P- 34

Computer Techniques in Environmental Studies

ENVIROSOFT 88 - 2nd International Conference,
Porto Carras, Greece, September 1988

Editor:
P. Zannetti

Computational Mechanics Publications
Southampton Boston

Springer-Verlag Berlin Heidelberg
New York London Paris Tokyo

Development of a Portable FORTRAN 77 Code for Monte Carlo Particle Modeling of Atmospheric Diffusion (MC-LAGPAR II). Validation against Analytical Solutions and Tracer Experiments

G. Brusasca and G. Tinarelli
ENEL - DSR - CRTN, Via Rubattino, 54 - 20134 Milano, Italy
J. Moussafir and P. Biscay
EDF - DER - AEE - EAA / MA - 6 Quai Watier, 78400 Chatou, France
P. Zannetti
Aerovironment 825, Myrtle Avenue, Monrovia, CA 91016-3424, USA
D. Anfossi
CNR - Istituto di Cosmo-geofisica, Corso Fiume, 4 - Torino, Italy

INTRODUCTION

Monte-Carlo particle methods are an extremely appealing way to model the transport and diffusion in the atmospheric boundary layer, because they allow to take into account directly turbulence parameters measured during field experiments, as well as complex wind shear patterns, so that their validation is quite straightforward. A FORTRAN 77 version of the MC-LAGPAR code, originally writen by P. Zannetti using the APL language, was defined. The main development choices were the following : full portability (including Input / Output operations), high structuration, very high autodocumentation level. The challenge was to define a basic code that could easily be handled by several research teams, so that they could easily share information about new developments.

This paper describes the structure of MC-LAGPAR II, the physical assumptions, the existing options . In a previous paper (Brusasca and als., 1986,[1]), validations against analytical solutions and water tank experiments had been presented for the APL version. The FORTRAN version has been checked again against analytical solutions and tracer experiments in Karlsrühe. The inclusion of MC-LAGPAR II modules into the framework of a three dimensional complex-terrain time-dependent model is also discussed.

BACKGROUND

Several organizations provided direct or indirect support to the authors for the development of MC-LAGPAR II. They were: AeroVironment Inc. (AV), Monrovia, California, USA; the Kuwait Institute for Scientific Research (KISR); CRTN/ENEL (Research Center for Thermal and Nuclear Studies of the Public Electric Company of Italy in Milan, Italy; EDF/DER (Research Center of the French Electricity Board) in Chatou, France; and the Joint Research Center (JRC) of the ECC (Commission of the European Communities), Ispra Establishment, in Ispra (Varese), Italy.

The publication of this paper is one of a series of research activities in particle modeling in which several scientists were involved. The theoretical basis of MC-LAGPAR was defined by Zannetti (1981) [2] and subsequently expanded (MC-LAGPAR II) by Zannetti (1986) [4] . A first organized version of the computer code, written in APL language was developed by Zannetti and Al-Madani (1983,[3]) and further expanded, tested and documented by Brusasca et al. (1986) [1]. An improved FORTRAN version of the code (MC-LAGPAR II) was developed in 1987 at EDF Chatou and is described in this paper.

OVERVIEW OF ATMOSPHERIC DIFFUSION MODELS

The transport and diffusion of pollutants in the atmosphere is a complex modeling problem for two main reasons:

1. These processes may be studied at **several spatial and temporal scales**, which encompass several orders of magnitude (e.g., to 1000 km, a few minutes to several days). So there is a considerable variety of **flow models** to study each of these scales.

2. Conceptually there should be a clear separation between atmospheric **flow models**, which may include different levels of complexity for the representation of turbulence, and **transport and diffusion models** for passive or reactive contaminants (see Figure 1). This separation is important while designing a diffusion model because it allows to use the same transport and diffusion model with several different flow (or meteorological) models.

Transport and diffusion of pollutants in the atmosphere can be simulated by several numerical techniques, primarily in three categories: Gaussian straightline models, Eulerian models (grid models), Lagrangian models, where Gaussian models can be seen as a particular solution, under simplifying assumptions, of both the Eulerian and Lagrangian approach. In the next sections, we shall discuss Lagrangian and particle models.

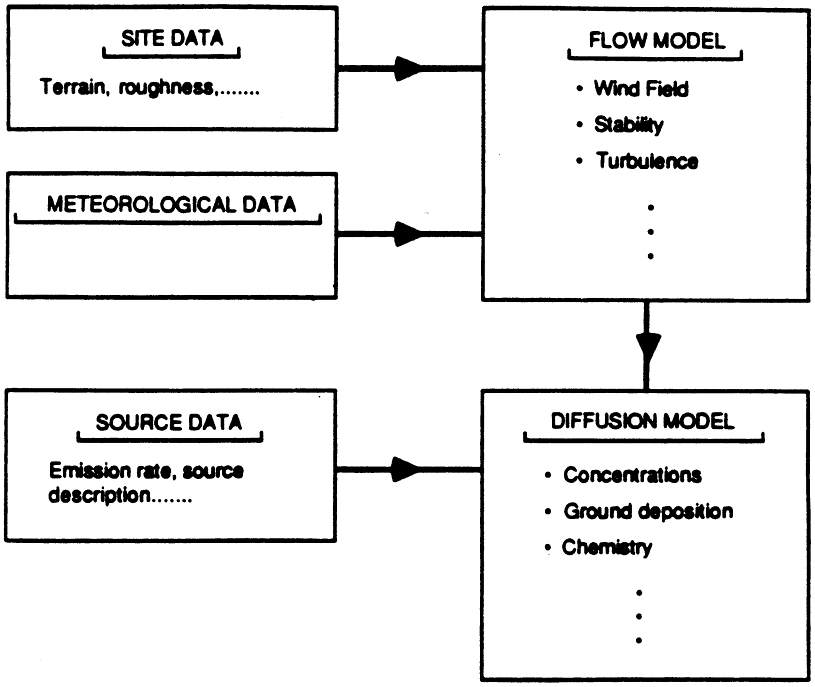


Figure 1 : Separation between flow models and diffusion models.

