MMESD B.13-005
Rev. 0

DESCRIPTION AND VERIFICATION OF PTRAX: A RANDOM WALK MODEL FOR PREDICTING GROUNDWATER SOLUTE TRANSPORT

by

Dudley J. Benton
Steven C. Young
and
Nicolas J. Williams
Environmental Consulting Engineers, Inc.
P.O. Box 22668
Knoxville, TN 37933

for

Martin Marietta Energy Systems, Inc.
Oak Ridge, Tennessee 37831

for the
U.S. Department of Energy
under Contract No. DE-AC05-840R21400

August 31, 1995

File PTRAX1.WPD

EXECUTIVE SUMMARY

In order to efficiently simulate solute transport and particle tracking in complex stratigraphy, Environmental Consulting Engineers, Inc. (ECE) developed the particle-tracking code PTRAX. Particle tracking is a powerful and versatile tool that offers several advantages over finite-difference and finite-element modeling for solute transport. These advantages include the mapping of flow paths, reverse tracking to delineate capture zones, and less grid refinement in regions where dispersion length values are small. PTRAX is a fast, three-dimensional particle tracking code capable of two- and three-dimensional simulations for grids based on an assortment of element types After reading in a velocity file, PTRAX can delineate flow paths and/or simulate groundwater transport. Because its mathematics are based on triangles and tetrahedra, PTRAX can couple with virtually any velocity field. This capability is particularly valuable where a non-layered finite-element grid has been used to simulate flow through complex stratigraphy. In addition, because it uses distance as the variable of integration, PTRAX can simulate particle movements significantly faster than conventual particle tracking codes such as MOC, RANDOM WALK, and MT3D, which all use time as the variable of integration. Three test problems are presented to demonstrate that the algorithms in PTRAX have been properly implemented.

CONTENTS

FIGURES

TABLES

NOMENCLATURE

α . . . . dispersion length
n . . . . number of steps and/or particles
R . . . . a normalized random number
ΔS . . . random step length
ΔT . . time step
U . . . . velocity component in the X direction
V . . . . vector velocity
V . . . . velocity component in the Y direction
W . . . velocity component in the Z direction

Subscripts
I . . . . particle step
L . . . . in the longitudinal direction
M . . . mean (i.e., non-random)
R . . . . random
T . . . . in the horizontal-transverse direction
V . . . . in the vertical-transverse direction
X . . . . in the X direction
Y . . . . in the Y direction
Z . . . . in the Z direction

1. INTRODUCTION

1.1 OVERVIEW OF PARTICLE-TRACKING CODES

Particle tracking involves determining flow paths by tracking particles through a velocity field. The simulation of particle tracks can be used to delineate capture zones and to predict solute migration. Although finite-difference and finite-element codes are primarily used to simulate groundwater flow systems, the application of these codes to predict groundwater solute transport is not as popular as particle tracking codes.

Widely used particle-tracking codes include MOC (Konikow and Bredeheoft, 1978), RANDOM WALK (Prickett et al., 1981), PATH3D (Zheng, 1992), MODPATH (Pollock, 1989), and MT3D (Zheng, 1994). Factors that affect the application of these models include: if, and how, hydrodynamic dispersion is included and the method used to track the particles. The codes MODPATH and PATH3D cannot simulate hydrodynamic dispersion. These codes only calculate flow paths and are primarily useful for delineating well capture zones. These particle tracking codes have the advantage over finite-difference and finite-element codes of avoiding numerical error when the grid spacings are much larger than the characteristic dispersion length. Particle-tracking codes such as MOC and RANDOM WALK have solution techniques that prevent numerical error from occurring regardless of the ratio of grid spacing to dispersion length. This capability is a significant benefit because dispersion length in the vertical transverse direction typically rangesbetween 0.1 and 0.001 m. Such element lengths are too small for practical numerical model applications.

A limitation of MOC and RANDOM WALK is that they are two-dimensional. MT3D is a three-dimensional particle-tracking code, but it also has limitations. Because of its mathematical coding, MT3D is not designed for applications with grid meshes other than layers of rectangular blocks. MT3D was specifically designed to couple with the groundwater flow model MODFLOW (McDonald and Harbaugh, 1988). As a result, MT3D is only compatible with block-centered finite difference codes. For many aquifers, this type of grid mesh is sufficient to represent the aquifer properties and boundary conditions; however, for very heterogeneous multilayered aquifers that have complex stratigraphy, grid elements of various thicknesses and shapes are required. As a result, MT3D is limited to situations where the geology can be simplified to a layer-cake model.

A second limitation of MT3D is that the code is not optimized to efficiently process the tracks of a large number of particles. MT3D, like RANDOM WALK and MOC, does not use an efficient algorithm for particle tracking. MT3D operates by tracking all the particles simultaneously according to the same time-steps. In this scheme, the time-step must be selected such that no particle will move through more than one element. In some model simulations the resident time associated with the particles in various elements differ by orders of magnitude. In this case, the smallest resident time is selected as the uniform time-step increment for all particle movement. This small time-step results in greatly increased computational run-time. This occurs with elements having large differences in their dimensions and/or velocities.

An efficient method for tracking particles through elements with variable dimensions and velocities involves a piece-wise integration over the spatial domain of each element as described by

Kinzelbach et al. (1991), Pollock (1989), and Scheibe (1993). The method involves projecting a particle by nonuniform time-steps that are selected so that the particle transverses an element in a single time-step. In this method, the amount of computation required to move a particle through a velocity field depends only on the number of elements that a particle intersects along a flow path. Implicit in the application of this spatial integration, or Hamilton framework, is the asynchronous movement of the particles. Asynchronous movement in the particles makes this an efficient particle tracking method that is incompatible with the algorithms used by method-of-characteristic codes[1] and is difficult to implement with algorithms used by random walk codes. Currently, no commercially available solute transport code has successfully integrated an efficient particle tracking algorithm with fully three-dimensional random walk algorithms.

In order to efficiently simulate solute transport and particle tracking in complex stratigraphy, Environmental Consulting Engineers, Inc. (ECE) developed the particle-tracking code PTRAX. PTRAX is designed to be user friendly, modular, and computationally efficient. PTRAX has built-in documentation, is compatible with any type of conventional numerical meshes, and uses random-walk algorithms within a Hamilton framework for particle integration to provide exceptionally fast solute transport simulations.

1.2 PURPOSE

The purpose of this report is to provide a short description of PTRAX for potential users and to document some of the PTRAX verifications that ECE has performed. Several applications of PTRAX are presented to illustrate that the basic components of PTRAX have been properly implemented. The report does not provide a description of the code algorithms nor does it document the advantages of using PTRAX versus other particle tracking codes.

---

[1] The method-of-characteristics treats an ensemble of particles like an advancing wave-front.

2. PTRAX DESCRIPTION

The following is a brief description of the process by which PTRAX tracks particles within a two- or three-dimensional domain.

2.1 GRID TRANSFORMATION

PTRAX is a code that uses a velocity field generated from a finite-difference or finite element groundwater flow code. Current applications of PTRAX involve the groundwater flow code FRAC3DVS (Therrien et al., 1995), which supports finite difference and finite element solution techniques along with a variety of elemental shapes. The first series of tasks performed during a PTRAX 3D simulation is the decomposition of each grid element into a group of tetrahedra. All of PTRAX's computations are based on triangular or tetrahedral elements. This transformation enables PTRAX to support different types of elemental shapes without changes in boundaries or distribution of elemental properties. All original elemental boundaries are maintained. The process of transforming a grid into a cell-linked space within which particles can be tracked constitutes the first section of PTRAX. The following steps accomplish this transformation.

2.1.1 Defining Nodes

Nodes are defined by unique locations in space that are read from a data file. The number of dimensions is inferred from the number of coordinates. Two coordinates are interpreted as X,Y and imply a 2D space. Three coordinates are interpreted as X,Y,Z and imply a 3D space. Non-unique or coincident nodes lead to singular basis equations and must be rejected. A check is performed as the nodes are read.

