PNL-10586
UC-402

VALDRIFT 1.0–A Valley Atmospheric Dispersion Model with Deposition

K. J. Allwine
X. Bian
C. D. Whiteman

May 1995

Prepared for
the U.S. Department of Agriculture Forest Service
Missoula Technology and Development Center
under a Related Services Agreement
with the U.S. Department of Energy
Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Operated for the U.S. Department of Energy
by Battelle Memorial Institute

Battelle

PNL-10586

PACIFIC NORTHWEST LABORATORY
operated by
BATTELLE MEMORIAL INSTITUTE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC06-76RLO 1830

Printed in the United States of America

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615) 576-8401. FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

The contents of this report were printed on recycled paper

PNL-10586
UC-402

VALDRIFT 1.0 - A VALLEY ATMOSPHERIC DISPERSION MODEL WITH DEPOSITION

K. J. Allwine
X. Bian
C. D. Whiteman

May 1995

Prepared for
the U.S. Department of Agriculture Forest Service
Missoula Technology and Development Center
under a Related Services Agreement
with the U.S. Department of Energy
Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Richland, Washington 99352

PREFACE

Under an Interagency Agreement with the U.S.D.A. Forest Service (FS), Missoula Technology and Development Center, the Pacific Northwest Laboratory has developed a valley atmospheric dispersion model, VALDRIFT, to predict the concentration and deposition patterns arising from the off-target drift of aerially applied pesticides. Aerial pesticide applications, conducted using fixed-wing aircraft and helicopters, are frequently required for the protection of forest health, and the prediction of the off-target drift of such pesticides in complex terrain is a continuing problem which, up to now, has not been adequately modeled.

The development of VALDRIFT was initiated by Robert Ekblad (FS-retired) to improve the performance of existing pesticide drift models in complex terrain. The regularity of the diurnal variations of the wind flow in mountain valleys under certain synoptic-scale meteorological conditions offered a scenario which lent itself to phenomenological modeling, where the dominant atmospheric processes are modeled explicitly in a highly parameterized fashion. Since aerial spraying in mountain valleys occurs relatively frequently in both the Rocky and Appalachian Mountains, the development of this operational model was pursued.

This report provides detailed technical information and a user's guide for the valley atmospheric dispersion model, VALDRIFT version 1.0. The completion of this model is a key milestone in our strategic plan to improve predictions of pesticide drift from spraying operations in complex terrain. VALDRIFT will be incorporated into the Forest Service Cramer-Barry-Grim (FSCBG) modeling framework. The addition of VALDRIFT to this modeling framework will extend the range of modeling capability in a framework that is well-known to a large group of forest managers.

Dr. Harold Thistle, Project Manager
Missoula Technology and Development Center
Fort Missoula, Building 1
Missoula, Montana 59801

SUMMARY

VALDRIFT version 1.0 is an atmospheric transport and diffusion model for use in well-defined mountain valleys. It is designed to determine the extent of drift from aerial pesticide spraying activities, but can also be applied to estimate the transport and diffusion of various air pollutants in valleys. The model is phenomenological - that is, the dominant meteorological processes governing the behavior of the valley atmosphere are formulated explicitly in the model, albeit in a highly parameterized fashion. The key meteorological processes treated are: 1) nonsteady and nonhomogeneous along-valley winds and turbulent diffusivities, 2) convective boundary layer growth, 3) inversion descent, 4) nocturnal temperature inversion breakup, and 5) subsidence. The model is applicable under relatively cloud-free, undisturbed synoptic conditions and is configured to operate through one diurnal cycle for a single valley.

The inputs required are the valley topographical characteristics, pesticide release rate as a function of time and space, along-valley wind speed as a function of time and space, temperature inversion characteristics at sunrise, and sensible heat flux as a function of time following sunrise. Default values are provided for certain inputs in the absence of detailed observations. The outputs are three-dimensional air concentration and ground-level deposition fields as a function of time.

The basic computational approach of VALDRIFT is the solution of a one-dimensional (along-valley) conservation equation for the pesticide or pollutant species for each of a number of "flowtubes" aligned along the valley. Lateral and vertical turbulent diffusion, vertical advection, variable emissions, and deposition are treated as source/sink terms in the species conservation equation. The model conserves species mass and total air mass within the flowtubes, both for the along-valley flow in each flowtube and for the entire valley, using the mass conservation equations.