LAGRANGIAN MODELS

The fundamental Lagrangian equation for atmospheric dispersion of a single pollutant species is :

$$< c (r, t) > = \int_{-\infty}^{t} \int_{D} p (r , t \mid r', t') S (r',t') dV(r') dt' \quad [1]$$

where the integration in space (**dV(r')**) is performed over the entire atmospheric domain **D** ; **< c(r,t) >** is the ensemble average concentration at **r** at time **t** ; **S (r', t')** is the source term (mass per unit volume per unit time); **p (r , t | r', t')** is the probability density function (per volume) that an air parcel moves from **r'** at **t'** to **r** at **t**, where, for any **r'** and **t > t'**,

$$\int_{D} p (r , t \mid r', t') S (r',t') \; dV(r') = 1 \qquad [2]$$

The expression in Eq. (2) can be less than one when chemical or deposition phenomena are considered; otherwise mass conservation always requires the value to be equal to one. For a primary pollutant, **S(r',t')>0** only at points **r'** where the pollutant is released (e.g., exit point of stacks). For a secondary pollutant, **S(r',t')** can be nonzero virtually anywhere. For both primary and secondary pollutants, however, Eq. (2), which represents mass conservation, must be satisfied.

Since it is often difficult to evaluate the entire emission "history" , **S(r',t')** for **-∞ < t' < t**, Eq. 1 can be rewritten as the sum of two integral terms

$$< c (r, t) > \; = \; \int_{D} p (r , t \mid r',t_o) <c (r', t_o)> dV(r')$$

$$+ \int_{t_o}^{t} \int_{D} p (r , t \mid r', t') \; S(r',t') dV(r') dt' \qquad [3]$$

in which only the contribution of the sources during $t_o < t' < t$ needs to be included, since the first integral term accounts for the source contribution before t_o. However, Eq. (3) requires some estimate of the average concentration **<c>** at t_o throughout the computational domain.

The key parameter in the above equations is the probability density function **p**, which, for nonreactive pollutants, is a function of only the meteorology and the type of pollutant. Eqs. (1) or (3) represent a possible description of transport and diffusion processes, expressed in a probabilistic notation.

Different assumptions concerning the probability density function **p** allow the derivation of both Gaussian and **K**-theory equations. In particular, Seinfeld (1986) [9] shows that all Gaussian plume and puff formulas can be derived from the Lagrangian Eq. (1) under simplifying assumptions.

Several types of models can be classified as Lagrangian:

1. Lagrangian box, or trajectory, models, which are used for photochemical simulations.
2. Gaussian segmented plume models.
3. Gaussian puff models.
4. Particle models, which are fully discussed below.

PARTICLE MODELS

Eqs. (1) or (3) can be solved analytically or numerically. For example, as discussed before, a Gaussian distribution of **p**, together with other simplifying assumptions, allows the derivation of Gaussian plume and puff equations for <c>. More complex functions for **p** require numerical integrations.

An intuitive solution of Eqs. (1) or (3) can be given by assuming to follow a set of atmospheric trajectories of pollutant mass. Then, for each trajectory originating from **r'** at **t'**, we have **p (r , t | r', t') = 0** everywhere, except at the exact location **r = r***, where the trajectory passes at **t** thus giving **p(r*, t | r',t') = 1**. Therefore, if air parcel trajectories can be computed, the simple calculation of the density of the trajectory points, allows a proper calculation of Eqs. (1) or (3), i.e., estimate of <c>. This is the conceptual basis of a "particle" model, i.e., a model in which a set of "tracers" (or computational particles) are used to describe the dynamics of a system.

Particle modeling is a recent and powerful computational tool for the numerical discretization of a physical system. It has been particularly successful in a wide spectrum of applications (Hockney and Eastwood, 1981 [5]) that range from the atomic scale (electron flow in semiconductors, molecular dynamics) to the astronomical scale (galaxy dynamics), with other important applications to plasma and turbulent fluid dynamics. Using particle models, the temporal evolution of a physical system is described by the dynamics of a finite number of interacting particles.

Transport terms, whose correct numerical treatment is difficult with Eulerian (grid) models, are handled in a straightforward manner by particle models, since particles simply move following the main flow; for this reason, they are often called Lagrangian particles.

Particle models can be purely deterministic or can possess statistical (random) characteristics. In the first case, particle motion is generated by forces originated from particle interactions and/or potential fields. In the second case, semirandom pseudovelocities are generated using Monte-Carlo computer techniques. In the first case, the simulation of particle evolution with time is unique. In the second case, Monte Carlo techniques produce semirandom "perturbations" and, therefore, the dynamics of each particle represents just one realization of an infinite set of possible solutions. Particle models using Monte-Carlo methods are particularly important since, in several applications, they allow each particle to move independently of the others. Thus, they provide a computational algorithm that is generally faster than the corresponding deterministic computations, where interactions between neighboring particles need to be computed.

In air pollution applications using Lagrangian particle methods, the emitted material, gaseous or particulate matter, is represented by computer simulation particles and each particle is "moved" at each time step by an efficient pseudo velocity $\mathbf{u_e}$, which takes into account the three basic dispersion components: the transport due to the mean fluid velocity, the (seemingly) random turbulent fluctuations of wind components (both horizontal and vertical), and the molecular diffusion (if not negligible). In mathematical notation, we can define this pseudovelocity as :

$$\mathbf{u_e} = \overline{\mathbf{u_a}} + \mathbf{u_e}' \qquad [4]$$

where $\overline{\mathbf{u_a}}$ is the best estimate of the **average** Eulerian wind vector (transport) at the particle location, and $\mathbf{u_e}'$ is a "diffusivity velocity". In other words, the average velocity (a smoothly variable term) represents our deterministic understanding of the average transport process, based on Eulerian wind measurement interpolation or provided by a meteorological model, while $\mathbf{u_e}'$ is an artificial numerical perturbation, which is related to the turbulence intensities of those eddies that are not included in the average field.