2.1.2 Defining Elements

Elements are defined by groups of nodes and are read from a data file. In a 2D space, three nodes imply triangular elements and four nodes imply quadrangular elements. In a 3D space, four nodes imply tetrahedral elements, six nodes imply prismatic elements, and eight nodes imply hexahedral brick elements. The nodes composing an element must be unique. A check is performed for this as the elements are read. Each node must appear in at least one element. Every node must be connected to every other node through the elements in order to have a unified (as opposed to a disjointed) domain. A check is performed for each of these after all of the elements have been read. No assumptions are made as to the angles formed by the sides of the elements (e.g., sides of 3D bricks are not assumed to form right angles).

2.1.3 Element Orientation

Elements must be numbered according to some convention in order to determine their connectivity. The convention adopted by PTRAX is counter-clockwise orientation. In 2D, this means that the area of each element is computed as a positive number. Any element having an area less than some small value (e.g., 10^{-9} times the total area) is rejected as degenerate. In 3D, this means

that the volume of each element is computed as a positive number. Any element having a volume less than some small value (e.g., 10^{-9} times the total volume) is rejected as degenerate.

Three-D elements have the added complexity of orientation of the faces. These too must be positive in the vector sense[2]. As numbering schemes differ, and so as to provide the greatest convenience, PTRAX analyzes each element and sets bit flags to indicate the orientation of each element and face.

In the case of 2D elements, it is always possible for PTRAX to automatically renumber the elements so as to conform to their orientation. In the case of 3D elements, however, ambiguities may arise that force the element to be rejected. If the elements can be renumbered without ambiguity, the process continues and the total number of miss-oriented elements is listed. If a minor ambiguity occurs (i.e., involving symmetrically opposed faces), a warning is issued and the process continues. If a major ambiguity occurs (i.e., involving adjacent faces), the element is rejected. The necessity of this distinction between fatal and non-fatal ambiguities will become apparent as the element splitting is described. As the orientation of each element is analyzed, various numbering conventions can be mixed within a single grid, and the end result will be the same.

2.1.4 Node:Element Links

The first step in connecting the elements involves building a list of Node:Element links. When complete, this list contains each element in which each node appears. As PTRAX uses dynamic memory allocation and pointers, this list resembles a simple data base structure and requires a minimum amount of storage. Each node is assigned an index where its list of elements begins as well as a count. The maximum count for all nodes is a measure of the bandwidth.

2.1.5 Element:Element Links

The second step in connecting the elements is to build the list of Element:Element links. This is structured like the Node:Element list and represents a recursional search of the Node:Element list for nodes common to each face of each element. Any element face that is not connected is external (i.e., represents a boundary), while any connected face is internal.

A recursional search of the Node:Element list for nodes common to each face represents a potentially immense calculation. If an exhaustive search were performed using nested loops, the number of comparisons would be proportional to the number of elements cubed. Clearly, this would be impractical for any sizable grid. PTRAX uses a *hashing*, followed by a *Q-sort* on the element faces, followed by a *bubble-up* on the facial nodes. This combined algorithm selects, in order, the members common to a variable number of lists, each of variable length. The time required for this procedure is roughly twice that required to determine the connections to a single face using nested loops. For a grid containing thousands of elements, this procedure reduces the computational time by many orders of magnitude. The time required for this procedure is of the same order of

---

[2] The faces of 3D polyhedra have a positive orientation if the dot product of the outward normal area vector with the vector beginning at the volume centroid and passing through the area centroid of each face is positive.

magnitude as reading the node and element files. The maximum number of connections between elements is a measure of the bandwidth and is listed for information.

2.1.6 Element Face Orientation

As the elements are analyzed, bit flags are stored indicating the element orientation as w ell as the orientation of each face of 3D elements. The elements are not actually renumbered; only the bit flags are set to indicate their orientation. When the elements are connected at the faces, it is necessary to mate their orientation. The orientation flags must be adjusted so that the faces where two elements are connected satisfy orientational reciprocity[3].

2.1.7 Element Splitting

In numerical analysis, the variation of parameters within an element is approximated by basis functions[4]. These basis functions vary over the element and are typically assumed to combine linearly (i.e., superimpose). The only basis functions that assure continuity of a varying parameter at every point on a face between two elements are the linear combinations of the spatial directions (i.e., $C_1+C_2X+C_3Y+C_4Z$). An added benefit of this selection of basis functions is that the area of triangular elements and the volume of tetrahedral elements are equal to the determinant of the basis matrix and must be computed anyway.

Selection of these basis functions fixes the element type in 2D as triangular and in 3D as tetrahedral. In order to analyze the grid, PTRAX splits quadrilaterals into two triangles, prisms into three tetrahedra, and bricks into five tetrahedra. Splitting quadrilaterals into triangles is a simple matter; however, splitting 3D elements requires that adjacent sides have a certain orientation.

When split into three tetrahedra, uniformly oriented prisms do not have a line of symmetry, and thus do not match-up. Proper splitting of prisms requires that every other prism be split as a mirror image. As there is an odd number of sides to the triangles forming the ends of the prisms, this does not inherently lead to orientational conflict and is basically a matter of bookkeeping, which is handled by the element orientation flags.

Although 3D bricks do have a line symmetry, they must be alternately rotated 90°. Grids where 3D brick elements are inserted in odd numbers around an internal boundary cannot be properly split into tetrahedra, as there is no combination of rotations that will result in the tetrahedra properly connecting. A check is performed for this as the elements are split, and any such conflicts result in rejection of the grid.

---

[3]Connective reciprocity implies that if **A** is connected to **B**, then **B** is connected to **A**, but in the opposite direction. Thus, a minimum requirement for connective reciprocity is that the dot product of the outward normal area vector for each pair of element faces must be negative (i.e., opposite in direction).

[4]Basis functions are most often associated with the Finite Element Method (FEM), but are inherent to all numerical formulation of spatial domains. While basis functions are explicit in FEMs, they are implicit in Finite Difference Methods.

The basic 2D building block is the triangle. The basic 3D building block is the tetrahedra. As more complex elements are split into these basic building blocks, the term *cell* is used.

2.1.8 Node:Cell Links

Because PTRAX uses dynamic memory allocation and pointers, grids whose elements do not require splitting into cells only require that the pointers be equivalenced. Grids whose elements are split into cells require additional storage. Grids whose elements require splitting must be reanalyzed for Node:Cell links. During this process, several auxiliary parameters are calculated, such as the cell centroids. Grids whose elements are not split, already have the Node:Cell links established, as Node Element links. Some runtime is reported even in these cases, because of the auxiliary parameter calculations. Once again, each node must appear in at least one cell; and every node must be connected through the cells to every other node. Grids failing any one of these tests are rejected.

2.1.9 Cell:Cell Links

Grids whose elements do not require splitting into cells only require that the pointers to the list of Cell:Cell links be equivalenced to the list of Element:Element links. Grids whoseelements are split into cells require additional storage and must be reanalyzed. Cell:Cel connectivity is built into a list. This list contains an index for each face of each cell. If the index is greater than or equal to zero, then it is the number of the adjacent cell at that face. If the index is -1, there is no cell adjacent at that face. If there is no cell adjacent at a face, then that face is an external boundary. The same fast algorithm is used to establish the Cell:Cell links as was used for the Element:Element links.

Once the list of Cell:Cell links has been established, given a starting point within any cell, all of the paths leading from that cell are given in the list. A path leaving a cell may end at a boundary or enter an adjacent cell. This Cell:Cell link list is the *road map* for particle tracking.

2.1.10 Additional Checks and Information

While this Cell:Cell list is created, a number of checks and non-essential calculations are performed. These checks include coincident nodes, overlapping cells, degenerate cells, degenerate cell faces, and cell connection reciprocity. As these additional checks have been carefully optimized, the cost in runtime is minimal while valuable information is gained about the grid structure and important checks, not usually done in many codes, are performed. Varying amounts of this information can be listed through command options. In addition to these, an entire nearest neighbor node list, associated bandwidth, and pivot matrix, which are the core of finite element modeling, can be optionally generated and listed through command options. While this information is not used by PTRAX, it can be useful in grid analysis and can serve as a further check. These steps are included because PTRAX had its origin in a FEM code; eventually this information may be used for enhanced modeling.