The scientific foundations of VALDRIFT are documented in this report along with detailed operating instructions. The model input requirements are discussed, and detailed descriptions of the outputs are provided. An example model application is used to illustrate typical input values and model output files. The code is written in standard Fortran so that the model can be easily implemented on various computers.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Harold Thistle and Mr. Robert Ekblad, the U.S. Department of Agriculture Forest Service project managers, for their encouragement and suggestions throughout this work. We would also like to thank Mr. Stefano Ionescu-Niscov, of the Atmospheric Sciences Department of Pacific Northwest Laboratory, for the significant contributions he made during the initial phases of the code development. This work was supported by the U.S. Department of Agriculture Forest Service under a Related Services Agreement with the U.S. Department of Energy (DOE) under Contract DE-AC06-76RLO 1830, Interagency Agreement 5E52P29. Pacific Northwest Laboratory is operated for DOE by Battelle Memorial Institute.

CONTENTS

FIGURES

TABLES

1.0 INTRODUCTION

1.1 BACKGROUND

In 1990, the U.S. Department of Agriculture Forest Service (FS) initiated a project to select an existing complex terrain wind flow model for incorporation into existing aerial pesticide spray models AGDISP (Bilanin et al, 1989) and FSCBG (Teske et al, 1993), to facilitate the estimation of spray drift from target areas located in regions of complex terrain (Ekblad and Barry, 1990). This project was undertaken as part of the Forest Pest Management, Technology Development Program in cooperation with FS Region 6 and the FS Missoula Technology and Development Center. The project, "Selection and Verification of a Complex Terrain Wind Flow Model for Spray Transport," was to be accomplished in three phases: 1) model selection, 2) model testing and validation, and 3) model merger and technology transfer. Results of the first phase of the project were summarized by Ekblad et al. (1991), as follows.

> "An in-depth analysis of the current knowledge of mountain meteorology demonstrated that air movements in mountain valleys during the morning transition period are well enough understood to develop useful planning tools to improve aerial spraying. A systematic review of mesoscale meteorological models was made to select a model for adaptation. We concluded that a phenomenological model was needed because of the rapid changes in the convective boundary layer in the mountainous terrain. The model selected was VALMET. It is phenomenological, mesoscale, has good documentation, is operational, and uses computer resources compatible with AGDISP and FSCBG... Several additions will be made to VALMET to incorporate varying emissions and winds, cross-valley circulations, and subsidence."

The original VALMET model, developed for the U.S. EPA by Whiteman and Allwine (1985), is an atmospheric dispersion code for estimating ground-level concentrations from an elevated point source located in a valley. The model, written in Fortran, treats the nighttime down-valley transport of a continuous steady release in steady winds, and the subsequent fumigation of the elevated plume to the surface during the morning transition period. The model is applicable under relatively cloud-free, undisturbed synoptic conditions. The inputs required are the valley physical characteristics, the source characteristics, the nighttime down-valley wind speed at the release height, temperature inversion characteristics at sunrise, and sensible heat flux as a function of time following sunrise. The outputs are ground-level concentrations on a valley cross-section.

Adaptation of the VALMET code to the forest pesticide drift problem involved major changes to the code structure, so that the modified model has been given a new name, VALDRIFT. This report documents the VALDRIFT version 1.0 model and its use in estimating drift from aerial spraying operations.

1.2 OVERVIEW OF VALDRIFT

VALDRIFT treats the transport, diffusion, and deposition of an inert substance released from multiple point and/or line sources in a valley atmosphere, where the sources may be elevated or at ground level and the release rate from each source can vary with time. The released substances can be either gases or aerosols having negligible settling velocities so that they can be assumed to follow the flow. VALDRIFT is based on a formulation given by Allwine (1992) and is configured to operate through one diurnal cycle for a single well-defined mountain valley with relatively steep sidewalls (sidewall angles ~ 10° - 90°).

VALDRIFT is especially suited for estimating drift of pesticides from aerial spraying operations that would usually be accomplished in the morning hours after first light, under relatively cloud-free, undisturbed synoptic conditions. The spraying would generally continue until convective boundary layers and up-slope flows develop sufficiently to render it difficult to place the spray where needed in the canopy. On an undisturbed clear day the spraying would generally be terminated in mid to late morning. VALDRIFT would be used to estimate the drift throughout the day of the pesticide remaining in the air after application to the canopy.