Since, in Eq. (4), the average wind velocity is assumed to be known from measurements and/or meteorological model outputs, computing $\mathbf{u_e}'$ is the key problem of Lagrangian particle modeling. Two fundamental approaches can be followed: the deterministic and the statistical, as discussed below.

A typical example of the deterministic approach is given by the particle-in-cell method (Lange, 1978) [6], where, after some manipulation of the K-theory diffusion equation, the following relation is obtained

$$\mathbf{u_e}' = (- \mathbf{K} / \mathbf{c}) \ \nabla \mathbf{c} \qquad [5]$$

where **K** is the eddy diffusion coefficient and **c** the concentration, computed from the number of particles in a unit volume (or a cell) at the particle location.

This method requires partitioning the computational domain into cells in order to calculate **c**. It is able to duplicate **K**-theory dispersion results with the important feature of decreasing the numerical advection errors otherwise produced by finite-difference solutions.

Using this method, the motion of a single particle will be affected by the time-varying concentration field **c** ; i.e., by the positions of the other particles. In this sense, it is a particle-mesh method.

The statistical approach (Monte Carlo-type models) seems to be more flexible and appealing than the deterministic approach. According to the statistical approach, **ue'** is a semirandom component computed by manipulating computer-generated random numbers. The MC-LAGPAR model discussed here uses the statistical (Monte-Carlo) approach.

ADVANTAGES OF PARTICLE MODELS

Dispersion simulation by Lagrangian particles has been called "natural" modeling. These models do not need the input of artificial stability classes, empirical sigma curves, or diffusion coefficients that are practically impossible to measure. Instead, diffusion characteristics are simulated by attributing a certain degree of "fluctuation" to each particle, using, for example, the computer's capability to generate semirandom numbers. The basic advantages of this approach (e.g., Lange 1978 [6], Lamb et al. 1979a [10]) are :

- Compared with Eulerian models, particle models avoid the artificial initial diffusion of a point source in the corresponding cell and the advection numerical errors.

- Each particle can be tagged with its coordinates, source indicator, mass, activity, species or size class, allowing computation of wet and dry deposition, decay, and particle size distribution.

- If chemistry is required, a grid can be superimposed and concentrations in each cell computed by counting the particles of each species, allowing the use of any reaction scheme at each time step. A rigorous concentration computation should not just add up the number of particles in a given cell at a given time. In fact, concentrations should be computed using the total time spent by each particle in the receptor volume during each time step (as in Lamb et al., 1979b)[11].

- The meteorological input required can be inferred directly from measured data. The primary information needed is the variance of wind velocity fluctuations and the Lagrangian autocorrelation function, which can be related to Eulerian measurements (Hanna, in Nieuwstadt and vanDop, 1982 [7]).

- The validation against tracer field data of particle models is straightforward in that they allow to use the turbulence measurements without **K**-theory ad-hoc assumptions.

However, much research is still needed to extract, from meteorological measurements and our limited theoretical understanding of turbulence processes, the meteorological input required to run this model (i.e., the pseudovelocities to move each particle at each time step).

THE MC-LAGPAR II MODEL EQUATIONS

This section describes the basic equations of the MC-LAGPAR II model. The program allows the simulation of transport and diffusion of atmospheric pollutants (gases or particulate matter) by representing these dynamics with an arbitrary number of "fictitious" computer particles.

The emission term is represented by creating a selected number of new computational particles at each computational time step **Dt**, and injecting them into the computational domain at the source locations. If **Q** is the emission rate of a particular source and **Np** is the number of particles to be generated at each **Dt**, the pollutant mass represented by each particle will be **mp = Q Dt / Np**. The larger **Np**, the higher the possible resolution in the representation of the pollutant mass, hence of the concentration field to be computed.

In the current version of the program, new particles are generated within a rectangular "emission box". This box has appropriate sizes, which reflect the characteristics of both the source (i.e., its size) and the emission (e.g., the plume rise). More complex structures for the description of this initial "puff" will be considered in future releases. The box dimensions may be altered to describe point, line or area sources.

A particle is transported by the "average" wind vector **ua** (x,y,z,t) which, in general, varies with space (x, y, z) and time (t). Consequently, the coordinates (xp, yp, zp) of a particle p are updated at each time step Dt, using the components $u_x a$, $u_y a$, $u_z a$ of **ua** with respect to a generic rectangular coordinate system (x, y, z), and a simple time scheme (forward in time).

In order to simulate turbulent diffusion a particle is moved by a turbulent (or diffusivity) velocity **u'** (x, y, z, t, p) which, in general, varies with space (x, y, z), time (t), and is particle-dependent (p). In MC-LAGPAR II, the components u'_x, u'_y, u'_z of **u'** are updated at each time step $Dt = t_2 - t_1$ using the following Monte-Carlo scheme:

$$\begin{aligned} u_x'(t_2) &= \mathbf{f1}\, u_x'(t_1) + && u_x''(t_2) \\ u_y'(t_2) &= \mathbf{f2}\, u_y'(t_1) + \mathbf{f3}\, u_x'(t_2) + && u_y''(t_2) \qquad [6] \\ u_z'(t_2) &= \mathbf{f4}\, u_z'(t_1) + \mathbf{f5}\, u_y'(t_2) + \mathbf{f6}\, u_x'(t_2) + && u_z''(t_2) \end{aligned}$$

where u_x'', u_y'', u_z'' are uncorrelated zero-averaged Gaussian noises (random numbers) with standard deviations $\sigma(u_x'')$, $\sigma(u_y'')$, and $\sigma(u_z'')$, respectively.

This system provides a recursive computation of u_x', u_y' and u_z', which requires : (1) the knowledge of the initial values of u_x', u_y', and u_z' (they can be set equal to zero initially), (2) the parameters **f1** to **f6** , (3) the standard deviations $\sigma(u_x'')$, $\sigma(u_y'')$, and $\sigma(u_z'')$, that the model algebraically derives from user-provided meteorological input (See Zannetti, 1986) [4]. A computer program is used to generate the random numbers u_x'', u_y'', u_z''.