2.2 PARTICLE TRACKING

Particles can be seeded automatically, scattered throughout the grid, or specifically placed in a data file. The seed locations can be specified by element or by X,Y,Z location and initial mass. Because finding the cell containing the seed is a time-consuming process, the element in which the particle is seeded can be optionally specified along with the location. Thisdirectly specifies the cell if the elements are not split, or confines the range of cells to be searched for the nearest centroid if the elements are split. This can save considerable runtime.

A particle must lie unambiguously within a cell on the first step. Incorrectly specifying the starting cell will result in the particle being artificially trapped. As there is no prior step or history at the start, a particle must not be seeded on a boundary between two cells. (On subsequent steps it may frequently lie on cell boundaries.) As indicated previously, PTRAX can handle some ambiguously numbered 3D elements. Ambiguously numbered elements can result in particle reflection. Reflection will trap a particle if it isnot seeded unambiguously within the interior of the cell. Whether or not a particle is unambiguously within the interior of a cell depends on factors such as cell aspect and round-off and is not easily quantified. In order to avoid these problems, PTRAX by default repositions all seeds at their start to the centroid of the nearest cell. This feature can be disabled by a command parameter.

2.2.1 Velocity Field and Properties

The velocity field can be specified as a default for all elements or separately for each element in a data file. The velocity is initially divided by theporosity and retardation factor to convert from Darcy to true velocity. Porosity and retardation factor can be specified as a default for all elements or separately for each element in a data file. The default values are set in the code or are read from the optional configuration file, which can be modified as needed. Theconcentration (mass/volume) is multiplied by the porosity upon completion of the simulation to convert total to liquid concentration. Cells that are split inherit the properties of the parent element.

2.2.2 Solving for Particle Direction

Within a cell, a particle moves from its current position in the direction of the local velocity vector until it intersects a face[5]. The intersected face is determined by the following procedure: the equation for the line in 2D or plane in 3D defined by each face of the current cell is determined. This arises directly from the assumed linear basis functions and results in an unambiguous calculation. The necessity of splitting prisms and bricks into tetrahedra can be seen from this: four unique points do not necessarily lie in the same plane in 3D. Four points may form a saddle. There is no ambiguity in the lines and planes defined by the faces oftriangles and tetrahedra. The length of the side in 2D or area of the face in 3D is the determinant of the basis matrix. As the cells have already been screened and the lengths and areas computed, this assures non-singular results and reduces the number of calculations within a deeply-nested loop.

---

[5] The random walk is a modification to this procedure and will be detailed subsequently.

The distance along the local velocity vector, from the current particle location to the intersection with a face, divided by the velocity magnitude is the cell traverse time. If the computed time to intersect a given face is zero, the particle does not move[6]. If the time is negative, this represents a backward step and is rejected. The face that corresponds to the minimum time, greater than zero, is the first intersected, and thus, the point of exit from the cell. If the cell:cell link corresponding to the exiting face is greater than or equal to zero, then the particle continues on into that cell. If the link is equal to -1, the particle exits at the boundary.

2.2.3 Particle Track Termination

Several causes may result in the termination of a particle track. These include: capture by a well, complete decay of mass, escape through a boundary, stagnation, and the end of the tracking period.

Wells are defined by capture zones. Wells are specified in a data file by nodes, elements, or location, screen opening, and capture radius. If a particle enters the capture zone of a well, its track ends and its mass and time of capture are transferred to the corresponding well.

The decay of particle mass may be specified as a half-life for each element or a constant value for all elements. If the half-life for a cell is greater than zero, then the particlemass decays based on how long the particle stays in the cell. The decayed mass is transferred permanently to the cell in which the decay occurred. If the half-life within a cell is zero, then all of the particle mass is transferred to the current cell at the time it enters the cell, and the track is terminated.

If a particle escapes at a boundary, its track is terminated and its mass and time of escape is transferred to the global counters for escaped particles.

If a particle enters a cell having zero velocity, thetime to reach any face would be infinite, so its track is terminated, and its mass is transferred to the current cell along with its time of entry.

If a particle enters a cell where the velocity is not zero, butno exit times are computed for the faces that are greater than some small value (e.g., 10^{-9} times the previously described small length divided by the r.m.s. average field velocity), then it is considered to be *trapped.* Any occurrence of this trapping is considered anomalous (i.e., should not occur under normal circumstances). A count of such *trapped particles* is listed as an additional check.

A final cause for particle track termination is the maximum steps along a track. Before tracking a particle, it is necessary to allocate storage for its history. This is used for bookkeeping and calculation of *snapshots* or field samples at specific times. The maximum steps along a track is defined in the code or specified in the optional configuration file, which can be modified as needed.

The number of particle tracks terminating for each of these causes is listed after all of the particle tracks are computed. In addition, if particle tracks are to be savedfor plotting, the cause for

---

[6] This may seem to be a trivial case, but will have applicability in the random walk as detailed subsequently.

termination of each particle track is filed in both numerical form (i.e., an index) and text form (i.e., a string such as *boundary* or *decay*).

Two separate lists are kept for particle track termination: one associated with the particle and one associated with the receptor of the particle (e.g., a well or boundary). Any disagreement between these two lists is considered anomalous (i.e., should not occur under normal circumstances). A count of the difference between these lists, or the *missing particles*, is listed as an additional check.

2.2.4 Snapshots and Well Logs

The ensemble of particles is sampled at specific times as defined in the code or specified in the optional configuration file, which can be modified as needed. The information associated with these specific times, or *snapshots*, is saved in sequentially-named files after all of the particles have been tracked. The contribution of each particle is added to each snapshot at the end of its track. Although the particles are tracked on a cell basis, the snapshots are accumulated on an element basis. The storage for these snapshots must be allocated before any particles are tracked. If the particle tracks are to be saved for plotting, each track is filed after the snapshots are updated. The storage for a particle track is used over again so that the requirement does not grow with the number of particles.

Every time a particle is captured by a well, the mass and time of capture is recorded. Two separate lists are maintained for this capture. One is based on the snapshot interval. This requires minimal storage, which is allocated before any of the particles are tracked. A second optional list is kept that contains every particle captured by every well, its mass when captured, and when it was captured. This list grows with the number of particles and may become very large. After all of the particles are tracked, this optional list is sorted and filed by well.

Each snapshot file contains a summary by particle and mass. This summary includes the particle count and mass for each track termination cause as well as the double-checking for *missing* and *trapped* particles[7]. The centroid, concentration (mass/volume), mass, accumulation[8], and element number are filed for each element. By default, only those elements containing some mass are filed. Optionally, a command parameter can be used to force all elements to be filed.

If there are any wells, the total mass captured by each well is listed at the bottom of each snapshot file. As these entries contain fewer numbers than the element-by-element concentrations, data analysis and presentation programs should be able to directly distinguish these results. Alternately, these results might be stripped off and filed separately.

2.2.5 Random Walk/Dispersion

---

[7]The number of trapped particles should always be zero if the grid, velocity and property fields, seeds, and wells are properly defined.

[8]Cells accumulate mass as particles decay or stagnate within them.

The random walk is a means of quantifying the results of hydrodynamic dispersion, which is classically considered analogous to brownian motion in groundwater transport (Bear, 1979). Hydrodynamic dispersion includes the effects of diffusion and mechanical dispersion. The coefficient for mechanical dispersion equals the product of a dispersion length and the mean velocity.

Default dispersivities can be assigned to all elements, or separate values can be specified for each element in a data file. The default dispersion length values are defined in the code or specified in the optional configuration file, which can be modified as needed. Cells that are split inherit the properties of the parent element, including the dispersion lengths. The dispersion lengths can be applied along the grid axes (i.e., X,Y,Z) or along the local axes (i.e., longitudinal, horizontal-transverse, and vertical-transverse).

The random step length associated with a dispersion length is defined by the following equation:

$$\Delta S = R\sqrt{2\alpha |V_M| \Delta T} \qquad (1)$$

where ΔS is the random step length, R is a normalized random number[9], α is the dispersion length, $|\mathbf{V}_M|$ is the magnitude of the mean (i.e., non-random) velocity, and ΔT is the time-step.

