1,Nxy(1))	*g(y)* values for f=1 (one line for each s=1, ... , Ns)
(xy2(1.i), i=1, Nxy(1))	*z* (2nd) coordinates for the west face (f=1)
(vl2(1,s,i), i=1,Nxy(1))	*h(z)* values for f=1 (one line for each s=1, ... , Ns)
(xy1(1.i), i=1, Nxy(2))	*z* (1st) coordinates for the south face (f=2)
(vl1(1,s,i), i=1,Nxy(2))	*h(z)* values for f=2 (one line for each s=1, ... , Ns)

(xy2(1.i), i=1, Nxy(2))	*x* (2nd) coordinates for the west face (f=2)
(vl2(1,s,i), i=1,Nxy(2))	*f(x)* values for f=2 (one line for each s=1, ... , Ns)
(xy1(1.i), i=1, Nxy(3))	*x* (1st) coordinates for the bottom face (f=3)
(vl1(1,s,i), i=1,Nxy(3))	*f(x)* values for f=3 (one line for each s=1, ... , Ns)
(xy2(1.i), i=1, Nxy(3))	*y* (2nd) coordinates for the bottom face (f=3)
(vl2(1,s,i), i=1,Nxy(3))	*g(y)* values for f=3 (one line for each s=1, ... , Ns)
(xy1(1.i), i=1, Nxy(4))	*y* (1st) coordinates for the east face (f=4)
(vl1(1,s,i), i=1,Nxy(4))	*g(y)* values for f=4 (one line for each s=1, ... , Ns)
(xy2(1.i), i=1, Nxy(4))	*z* (2nd) coordinates for the east face (f=4)
(vl2(1,s,i), i=1,Nxy(4))	*h(z)* values for f=4 (one line for each s=1, ... , Ns)
(xy1(1.i), i=1, Nxy(5))	*z* (1st) coordinates for the north face (f=5)
(vl1(1,s,i), i=1,Nxy(5))	*h(z)* values for f=5 (one line for each s=1, ... , Ns)
(xy2(1.i), i=1, Nxy(5))	*x* (2nd) coordinates for the north face (f=5)
(vl2(1,s,i), i=1,Nxy(5))	*f(x)* values for f=5 (one line for each s=1, ... , Ns)
(xy1(1.i), i=1, Nxy(6))	*x* (1st) coordinates for the top face (f=6)
(vl1(1,s,i), i=1,Nxy(6))	*f(x)* values for f=6 (one line for each s=1, ... , Ns)
(xy2(1.i), i=1, Nxy(6))	*y* (2nd) coordinates for the top face (f=6)
(vl2(1,s,i), i=1,Nxy(6))	*g(y)* values for f=6 (one line for each s=1, ... , Ns)

NOTES:

- At least 2 points should be used to define any of the functions *f*, *g* and *h*.
- The endpoints of the functions *f*, *g* and *h* must coincide with the domain ends.
- For any pair of Dirichlet BCs meeting at a corner, the values at that corner should be identical (but for other types of BCs pair meeting at a corner, the values may be different, because they are prescribed ***normal*** values, and normals at a corner have two distinct directions).
- If input errors are found in **meshinp.txt**, error messages will appear at the end of **param.txt** and the run will abort.

3.4 Post-processors for ADE-3D

3.4.1 The 3D post-processor

The general FEM post-processor GMSH [45] (freely available) is used for graphical display of the 3D results for each species, *s*: the nodal resident concentration, C_s, the nodal RHS, T_s, and the nodal advection, dispersion and total flux vectors, q_{is}^c, q_{is}^d, and j_{is}, respectively. The input (at elements centers) velocity vector, v_i, and dispersion tensor, D_{ij}, are also available.

3.4.2 Other tools

FEM_to_Matrices.m is a MATLAB script that reads the I/O files and rearrange the results for an orthogonal parallelepiped domain and mesh. **post_res.m** and **post_RMS.m** are MATLAB scripts to analyze and display the element residuals and nodal jumps.

3.5 Examples

3.5.1 ADE-3D example

The example of the 3D solver is a comparison to an analytic solution from [6] pp 318-321). All inputs are non-dimensional. Fluid at a constant velocity of 2 in the x direction flows through an inert purely fluid domain $-1 \le x \le 3$, $0 \le y,z \le 2$ with dispersion coefficients of 0.4 and 0.2 in the longitudinal and transverse directions, respectively. The concentration is initially 0, and at $t=0$ a delta-function of strength 1 is imposed at the origin. Zero flux is prescribed at both x end faces and symmetry – at all other faces. The solution up to t=0.5 is obtained. The comparison to the reference solution is depicted in **compare.xlsx**, sheet FEM-3D.

To replicate these results:

- Create a new folder, and copy the input files meshinp.txt and finp.txt from the example folder.
- Create shortcuts to the MeshGenRect pre-processor and to ADE-3D executables in the new folder. On Windows OS it is important to set the shortcut properties (right-click on the shortcut, choose "Properties") such that it starts in the current folder (i.e., clear the "Start in" box in the Shortcut tab).
- Run the MeshGenRect executable by double-clicking on the shortcut. **NOTE**: An empty DOS window opens and indicates that calculations are in progress. This window is closed automatically when the run ends.
- Edit the finp.txt file, and change the values of Ne, Nn, Nb, Nm and Ns according to those in param.txt.
- Run the ADE-3D executable by double-clicking on the shortcut. **NOTE**: An empty DOS window opens and indicates that calculations are in progress. This window is closed automatically when the run ends.
- The output files produced should be identical (up to machine precision) to those in the example folder.
- You may also want to run the GMSH post-processor (should be downloaded and installed first) by double-clicking the output file post.msh to display the results graphically.

References

1. Ben-Zvi, R., Scher, H., Jiang S., Berkowitz, B. (2016) One-dimensional finite element method solution of a class of integro-differential equations: Application to non-Fickian transport in disordered media, *Transport in Porous Media*, 115(2), 239-263.
2. Ben-Zvi, R., Scher, H., Berkowitz, B. (2017) Two-dimensional Finite Element Method solution of a class of integro-differential equations: Application to non-Fickian transport in disordered media, *International Journal for Numerical Methods in Engineering*, 112(5), 459-478, doi:10.1002/nme.5524.
3. Ben-Zvi, R., Nissan, A., Scher, H., Berkowitz, B. (2018) A continuous time random walk (CTRW) integro-differential equation with chemical interaction, *European Journal of Physics B*, 91, 15, doi.org/10.1140/epjb/e2017-80417-8.
4. Ben-Zvi, R., Jiang, S., Scher, H., Berkowitz, B., The CTRW-FEM Package v1.0: A practical user's guide, http://www.weizmann.ac.il/EPS/People/Brian/CTRW/software
5. Geuzaine, C., Remacle, J.-F. (2009) GMSH: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *Int. J. Numerical Methods in Engineering*, 79(11), 1309-1331. http://geuz.org/gmsh).
6. Warrick, A. W. (2003). *Soil water dynamics*. Oxford University Press.