The meteorological input, to be provided by the user, that allows the calculations above is : (1) the standard deviation of the **u'** components : $\sigma(u_x')$, $\sigma(u_y')$, and $\sigma(u_z')$, (2) the autocorrelations of the u' components with time lag **Dt** : $\mathbf{r_x(Dt)}$, $\mathbf{r_y(Dt)}$, $\mathbf{r_z(Dt)}$, (3) the cross-correlations of the u' components with no time lag : $\mathbf{r_{xy}}(0)$, $\mathbf{r_{xz}}(0)$, $\mathbf{r_{yz}}(0)$.

It must be noted that the meteorological input above refers to the statistical description of the fluctuating component **u'**, which should not be seen as an intrinsic description of turbulence, but as a description of the unresolved component of the wind flow, i.e., the portion **u'** of the flow that is not contained in the average flow **ua**, as provided by the user.

In several cases a fully three-dimensional meteorological input, such as the one described above, is not available, and horizontal turbulence intensities are known only along and across the average wind flow.

In this case, using the subscripts a and c for the along-wing and the cross-wind horizontal components, respectively, we have, in a flux-coordinate system :

$$\mathbf{ua} = (u_a, 0, u_z) \qquad \mathbf{u'} = (u_a', u_c', u_z')$$

and the Monte-Carlo scheme can be simplified into the original MC-LAGPAR scheme:

$$\begin{aligned} u_a'(t_2) &= \mathbf{f1}\, u_a'(t_1) && + \; u_a''(t_2) \\ u_c'(t\,2) &= \mathbf{f2}\, u_c'(t_1) && + \; u_c''(t_2) \qquad [7] \\ u_z'(t\,2) &= \mathbf{f3}\, u_z'(t_1) + \mathbf{f4}\, u_a'(t_2) && + \; u_z''(t_2) \end{aligned}$$

which assumes that the only nonzero cross-correlation among **u'** components is between u'_a and u'_z. If this simplified scheme is used, the meteorological input to be provided by the user becomes : (1) the standard deviation of the **u'** components, $\sigma(u_a')$, $\sigma(u_c')$, and $\sigma(u_z')$, (2) the autocorrelation of the **u'** components with time lag **Dt**, $\mathbf{r_a(Dt)}$, $\mathbf{r_c(Dt)}$, $\mathbf{r_z(Dt)}$, (3) the cross-correlation Υ_{az} between u'_a and u'_z with no time lag. Then, the model will calculate the required parameters [**f1**, **f2**, **f3**, **f4**, $\sigma(u''_a)$, $\sigma(u''_c)$ and $\sigma(u''_z)$] from algebraical manipulations of the above input.

This scheme will also perform an appropriate projection of the u'_a and u'_c components, in the flux-coordinate system, into a fixed (x, y, z) coordinate system, since the average flow direction can generally vary with space and time. Clearly, for a wind homogeneously blowing along the x direction, and under the assumptions that $r_{xy}(0) = r_{yz}(0) = 0$, the scheme of Eqs. (6) and the scheme of Eqs. (7) are identical.

The model allows to account for plume rise effects by adding an appropriate vertical displacement to a particle in its initial dispersion phase. Several standard plume rise formulations can be selected by the user. However, it would be much more interesting to develop a new 'particle' formulation for plume rise, where an additional vertical velocity, depending on particle initial buoyancy and on its age, for example, could be added to the dynamics of each particle.Work along those lines is under way.

Ground deposition phenomena can be simulated at each time step **Dt** using two alternative approaches: either by exponentially decreasing the mass of all those particles that have been moved below terrain, using some user-defined time constant, either by a selective probabilistic reflection, in which some of these particles will be reflected and the rest of them will be deposited on the ground.

If the second approach above is used, particles deposited on the ground can be resuspended back into the computational domain or permanently absorbed by the ground. In this case the user should provide a time scale of the resuspension process, and each deposited particle will have a probability of being resuspended linked to that time scale. However, if a particle remains deposited on the ground for a period of time greater than a user-specified critical value, the particle will be permanently deposited.

SOME MC-LAGPAR II MODEL FEATURES

We describe here some important model features, that were designed to allow easy implementation of modifications into the computer package.

First, a **three level** distinction in introduced throughout the code for geometrical domains :

1 . **Topography horizontal domain.** The horizontal area (rectangular), for which topographic data are available, is defined through specification on a topographic file, where terrain and roughness matrixes may be given.

2 . **Particle domain** (horizontal and vertical). This is the geometrical domain inside which the particles are tracked. It may differ from the topography domain horizontally, in order to allow the analysis of subsections of the topography domain. The particle domain is computationally important for the following reason : once a particle is out of that domain, it is still moved with the other particles until the memory zone it occupies is needed by some new particle, and the old particle is canceled. Through specifying the dimensions of the particle domain, the user may control the global size of the problem (total number of particles moved), and choose whether he prefers a good resolution close to the source or a global description of the field.

3. **Concentration grid domain** (horizontal and vertical).This domain is divided in cells to allow a grid estimation of concentrations. Different grid estimations of concentrations may be produced with the same particle dynamics computation : hence, it is necessary to be able to move the concentration box independently from the particle domain definition. In complex terrain, the cells are defined in a terrain-following coordinate system, where the elevation for an horizontal location of the lowest cell of the concentration grid is computed through interpolation on the topography gridded data. Consequently, in complex terrain, the horizontal concentration mesh, should have at least the same (or a higher) resolution than the topographic mesh.

Once the particle domain is defined, **each particle is "tagged"**, in the computer program, with a series of variables that describe its dynamic characteristics (e.g., location, mass, etc.). Among these characteristics is the "status" of each particle which indicates that the particle is currently :

- not yet generated
- active (i.e., moving within the particle domain)
- temporarily deposited, where resuspension is still possible
- permanently deposited, where no resuspension can occur
- outside the horizontal boundary of the domain
- above the top of the domain
- active, but resuspended
- active, but still affected by dynamic plume rise

Different computational strategies are selected, according to the current status of each particle.