For dispersion in several directions, multiple random numbers (i.e., Rs) and directionally associated dispersivities (i.e., α_X, α_Y, α_Z or α_L, α_T, α_V) are combined to form the random steps (i.e., ΔS_X, ΔS_Y, ΔS_Z or ΔS_L, ΔS_T, ΔS_V). For a particle traversing a cell, there is an effective random velocity associated with the random step length and implied time-step.

$$|V_R| = \frac{\Delta S}{\Delta T} \qquad (2)$$

For dispersion in several directions, the effective velocity components can be represented by a mean and random part:

$$U_T = U_M + U_R = U_M + \frac{\Delta S_X}{\Delta T} \qquad (3)$$

$$V_T = V_M + V_R = V_M + \frac{\Delta S_Y}{\Delta T} \qquad (4)$$

---

[9] A normalized random number has a mean of 0 and a standard deviation of 1.

$$W_T = W_M + W_R = W_M + \frac{\Delta S_Z}{\Delta T} \tag{5}$$

where U, V, and W are the velocity components in the X, Y, and Z directions, respectively. If dispersion lengths are specified along the longitudinal, horizontal-transverse, and vertical-transverse directions, the corresponding steps along the principle axes are computed using standard trigonometric relationships.

2.2.6 Statistical Requirements

For a statistically large sample (i.e., many particles), the net influence of the random walk on the ensemble of particles must exhibit several properties:

1. The spreading (over that without dispersion) in the direction associated with each α is proportional to the square-root of α and ΔT.

2. The net displacement of the particles (compared to that without dispersion) is zero.

3 The net movement of the mass-weighted centroid of the particles is the same with or without dispersion.

Given these properties and the relationships between the random step length, mean and random velocity components, and time-steps, the following requirements can be deduced:

$$\sum_{i=1}^{n} \Delta S_i \approx 0 \tag{6}$$

$$\sum_{i=1}^{n} \frac{\Delta S_i}{\Delta T_i} \approx 0 \tag{7}$$

These summations must hold for a single particle as well as for the ensemble, and they must hold in each dimension. In order to simultaneously satisfy these pairs of relationships, the time-steps must be equal (i.e., if they are equal, then ΔT can be brought outside the summation).

Because these statistical relationships require equal time-steps, an immediate problem arises, regardless of whether classical Lagrangian particle tracking or the present method is used. Efficient implementation of a Lagrangian method requires a dynamically adjusted step length. Implementation of the present scheme results in time-steps varying over orders of magnitude, as a particle may pass through a cell near a vertex and cover a small distance in a correspondingly small time. If a constant time-step is required, then the smallest required time-step becomes a limiting factor and results in

impractical runtimes. It is for this reason that the random walk is not frequently used in large particle tracking applications or such applications are run on super computers.

2.2.7 Synchronous Time-steps

If variable time-steps are adjusted such that they become a constant time-step with sufficient frequency to represent a statistically significant sample, and the statistical calculations (i.e., the random walk steps) are performed on these synchronous time-steps, the limitation of uniform small time-steps can be overcome. This requires keeping two lists of particle history: one based on the constant time-step and one based on the variable steps (which periodically add-up to, and thus, synchronize with, the constant time-step). This can be accomplished with a Lagrangian tracking scheme by requiring the refined time-steps to be integer divisions of the coarse time-step. It is accomplished in the present scheme by summing and/or truncating sequential particle steps (i.e., *lurching*) through the cells so as to synchronize with a constant time-step. The present scheme uses an innovative bookkeeping algorithm to combine these two lists.

Computational experiments comparing the present scheme and an analytical solution indicate that the synchronous time-step need not be equal to the smallest cell traverse time. The synchronous time-step need only be small enough such that there are sufficient steps along a single particle track to represent a significant statistical sample (25 has proven to be sufficient for these experiments). If the statistical results are satisfied for each particle, then they will necessarily be satisfied for the ensemble. On a practical level, even if the number of synchronous steps along some of the particle tracks is smaller than this, the combined result for the ensemble will be satisfied if there are many particles (100 has proven to be sufficient for these experiments). The present scheme selects a synchronous time-step that is no more than $1/25^{th}$ of the total particle tracking time. A typical Lagrangian scheme may require 1000 steps for the average particle track; thus, the present scheme represents a significant improvement in runtime and storage.

3. PTRAX VERIFICATION

Three example problems are presented to test PTRAX. The first problem is qualitative and illustrates the effect of dispersion on a particle's flow path. The second problem illustrates a delineated well capture zone. The third problem provides a quantitative evaluation of concentration plumes generated by PTRAX.

3.1 PROBLEM 1: PARTICLE TRACKS

This test problem was constructed to validate the deterministic particle tracks (or flow paths) and to illustrate the effects of dispersion. The grid is evenly-spaced and rectangular. The elements are 3D bricks (hexahedra). The flow field is uniform as are the aquifer hydraulic properties.

Fig. 1 shows the tracks of 11 particles seeded near the origin (i.e., X = Y = Z = 0). In this first case there is no dispersion. The resulting tracks are straight lines. Fig. 2 shows the same 11 particles with dispersivities of $\alpha_X = 12$, $\alpha_Y = 6$, $\alpha_Z = 0$. Fig. 3 shows the track of 100 particles seeded at the origin with the same dispersion factors. The spreading effect produced by the random walk can be clearly seen in this figure.

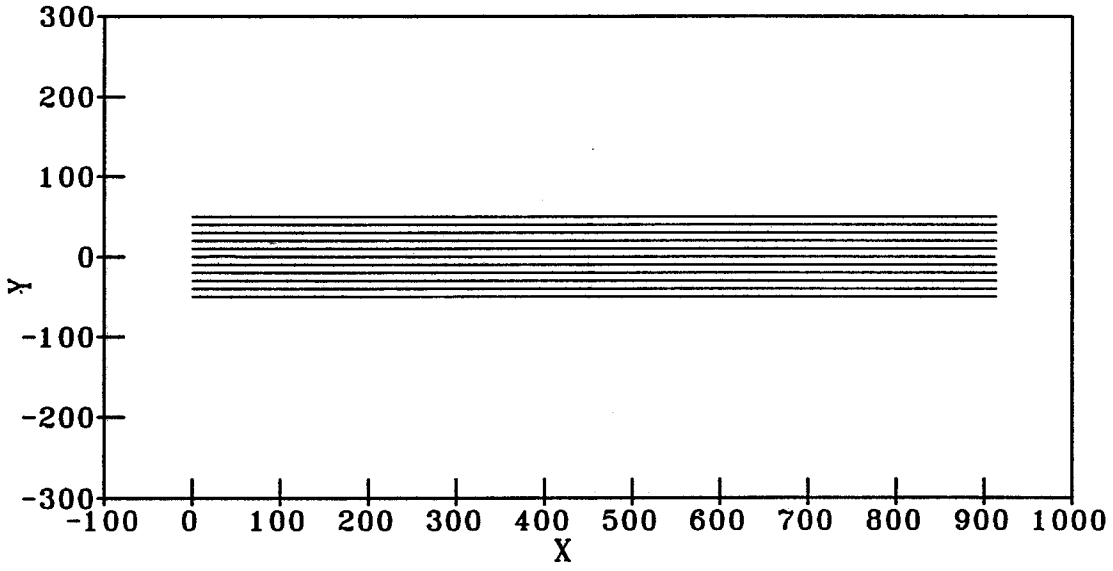

Fig. 1. Track of 11 particles without dispersion.