The basic approach for using VALDRIFT to estimate the extent of the drift from spraying operations is briefly outlined below, with more details given in Sections 2 and 3.

1. Use topographic maps to define the modeling domain
2. Create a text file to describe the valley terrain
3. Create a text file to describe the release locations and rates

4. Create text files to describe the valley wind field and its evolution with time, using wind data, if available
6. Construct a run specification file
7. Run VALDRIFT
8. Create desired model output text files from the binary model output file using a post-processor program

This report provides sufficient technical and operational information on VALDRIFT to enable a user to set up and operate the model for their specific application. The first section of this report, the Technical Description section, provides the technical foundations of the model. Following is the User's Guide section that gives detailed instructions for operating VALDRIFT, using an example simulation illustrating the model's application to spray drift problems. The next section describes the basic code architecture and provides a brief narrative description of each of the model subroutines. The last section of this document provides conclusions and recommendations concerning the use of the VALDRIFT model. A report entitled "Effects of Valley Meteorology on Forest Pesticide Spraying" prepared by Whiteman (1990) for the FS is given in Appendix A as a basic reference to the physical processes governing drift in valley atmospheres. A complete listing of the Fortran source code is provided in Appendix B.

2.0 TECHNICAL DESCRIPTION

This section discusses the technical basis of the VALDRIFT version 1.0 model in sufficient detail to allow the user to understand the uses and limitations of the model. The technical discussion of VALDRIFT is divided into four main topical areas, which are discussed in Sections 2.1 through 2.4. These topical areas are Basic Conceptual Model, Basic Mathematical Model, Computational Grid, and Meteorological Model. Table 2.1 provides the names of the subroutines corresponding to these topics. For more technical detail the user can refer to the documented computer code given in Appendix B.

TABLE 2.1. VALDRIFT Subroutines Within Each Topical Area

TOPIC	RELEVANT SUBROUTINES
2.1 BASIC CONCEPTUAL MODEL	none
2.2 BASIC MATHEMATICAL MODEL	INITIALZ, MASSBALN
2.2.1 Emission Source/Sink	CSSOURCE, READSRCE, CASOURCE
2.2.2 Deposition Source/Sink	CPSOURCE, SLFLUXES
2.2.3 Diffusion Source/Sink	CDSOURCE, DIFFUSIV, STABCORE
2.2.4 Numerical Method	COURANTC, SLVECONC
2.3 COMPUTATIONAL GRID	GNDLOCAT, LINLOCAT, MAKEGRID, PNTLOCAT, INITIALZ, STABCORE, MAKEGRID
2.4 METEOROLOGICAL MODEL	INITIALZ, MAKEGRID, WNDSCONV
2.4.1 Wind Fields	VFLWRATE, INTEGRAL
2.4.2 Morning Transition	DESCENTS, SOLRFLUX, SLFLUXES

2.1 BASIC CONCEPTUAL MODEL

Many population centers around the world are located in valleys. Air quality problems routinely occur in mountain valleys from pollutant sources located within or near the valleys (Hewson and Gill, 1944; Tyson, 1969; and, Wanner and Hertig, 1984). Over the past several years, numerous meteorological and tracer experiments have been conducted to investigate the dispersive characteristics of valley atmospheres (Start et al., 1975; Willson et al., 1983; Gudiksen et al., 1984; Gryning and Lyck, 1983;

Clements et al., 1989; Whiteman, 1989; and Doran et al., 1990). Also, the dynamic and dispersive behavior of valley atmospheres has been investigated theoretically and with numerical models (McNider, 1981; Bader and McKee, 1985; Vergeiner et al., 1987; Segal et al., 1988; Bader and Whiteman, 1989; and Allwine, 1993). These, and other such studies, have led to the identification of key physical processes (e.g., up- and down-valley winds, up- and down-slope winds, convective boundary layer growth, inversion descent, tributary flows, cross-valley flows, and interactions with above ridgetop winds) governing dispersion in valleys, and a more complete understanding of these physical processes is emerging from ongoing research efforts.