Particle age is used for several calculations. In MC-LAGPAR II the age is recorded in different ways, depending upon the status of the particle, as follows:

- active particles: age is the particle life since generation
- deposited particles: age is the time since particle deposition time
- resuspended particles: age is the particle life since resuspension

The meteorological input, which affects particle dynamics, can be provided in many forms. They are:

- **homogeneous meteorology**, in which the meteorological parameters do not vary with space (flat terrain case)

- **meteorological profiles**, in which the meteorological parameters vary only with the altitude z (flat terrain case)

- **three-dimensional meteorological input**, in which it is assumed that a 3D meteorological model is used to calculate the meteorological fields. In this case, in flat terrain the meteorological input needs to be provided on a three-dimensional grid (x,y,z), while, in complex terrain, the same input needs to be provided on a grid (x,y,s) where s is a terrain following vertical coordinate.

PORTABILITY OF THE COMPUTER PACKAGE

Portability is the quality of a computer package that allows it to be easily transferred from one machine to another without any major change in the computer code itself. System independence is a condition to achieve a reasonable level of portability.

In theory any good scientific language can be used to achieve portability, so that one could choose the language that best fits his needs. We chose FORTRAN 77 as the programming language for MC-LAGPAR-II, despite all its shortcomings, in view of the experience gained with the first version of the code, written in APL. Even if many modern programming languages, such as C or ADA, are more efficient than FORTRAN, they are still known well by a relatively small number of people, thus reducing the actual portability of the code. Another good reason for choosing FORTRAN is the idea that some version of this particle model will be used some day as a set of subroutines in the framework of three-dimensional meteorological models, which are generally written in FORTRAN. We restricted ourselves to standard FORTRAN 77 instructions, without making use of any system-dependent extension.

However, one may consider four major sources of trouble to achieve portability for a FORTRAN program:

1. User interface and Input/Output operations should be such that no system-dependent or terminal-dependent feature is needed.

2. File handling (file naming conventions and dynamic file allocation processes are generally system-dependent).

3. Graphical output.

4. Functions that do not exist in the FORTRAN language (e.g., random number generation for the present code).

Problem 1 is linked to the fact that the standard I/O "interface" for a FORTRAN code is the connection between logical units and files. The simplest way to go for a package is to say that the code will read input files and write output files. This approach is not quite user-friendly in all circumstances. Users are more and more familiar with terminal interfaces that help them to command code operations. This is why we included a set of portable FORTRAN I/O subroutines which emulate unrolling menus with an acceptable degree of flexibility for conversational command of the code, and allow the user to store this information for further batch execution of the model. This set of routines was taken from a general purpose data-management package developed at EDF (ADSO V4.0).

Problem 2 is partly solved by saying that the user must connect user-defined file-names to the logical units defined in the code. Again, this is not really user-friendly. The user must recall which files are needed by the program, and which logical units he must connect to them, as opposed to dynamic file allocation, when the program opens and closes directly the files he needs. But dynamic file allocation is in conflict with system-independence because the file names have to be given inside the program so that their names will be depending of the name structure of a specific operating system. For the MC-LAGPAR-II package the file handling problem is solved in the following way: the first file that the package reads on a fixed logical unit gives the logical units to be used for the various devices (terminal output, keyboard input) or files used by the program.In that file the user may also specify with a flag if he wants file allocation to be done externally (default), or internally (dynamic allocation). In the latter case the user must complete two dummy subroutines with the proper instructions to achieve dynamic allocation on his own computer system.

Problem 3 comes from the fact that the FORTRAN language does not include graphic instructions, so that the simpler way to go is to limit the output of the model to output files, that have to be read by a user-written graphical post-processor in order to obtain a graphical display of the results. This means that no on-line graphics are available.Since the developers of the MC-LAGPAR-II package have been writing several graphic subroutines on their own systems to have on-line graphics available, the calling structure of these routines have been kept in the whole program package, but the system-dependent calls to specific graphic packages have been suppressed (dummy graphic subroutines).The full graphic subroutines are also provided separately. This should enormously facilitate the work of the user in emulating the graphical output of MC-LAGPAR-II with his own graphical software.

Problem 4 is the more cumbersome, because it deals with the very heart of Monte-Carlo simulations, which call for random number generation. Random-number generation is a tricky problem, and there is no standard FORTRAN instruction to do it. There are hundreds of library subroutines or system-dependent subroutines available to do the job, but some of them are not so random as they claim to be.The main idea of all those subroutines is to initiate a random walk inside some memory zone of the machine. In a very simplistic way, the longer the walk and the larger the memory zone, the more random the generated numbers will be. This system dependency is solved in MC-LAGPAR II in two ways. First, there is one single name in the whole code to identify the random number generating routine (RANGEN). Second, this routine is dummy (does nothing) and has to be completed by the proper calls to a uniform random-number generator in the [0,1] interval. Versions of the "RANGEN" subroutine calling several well-known general-purpose scientific libraries, or system dependent generators are included in the code.The seed for random number generation is transmitted throughout the code, with a double precision number.

STRUCTURE OF THE MC-LAGPAR-II COMPUTER PACKAGE

The MC-LAGPAR-II package consists of a main program and several subroutines. According to the previous discussion, these subroutines may belong to three different groups:

1. NUCLEUS: these include all the strictly portable FORTRAN subroutines that may be called by the main program at runtime.Some of these routines are dummy, in that they do not actually perform their function (e.g.: the RANGEN subroutine).

2. SYSDEP: this group includes different versions of system dependent subroutines, provided as a help for the user. For example, the user will find in the SYSDEP group several versions of RANGEN (all with the same subroutine name) corresponding to different system-dependent implementations of the random number generator.

3. GRAPHS: this group includes different versions of graphic library dependent subroutines, provided as a help for the user. Typically, the user will find in GRAPHS several versions of the TRAPAR subroutine (on-line plot of particle cloud) corresponding to different graphic library dependent implementations of the same plot.