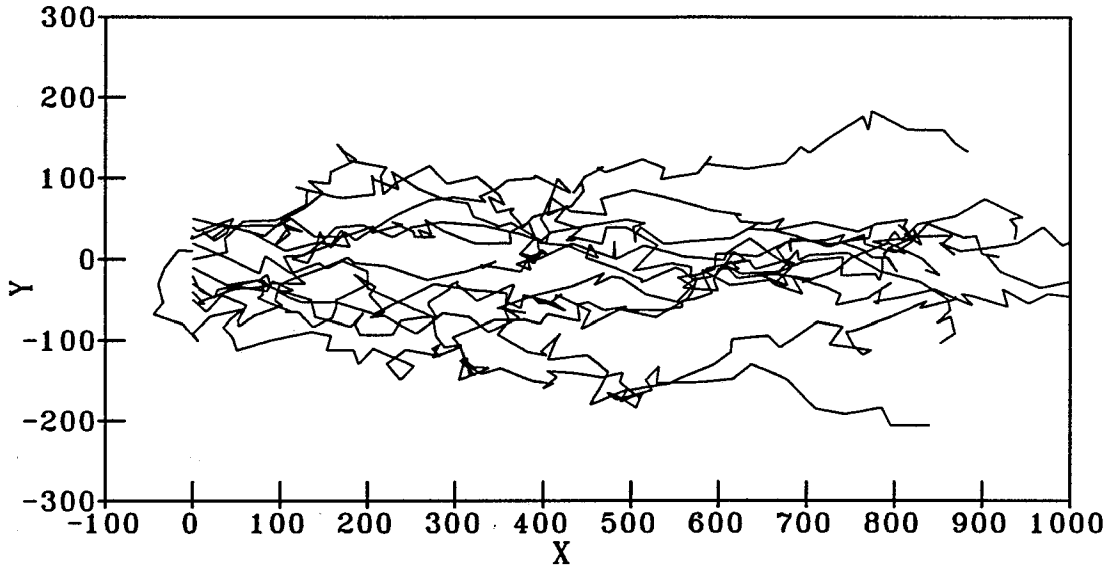

Fig. 2. Track of 11 particles with dispersion.

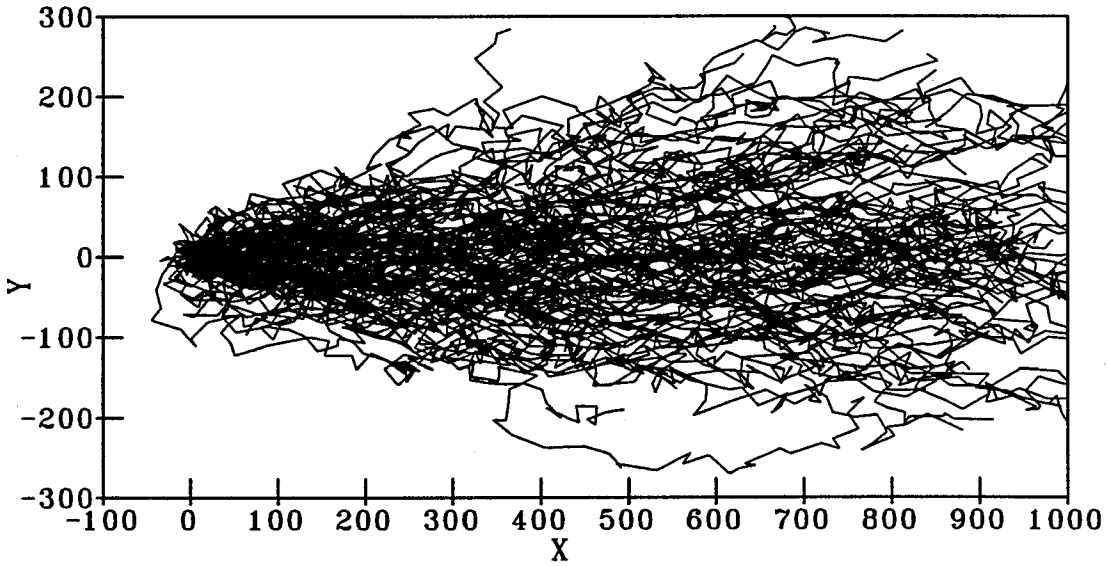

Fig. 3. Track of 1000 particles with dispersion.

3.2 PROBLEM 2: WELL CAPTURE ZONE

This test problem was constructed to illustrate a well capture zone and to demonstrate reverse particle tracking. Flow to pumping well was simulated with the FRAC3DVS code using a rectangular grid with variably-sized elements. The model domain extended from -100 m to 100 m in the X, from 0 to 100 m in the Y, and from 0 to 8 m in the Z directions. A fully-penetrating well pumping -10 m^3/day and having an effective radius of 6 m was located at the origin. A no-flow boundary was imposed along the plane where Y = 0, and a constant head boundary of 40 m was imposed along the other three sides. A no-flow boundary was imposed at the bottom and a constant recharge of 0.001 m/s was imposed at the top. The model discretization included 30,800 nodes and 27,702 elements.

After placing one particle in each element across the top the model domain, PTRAX simulated particle movement for 30 years and produced the path lines shown in Fig. 4. These path lines map a capture zone equal to a semi-circle with a radius of approximately 80 m. Integration of the flux across this capture zone produces a daily flux of 10 m^3/day, which is the amount discharged by the well. Reverse particle tracking was used to determine the effective well capture zone[10]. The particle tracks have been color-coded to show the capture radius. The red particle tracks are captured; whereas the blue are not.

3.3. PROBLEM 3: SIMULATION OF CONCENTRATION PLUMES

This third test problem was constructed to validate the dispersion component of PTRAX The example problem involves the spreading of a concentration point source[11] within a uniform flow field of 0.02 m/day. The uniform flow field was produced using FRAC3DVS (Therrien et al., 1995) with model boundaries of -95 to 605 m along the X-axis, -255 to 255 m along the Y-axis, and -34 to 34 m along the Z-axis. The grid network was constructed of equally-sized bricks, measuring 10 m by 10 m by 4 m along the X-, Y-, and Z-axis, respectively. This grid consisted of 60,690 elements and 66,456 nodes.

Four simulations were performed using α_x of either 3 or 12 m, a α_y between 0.3 and 12 m, and a constant α_z of 0.3 m. Simulations with $\alpha_x = 3$ began with 80,000 particles seeded at the origin (i.e., X = Y = Z= 0). Each of these particles were assigned a mass of 0.001 g, so that the averaged concentration in the element that initially contained all of the particles was 1 g/m^3. This initial concentration is obtained by dividing the 80 g by the brick volume of $10 \times 10 \times 4 = 400$ m^3 and a porosity of 0.2. As dilution increases with dispersion length, the number of particles used for simulations with $\alpha_x = 12$ was increased to 800,000. Each of these particles were assigned a mass of 0.0001 g in

---

[10] Reverse particle tracking is done by reversing the velocity field and seeding particles at the desired location (in this case, the well) and tracking the particles outward from that point.

[11] Theoretically, a point source would initially occupy an infinitesimal space and have an infinite concentration (mass/volume). For the purposes of numerical calculation, a small hexahedral source was used. This results in a finite width in the Y direction, even with a very small lateral dispersion length.