Whiteman (1990) discusses the effects of valley meteorology on forest pesticide spraying for the FS. He identified the dominant physical processes controlling the dispersion of pesticides in valley atmospheres, when the valley is undisturbed by external weather conditions such as cloudiness or high upper-level winds. Whiteman's report is included in Appendix A, and should be reviewed to gain a basic understanding of the physical processes governing the transport and diffusion of material released into a valley atmosphere. The key physical processes discussed by Whiteman are

1. along-valley flows
2. slope flows
3. turbulent diffusion
4. nocturnal temperature inversion breakup
5. convective boundary layer growth
6. inversion descent
7. subsidence
8. cross-valley circulations
9. interactions with above-ridgetop winds

Any computer model developed to predict dispersion of material released into valley atmospheres must incorporate (in addition to source characteristics, transformations and deposition) the effects of the dominant physical processes on transport and diffusion. One approach (used in VALDRIFT) is to describe each physical process in terms of empirical, semi-empirical or theoretical relationships. An example is Whiteman and McKee's (1982) parameterization of the temperature inversion breakup in a valley atmosphere during the morning transition period (this parametrization is included in VALDRIFT). The parameterization is based on the bulk thermodynamic behavior of the valley atmosphere as the valley is heated by the sun. The physical processes currently treated explicitly in VALDRIFT are

1. along-valley flows
2. turbulent diffusion
3. nocturnal temperature inversion breakup
4. convective boundary layer growth
5. inversion descent
6. subsidence

Slope flows, cross-valley circulations, interactions with above ridgetop winds, and tributary flows can be incorporated into the VALDRIFT modeling framework in the future, but are presently not handled explicitly by the model code.

2.2 BASIC MATHEMATICAL MODEL

VALDRIFT is based on a valley dispersion model developed by Allwine (1992) and is configured to operate through one diurnal cycle for a single valley. The basic approach is to solve a one-dimensional (along-valley) species conservation equation for each of a number of "flowtubes" aligned along the valley as illustrated in Figure 2.1. Figure 2.1 shows a section of valley divided into nine flowtubes. The flowtube in the center of the valley is outlined to show it's conformity with the shape of the valley. Interactions among flowtubes can occur and are handled through source/sink terms in each species conservation equation. The species conservation equation, to be solved for each flowtube, is

$$\frac{\partial(AC)}{\partial t} + \frac{\partial(\dot{V}C)}{\partial S} - \Gamma_{cs} - \Gamma_{cp} - \Gamma_{cc} - \Gamma_{ca} - \Gamma_{cd} = 0 \qquad (2.1)$$