It is commonplace to say that a well-born software package should be structured in several subroutines, each one of them performing a specific part of the computing process. However, there are two important requirements for a FORTRAN package that dramatically increase portability and ease of debugging, at the price of putting some more constraints on the programming process. These requirements are:

1. functional independence of each subroutine.
2. variable memory requirements for each subroutine.

Functional independence means that a given subroutine only interacts with other programs through one single path : its calling arguments input-output list. Since one of the goals of the MC-LAGPAR development was to develop a FORTRAN nucleus that could easily be linked to advanced meteorological models, it was decided that all data interchange between subroutines would go through input-output list for each subroutine, and that no commonblocks would be used.

Variable memory requirements for each subroutine means that all the arrays used in a subroutine should be of variable dimensions, and that those dimensions should be passed to the subroutine in the list of calling arguments. This allows an extreme flexibility for the software package : the same FORTRAN code may run on 200000 particles on a CRAY computer and on 5000 particles on a PC. The only changes to be done are minor modifications in the main program, and not a systematic review of all subroutines to change all array dimensions.

Other structured programming conventions of lesser importance were used in the present development, such as extended documentation of input-output list, systematic error handling, single exit point for each subroutine. An important feature is the presence of a standardized Input/Output system for code command, which may work in two modes.

In **conversational mode**, the code will write questions on the user screen terminal, and read input answers from the user keyboard. When the program is run in this mode, an exact track of all I/O operations on the keyboard unit is written to a file.The same file can later be used as an input command file for the batch execution mode, described below, or printed to recall which commands were given in the last execution.

In **batch execution mode**, the sequence of questions and answers will be directly read from the command file, and the program will write an image of the dialog to an output file.

In conversational mode, ALL input from keyboard is handled in the form of answers to unrolling menus, that are reasonably user-friendly. Unrolling menus are less user-friendly than full screen input menus, but full screen menus are generally poorly portable, and make the code very difficult to modify for new input. The unrolling menus are generated by a set of four strictly portable FORTRAN subroutines, that may be called anywhere in the code. While generating the command file if required, these routines do not write only the user answers on the file, which would result in an inextricable sequence of numbers, they also write the program questions - the unrolling menus - before each user answer.

The main interest of this I/O system for the user is related to the actual practice of the use of a numerical model. During the first stage, the user is helped by conversational input-output in designing his run, and deciding which options he wants to use. At this stage, the computational cost of a trial may be kept minimal by choosing very few time steps and very few particles. Once the run configuration is fully prepared, the user may switch to batch mode, and edit the command file (using any editor working on his own operating system) if he needs to change some of the answers he made in the conversational session. This is particularly useful for repetitive use of the code with single parameter variation for sensitivity studies for instance. On the contrary, purely conversational command is of invaluable help in debugging, checking completeness, and testing "what if?" scenarios.

Another advantage is of relatively little interest to the end user, but essential for the developers and for any person wishing to add a particular parameter or option inside the code itself. Since the subroutines driving unrolling menus are provided with the package, any simple new input value may be added by adding a call to one of these routines where desired. Once this modification is added, it will be implicitly supported for the command file generation in conversational mode and for the command file decoding in batch mode. Thus, the addition of new control parameters invokes one single modification of the code.

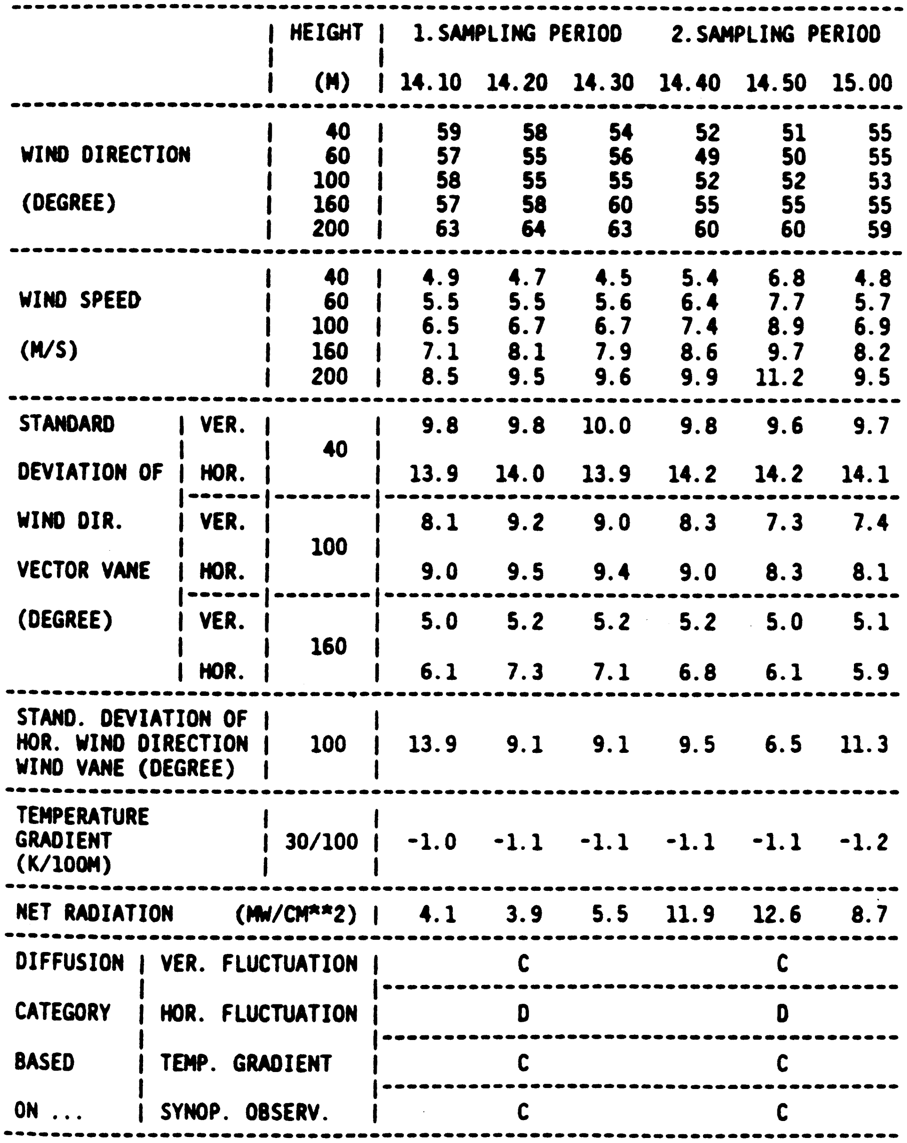
PRESENT VALIDATION STAGE

Model validation has been performed against standard analytical solutions for homogeneous turbulence. This process has allowed a good estimation of how many particles were needed in general to achieve stationarity on the concentration results, for stationary meteorological conditions, as a function of cell size. The total number of particles varies between 10000 and 100000.