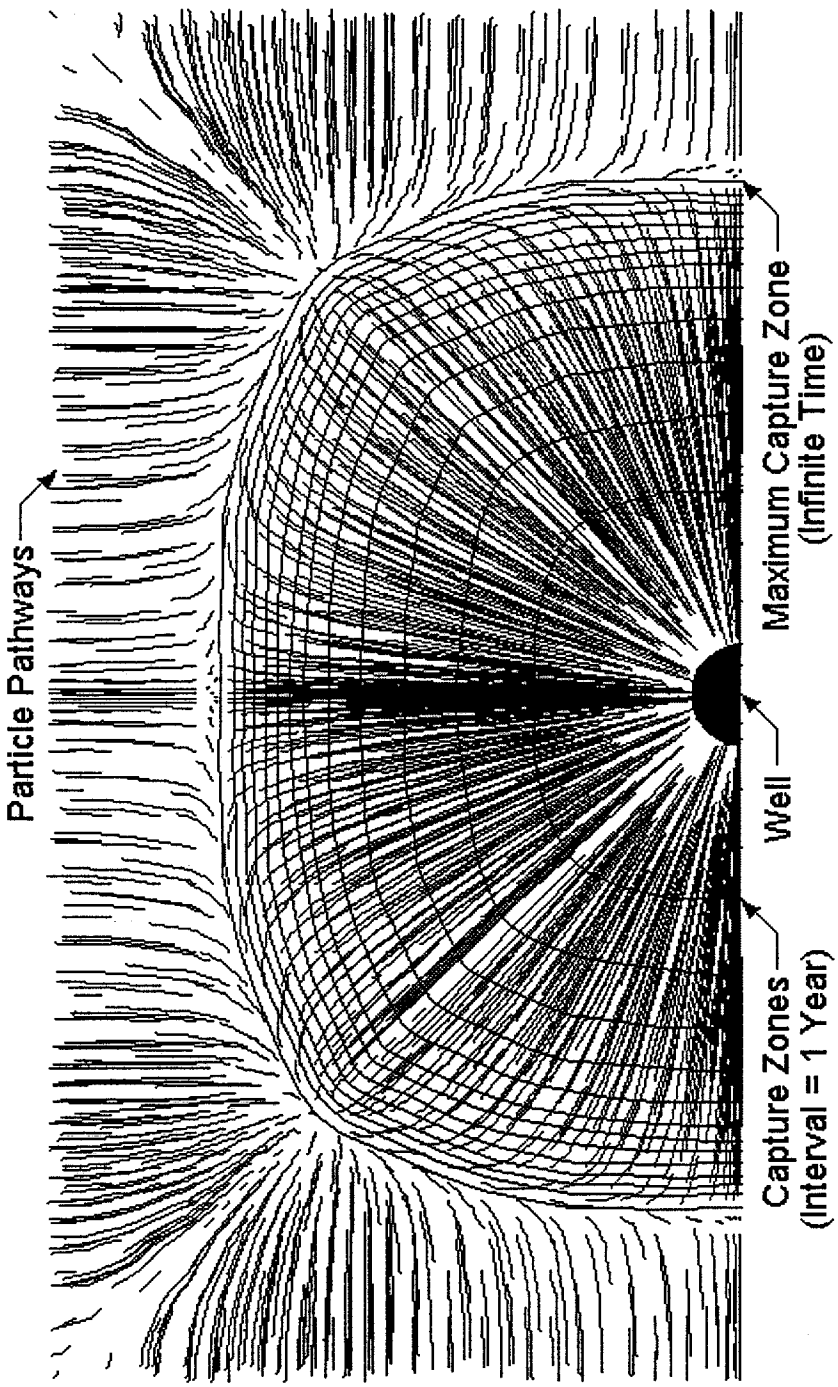

Fig 4. Particle track showing well capture

order to have the same initial concentration. A total of four simulations were made. For each simulation, concentration snapshots were produced at times of 15 years (5,475 days) and 30 years (10,950 days).

Figs. 5 through 8 show concentration contours in the X-Y plane at Z = 0 obtained with PTRAX and AT123D[12]. These figures show that the concentrations produced by PTRAX are consistent with AT123D. for the concentration contours in the X-Y plane at Z = 0. The slight difference in appearance between the numerical and analytical models is caused by the grouping of particle seeds into finite elemental volumes, as the initial conditions only approximate a point source. The differences between the PTRAX and AT123D simulations should continually diminish as the number of simulated particles and/or grid elements is increased.

An important evaluation criteria is proper simulation of the meanmovement and spreading of a plume. Mean plume movement can be calculated from the first moment of the plume concentration distribution (or the centroid of the mass). Plume spreading can be calculated from the second moment of the plume about the centroid. The mean movement and spreading of the plume were calculated using the numerical procedure explained by Davis (1986). The results of PTRAX and the analytical model, AT123D are listed in Table 1. The favorable comparison suggests that the random walk algorithm in PTRAX used to simulate dispersion is properly implemented and sufficiently accurate.

Table 1. Comparison of numerical model and analytical solution

Test Case	Dispersion length			Model	15 Years						30 Years					
	α_X	α_Y	α_Z		$\overline{X}$	$\overline{Y}$	$\overline{Z}$	σ_X	σ_Y	σ_Z	$\overline{X}$	$\overline{Y}$	$\overline{Z}$	σ_X	σ_Y	σ_Z
1	3	3	0.3	Numerical	109.4	0.4	0	25.6	25.8	8.0	218.7	0.8	0	37.1	36.9	11.3
				Analytical	109.5	0	0	25.6	25.6	8.1	219.0	0	0	36.2	36.2	11.4
2	3	0.3	0.3	Numerical	109.3	0.1	0	25.9	8.7	8.1	218.7	0.3	0.1	37.1	12.0	11.4
				Analytical	109.5	0	0	25.6	8.1	8.1	219.0	0	0	36.2	11.4	11.4
3	12	12	0.3	Numerical	109.1	0.5	0	51.1	51.1	8.1	217.7	1.4	0.1	73.5	72.8	11.4
				Analytical	109.5	0	0	51.3	51.3	8.1	219.0	0	0	72.4	72.4	11.4
4	12	1.2	0.3	Numerical	109.5	0.3	0.2	50.6	16.4	8.2	218.2	0.5	0.1	72.7	23.3	11.4
				Analytical	109.5	0	0	51.3	16.2	8.1	219.0	0	0	72.4	22.9	11.4

Notes: **Numerical** indicates the results of PTRAX
Analytical indicates the results of the AT123D analytical model
$\overline{X}$ is the mean distance traveled by the plume in the X direction (i.e., the first moment of the mass about X)
$\overline{Y}$ is the mean distance traveled by the plume in the Y direction (i.e., the first moment of the mass about Y)
$\overline{Z}$ is the mean distance traveled by the plume in the Z direction (i.e., the first moment of the mass about Z)
σ_X is the X dispersion (i.e., the second moment with respect to X of the mass about $\overline{X}$)
σ_Y is the Y dispersion (i.e., the second moment with respect to Y of the mass about $\overline{Y}$)
σ_Z is the Z dispersion (i.e., the second moment with respect to Z of the mass about $\overline{Z}$)

[12] AT123D is an analytical solution.

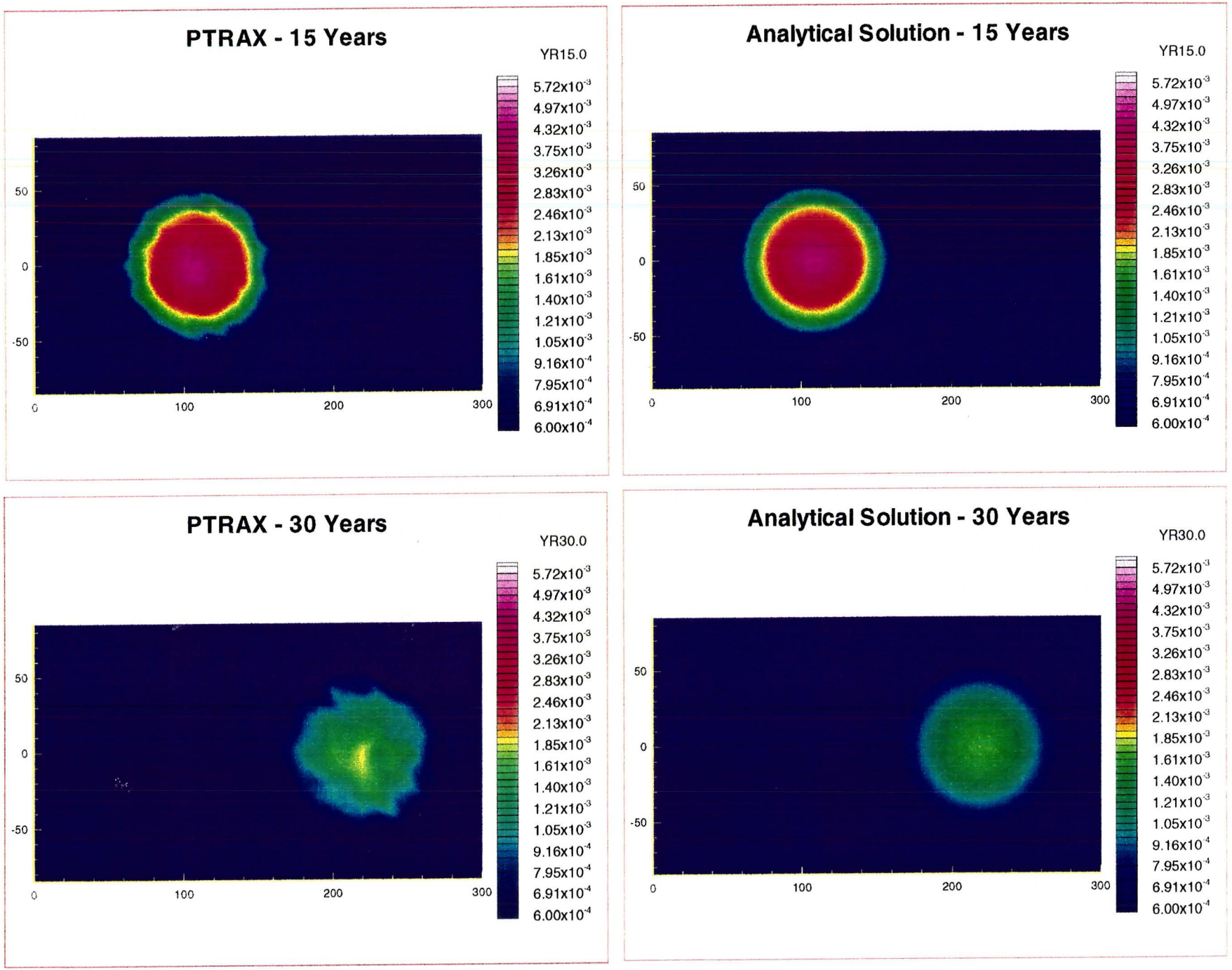

Fig. 5. Dispersion resulting from $\alpha x = \alpha y = 3m$.