A systematic validation against field experiments has been undertaken by ENEL and CNR, using data from an extended tracer experiment held at the Karlsrühe Nuclear Research Center (Thomas & als.,1983) [8] . This experiment was selected because most of the meteorological data needed to run the model were directly measured on a 200 m high meteorological tower. All the meteorological datasets include wind speed and direction at five levels and standard deviations (horizontal and vertical) at three levels. The flux coordinate version of the MC-LAGPAR formulation could then be used without the need of a turbulence model to estimate the input variables.

Extended reports on this validation should be available in a near future. As an example, we present here the results corresponding to the first sampling period of experiment number 55. The meteorological data are reported on Table 1, taken from the Karlsrühe report. The ground level isocontours, in micrograms per cubic meters, are plotted on Figure 2, and the concentration data are reported on Table 2. The numbers on Figure 2 refer to sensor numbers as in Table 2, but we only reported the sensors that gave measurements above threshold. Note that the contour interval is 4. micrograms per cubic meter. In this simulation, the situation is close to neutral, with medium to strong winds, and a constant direction. The particle domain was chosen as a 6 x 6 kilometers square, and the concentration cells are rectangular, with horizontal size 200 meters and vertical size 50 meters.The source position is indicated by an X on the plot. Source coordinates with respect to the particle domain were 5000, 5000, 160 meters. About 10000 particles were used. The agreement in this case is quite satisfactory, and better than with any gaussian straightline model.

The validation work is now proceeding along two lines. First, ENEL and EDF intend to examine the connection of this particle model to measurements provided by a Doppler acoustic sounder. This work involves more research on the parameterization of lagrangian time scales that appear in the fi coefficients, and hence on the use of some turbulence model to generate them, because only part of the input parameters is measured by commercial acoustic sounders. One of the interesting aspects of particle modeling is the possibility of separate validations for the transport and diffusion model (as in the Karlsrühe validation) and for the turbulence model to be used with the acoustic sounder data. Anyhow, it is hoped that the model will at least allow to take into account very strong wind direction shears which occur in weak winds situations. A second work which is planned by EDF is the validation against SF6 tracer experiments in complex terrain, using as an input 3D wind fields generated by an objective analysis model.



		HEIGHT	1.SAMPLING PERIOD			2.SAMPLING PERIOD		
		(M)	14.10	14.20	14.30	14.40	14.50	15.00
WIND DIRECTION (DEGREE)		40	59	58	54	52	51	55
		60	57	55	56	49	50	55
		100	58	55	55	52	52	53
		160	57	58	60	55	55	55
		200	63	64	63	60	60	59
WIND SPEED (M/S)		40	4.9	4.7	4.5	5.4	6.8	4.8
		60	5.5	5.5	5.6	6.4	7.7	5.7
		100	6.5	6.7	6.7	7.4	8.9	6.9
		160	7.1	8.1	7.9	8.6	9.7	8.2
		200	8.5	9.5	9.6	9.9	11.2	9.5
STANDARD DEVIATION OF WIND DIR. VECTOR VANE (DEGREE)	VER.	40	9.8	9.8	10.0	9.8	9.6	9.7
	HOR.		13.9	14.0	13.9	14.2	14.2	14.1
	VER.	100	8.1	9.2	9.0	8.3	7.3	7.4
	HOR.		9.0	9.5	9.4	9.0	8.3	8.1
	VER.	160	5.0	5.2	5.2	5.2	5.0	5.1
	HOR.		6.1	7.3	7.1	6.8	6.1	5.9
STAND. DEVIATION OF HOR. WIND DIRECTION WIND VANE (DEGREE)		100	13.9	9.1	9.1	9.5	6.5	11.3
TEMPERATURE GRADIENT (K/100M)		30/100	-1.0	-1.1	-1.1	-1.1	-1.1	-1.2
NET RADIATION		(MW/CM**2)	4.1	3.9	5.5	11.9	12.6	8.7
DIFFUSION CATEGORY BASED ON ...	VER. FLUCTUATION		C			C		
	HOR. FLUCTUATION		D			D		
	TEMP. GRADIENT		C			C		
	SYNOP. OBSERV.		C			C		

Table 1 : Meteorological Data for experiment 55./ April 5, 1978
Table taken from Thomas et als (1983) [8].