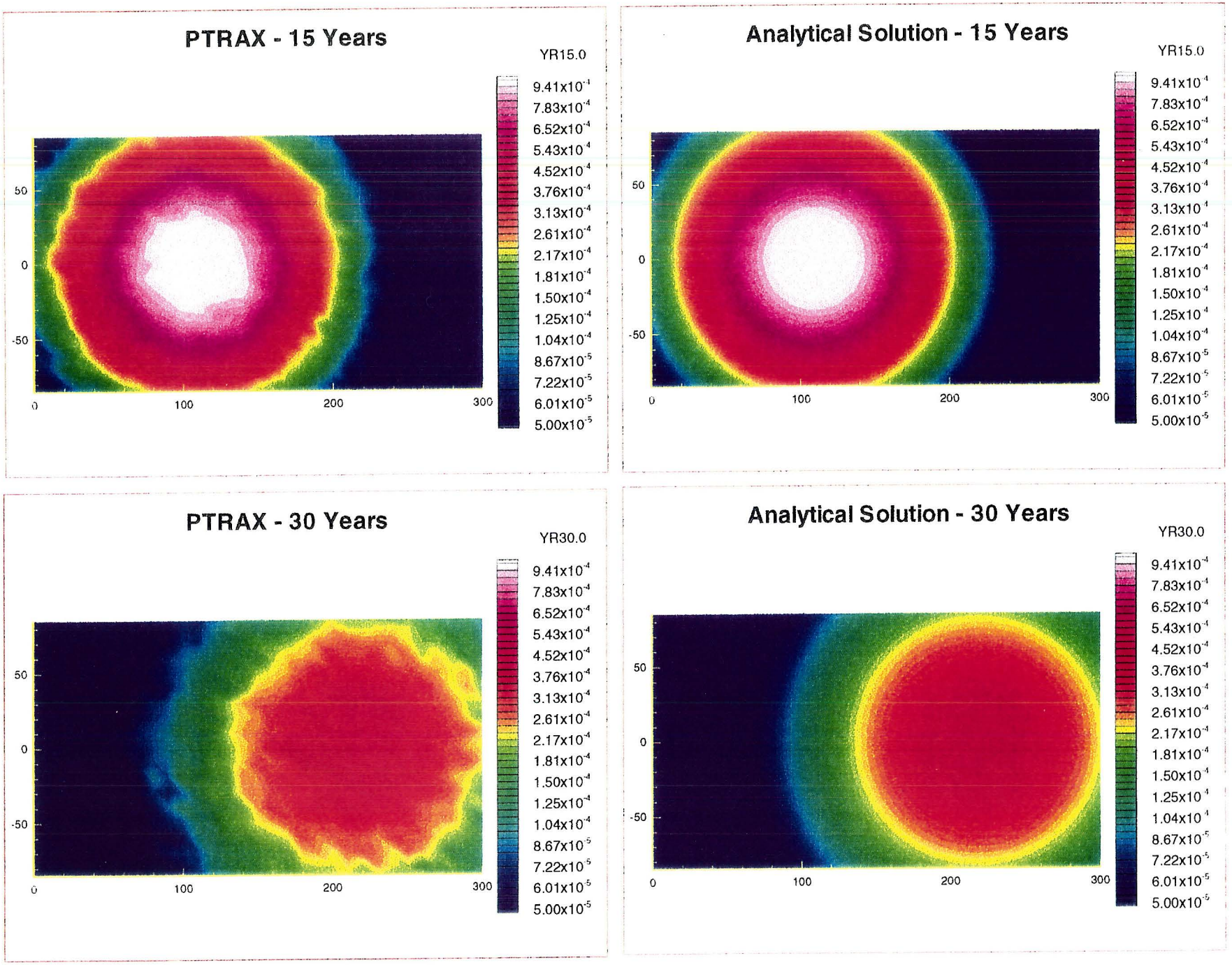

Fig. 6. Dispersion resulting from $\alpha x = \alpha y = 12m$.

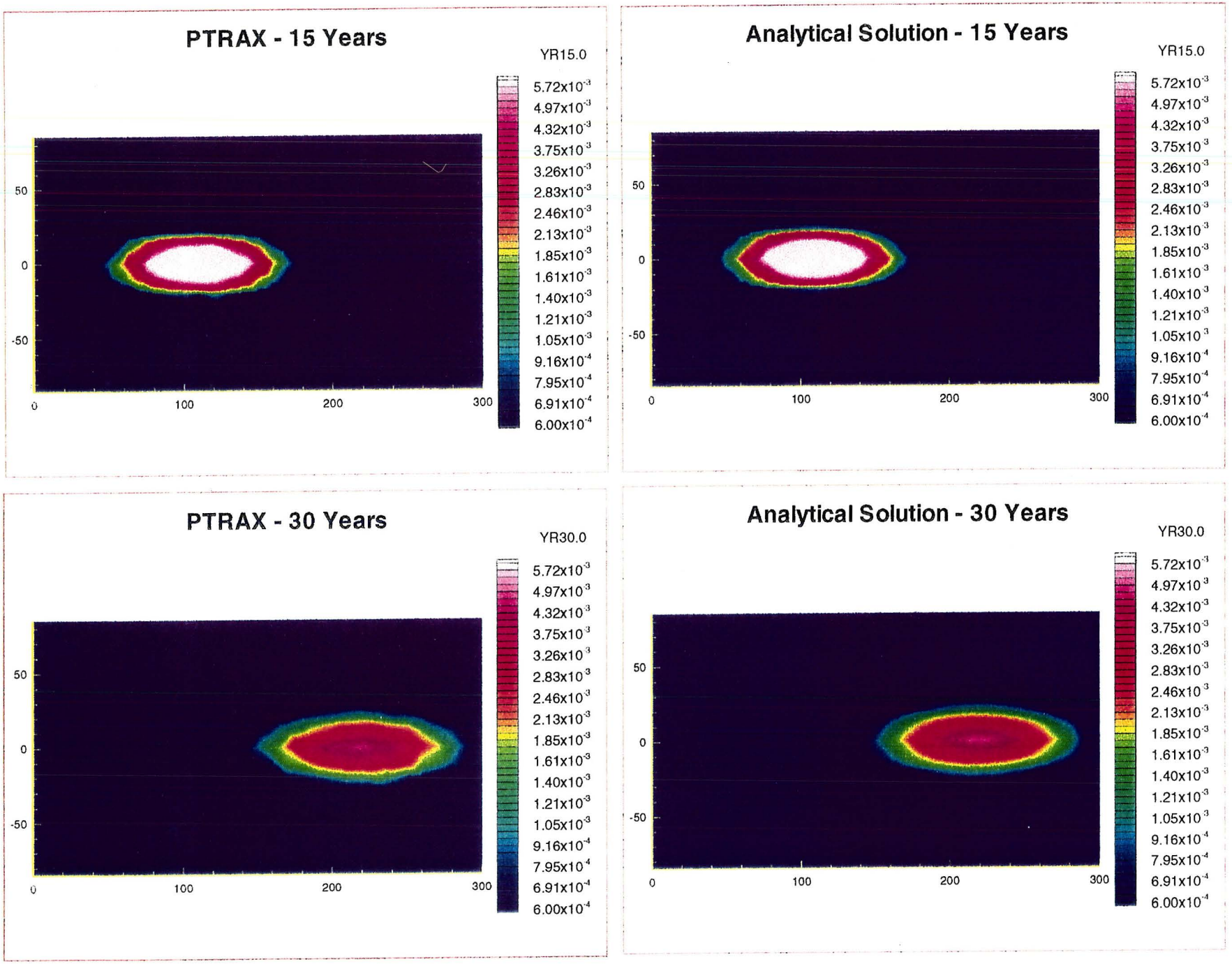

Fig. 7. Dispersion resulting from $\alpha x = 3$, $\alpha y = 0.3$.

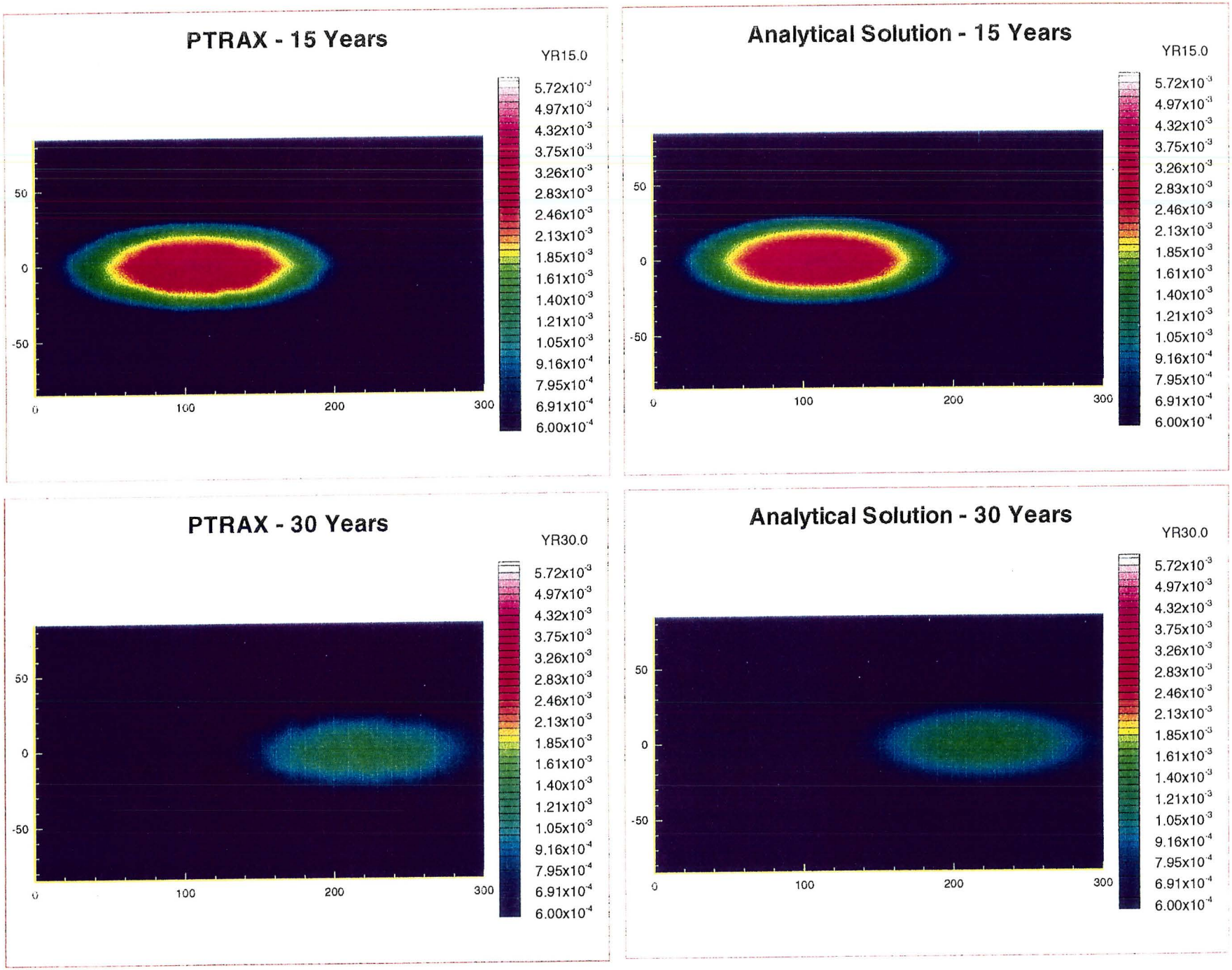

Fig. 8. Dispersion resulting from $\alpha x = 12$, $\alpha y = 0.3$.

4. PERFORMANCE

From the viewpoint of a practicing engineer, an extremely attractive feature of PTRAX is fast execution times. PTRAX was developed on and for the PC platform. Run times for the test cases with 80,000 particles and with 800,000 required approximately 7 and 70 minutes, respectively on a Pentium 90. Comparable transport simulations with finite-difference and finite-difference codes would require at least a ten-fold increase in the run time.

5. SUMMARY

In order to efficiently simulate solute transport and particle tracking in complex stratigraphy, ECE developed the particle-tracking code, PTRAX. PTRAX incorporates dispersion with a random walk component. PTRAX is compatible with grid meshes including prismatic, rectangular block, deformed block, and tetrahedral elements. The code contains particle tracking algorithms based on spatial integration that makes it significantly more efficient than the algorithms used by conventional particle tracking codes, such as MOC, RANDOM WALK, and MT3D. Three types of test problems have been presented to demonstrate that the algorithms in PTRAX used to create the flow paths and simulate dispersion have been properly implemented.

6. REFERENCES

Bear J, 1979. *Hydraulics of Groundwater.* McGraw-Hill, New York, NY.

Davis JC, 1986. *Statistics and Data Analysis in Geology.* John Wiley & Sons, New York.

Kinzelback W, Uffink G, 1991. *Random Walk Method and Extensions in Groundwater Modelling.* In: *Transport Processes in Porous Media.* Bear J, Corapcioglu MY, editors, Kluwer Academic Publishers, pages 761-787.

Konikow, Bredeheoft, 1978. *Computer Model of Two-Dimensional Solute Transport and Dispersion in Groundwater.* U.S. Geological Survey Water Resources Investigations, Book 7, Chapter C2, Reston, VA.

McDonald MG, Harbaugh AW, 1988. *Chapter A1: A Modular Three-Dimensional Finite-Difference Ground-Water Flow Model. U.S. Geological Survey Book 6 Modeling Techniques.* Scientific Software Group, Washington D.C.

Pollock DW, 1989. *Documentation of Computer Programs to Compute and Display Pathlines Using Results from the U.S. Geological Survey Modular Three-Dimensional Finite-Difference Ground-Water Flow Model.* U.S. Geological Survey, Reston, VA, Open File Report 89-381.

Prickett TA, Naymick TG, Lonnquist CG, 1981. *A Random Walk Solut Tranport Model for Selected Groundwater Quality Evaluations.* Illinois State Water Survey Bulletin 65, Campagne IL.

Scheibe TD, 1993. *Characterization of the Spatial Structuring of Natural Powerhouse Media and Its Impacts on the Subsurface Flow and Transport.* PhD thesis, Stanford University, Stanford, CA.

Therrien R, Sudicky EA, McLaren RG, 1995. *User's Guide to FRAC3DVS: An Efficient Simulator for Three-Dimensional, Saturated-Unsaturated Groundwater Flow and Chain-Decay Solute Transport in Porous or Discretely-Fractured Porous Formations.* Waterloo Institute for Groundwater Research, Waterloo, Ontario.

Yeh GT, 1981. *AT123D: Analytical Transient One- Two-, and Three-Dimensional Simulation of Waste Transport in the Aquifer System.* Oak Ridge National Laboratory, Oak Ridge, Tennessee, ORNL Report 5602, p. 37.

Zheng C, 1990. *PATH3D: A Ground Water Path and Travel-Time Simulator.* S.S. Papadolpulos & Associates, Inc., Rockville, MD.

Zheng C, 1992. *PATH3D: A Modular Three-Dimensional Transport Model, Version 1.5.* S.S. Papadolpulos & Associates, Inc., Bethesda, MD.