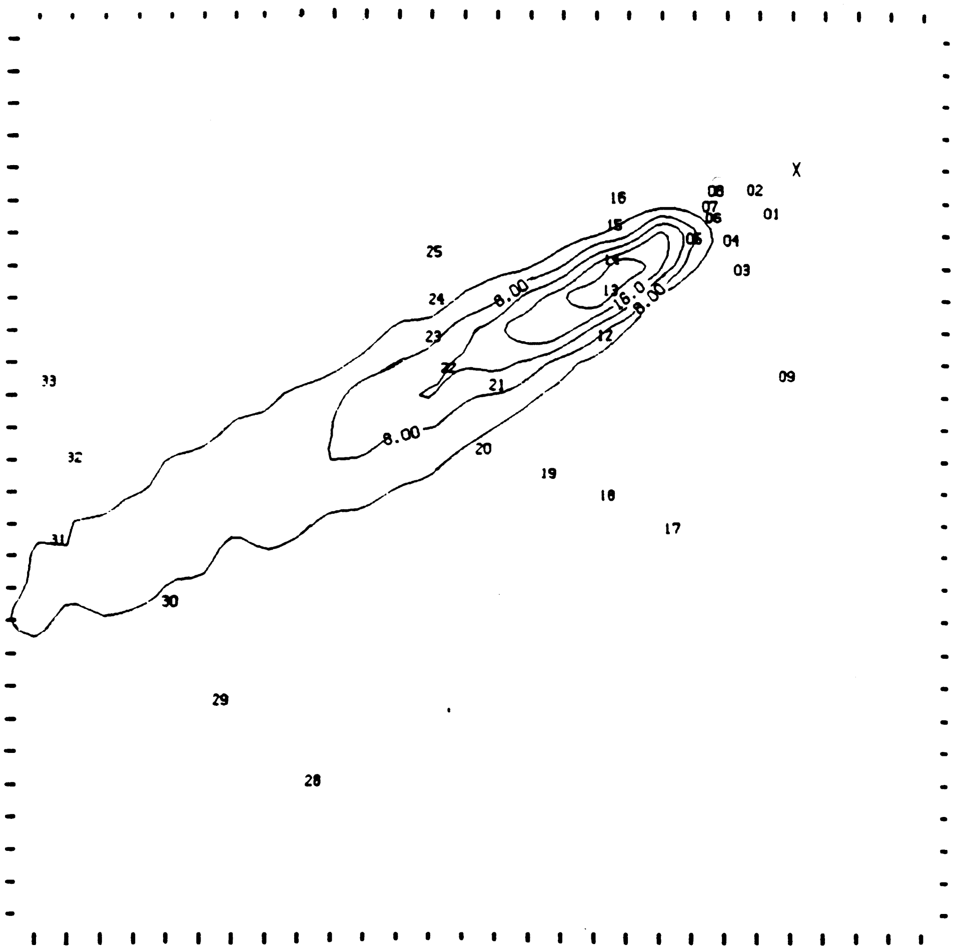


Figure 2 : Isocontours of ground level concentration (microgram/cubic meter) computed by MC-LAGPAR II for experiment 55 (1st part) Isopleths are spaced 4 micg/m^3, Envelope value is 24. micg/m^3. The numbers refer to tracer sensors as in Table 2.

Table 2 : Tracer Data for experiment 55./ April 5, 1978
Tracer is CF_2Br_2 Emission rate is 22.40 g/s
Data for sampling period 1.

Sensor number	Distance (m)	Alpha (Deg.)	Conc (mic-g/m^3)
05	770.	236.	6.681
06	600.	240.	0.329
12	1565.	229.	5.345
13	1370.	237.	21.845
14	1270.	244.	10.220
20	2580.	228.	2.035
21	2270.	234.	18.760
22	2460.	240.	7.024
23	2455.	245.	3.813
24	2345.	250.	0.136
30	4675.	235.	5.582
31	5060.	243.	1.481

CONCLUSIONS

The development of a common FORTRAN version of a Monte Carlo particle model was the starting point for a cooperative work on this subject between several organizations. A lot of research is still necessary before particle models can be used as operational tools, as in emergency preparedness for instance. On the other hand they are extremely promising research tools, and the first intercomparisons with field data are encouraging. It is hoped that the present approach will help each group in keeping up with new developments and sharing the cross-validation tasks.

REFERENCES

[1] . Brusasca G., Tinarelli G., Zannetti P., Anfossi D. (1986) , Monte Carlo Simulation of Plume Dispersion in Homogeneous and Nonhomogeneous Turbulence. in ENVIROSOFT 86 - (Ed. Zannetti P.) Proceedings of the International Conference on Development and Application of Computer Techniques to Environmental Studies, Los Angeles, U.S.A., Nov.1986 Computational Mechanics Publications, Southampton, U.K.

[2] . Zannetti P. (1981), Some aspects of Monte Carlo type modeling of atmospheric turbulent diffusion, in proceedings of the 7 th Conference on Probability and Statistics in Atmospheric Sciences, AMS. Monterey, CA Nov.1981.

[3] . Zannetti P. and N. Al-Madani (1983), Simulation of transformation, buoyancy and removal processes by Lagrangian particle methods, in proceedings of the 14 th ITM Meeting on Air Pollution modeling and its Applications, Copenhague, Denmark, September 1983.

[4] . Zannetti P. (1986) , Monte Carlo simulation of Auto and Cross-correlated Turbulent Velocity Fluctuations (MC-LAGPAR II Model), Environmental Software, Vol.1, pp. 26-30

[5] . Hockney R.W. and Eastwood J.W. (1981). Computer simulations using particles. McGraw-Hill, Inc.

[6] . Lange R. (1978), ADPIC - A three dimensional particle-in-cell model for the dispersal of atmospheric pollutants and its comparison to regional tracer studies, Journal of Applied Meteorology, Vol 17, pp. 320-329

[7] . Hanna S.R. (1982), Applications in air pollution modeling. Chapter 7 in Atmospheric Turbulence and Air Pollution Modeling (Ed. Nieuwstadt F.T.M. and Van Dop H.), pp. 275-310

[8] . Thomas P., Hübschmann W., Schüttelkopf, Vogt S. (1983) Experimental Determination of the Atmospheric Dispersion Parameters at the Karlsruhe Nuclear Research Center for 160 and 195 m Emission Heights (Part 1 : Measured Data) KFK Report 3456 Kernforschungszentrum Karlsruhe

[9] . Seinfeld J.H. (1986) , Atmospheric Chemistry and Physics of Air Pollution, John Wiley & Sons, New York, U.S.A.

[10] . Lamb R.G. , Hogo H., Reid L.E. (1979a), A Lagrangian Monte Carlo model of air pollutant transport, diffusion and removal processes. 4th Symposium on Turbulence, Diffusion and Air Pollution, AMS. Reno, Nevada, Jan.1979

[11] . Lamb R.G. , Hogo H., Reid L.E. (1979b), A Lagrangian approach to modeling air pollutant dispersion; development and testing in the vicinity of a roadway, EPA Research Report EPA/600/4-79-023