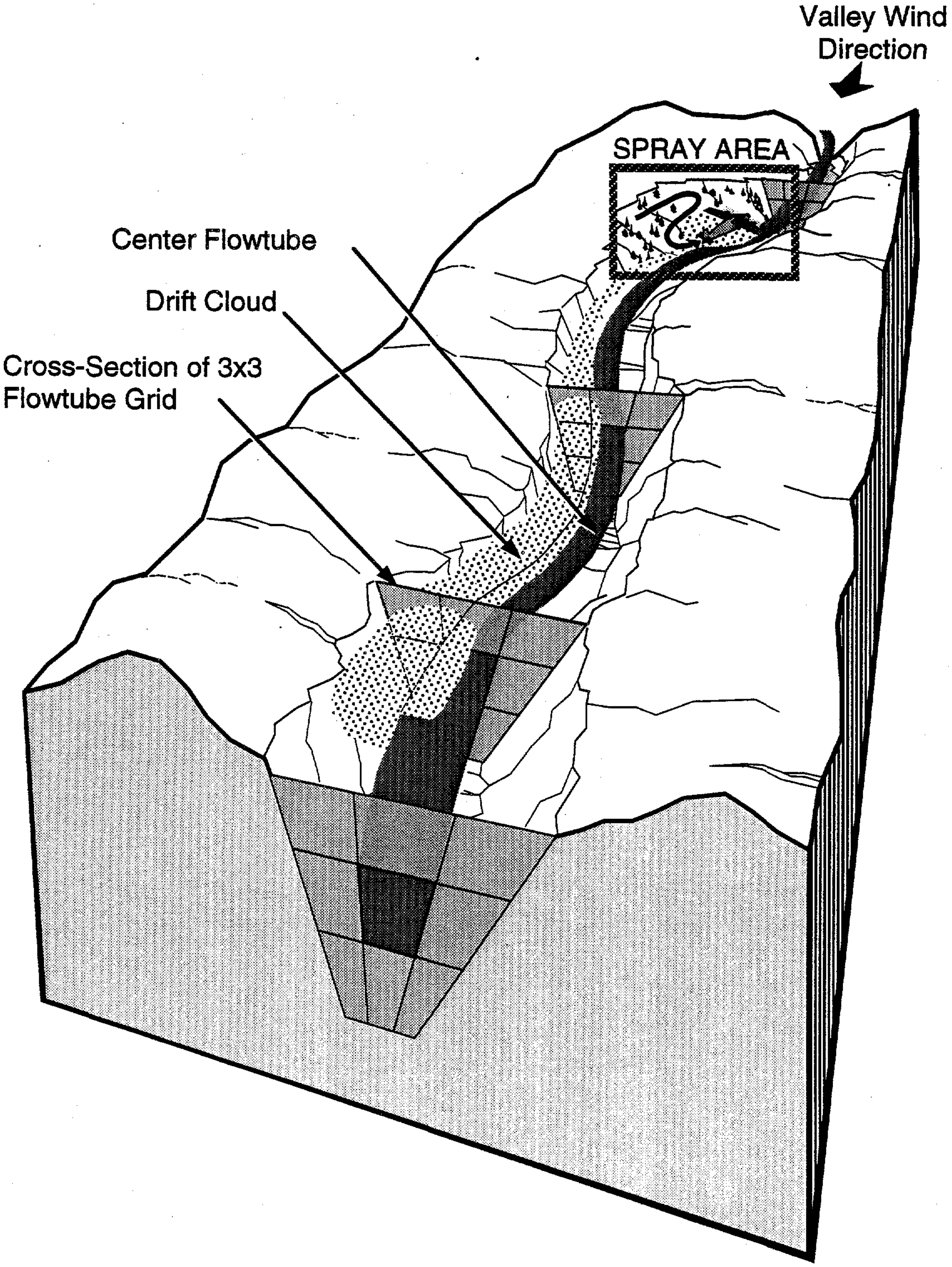


FIGURE 2.1. Example of the Flowtube Approach Used in VALDRIFT.

where t = time (s)

S = the along-valley coordinate (m)

C = the concentration along the flowtube (g/m^3)

A = the cross-sectional area of the flowtube (m^2), a function only of S

$\dot{V}$ = the along-valley air volume flow rate through the flowtube (m^3/s)

Γ_{cs} = the emission source/sink term for the flowtube (g/s-m)

Γ_{cp} = the deposition source/sink term for the flowtube (g/s-m)

Γ_{cc} = the chemical transformations source/sink term for the flowtube (g/s-m)

Γ_{ca} = the lateral and vertical advection source/sink term for the flowtube (g/s-m)

Γ_{cd} = the turbulent diffusion source/sink term for the flowtube (g/s-m)

The first term in Equation (2.1) is the rate of change of the storage of C in the differential control volume A•dS, the second term is the along-valley advection of C, and the remaining terms represent the various source/sink terms. Chemical transformations are currently not implemented in VALDRIFT, and the chemical transformations source/sink term is presently set to zero. The treatment of chemical processes in VALDRIFT will require the inclusion of a conservation equation for each species. The lateral and vertical advective effects source/sink term is also currently not implemented in this version of VALDRIFT, so the advective effects source/sink term is set to zero for all grid cells in the modeling domain. The other three source/sink terms are discussed next, followed in Section 2.2.4 by a description of the numerical method used to solve Equation (2.1).

2.2.1 Emission Source/Sink Term - Γ_{cs}

The emission source/sink term is simply the rate of pesticide mass released to each grid cell, per unit length of the grid cell, during a model time step. Most grid cells in the modeling domain will probably have Γ_{cs} equal to zero. VALDRIFT maps releases from all point and line (flight paths) sources specified by the user to the appropriate grid cells.

2.2.2 Deposition Source/Sink Term - Γ_{cp}

The deposition source/sink term in VALDRIFT currently treats dry deposition. The dry deposition parameterization is given by the equation

$$\Gamma_{cp} = V_d C \Delta A \tag{2.2}$$

where $$V_d = \frac{u_{*d}^2}{\overline{U}_d} \tag{2.3}$$

V_d = the dry deposition velocity (m/s)

C = the concentration in a grid cell adjacent to the ground (g/m^3)

ΔA = the ground surface area of the grid cell divided by ΔS (m)

ΔS = the along-valley length of the grid cell (m)

$\overline{U}_d$ = the characteristic along-valley wind speed used for deposition (m/s)

u_{*d} = the surface friction velocity used for deposition (m/s)

The friction velocity is a function of atmospheric stability and is determined by

$$u_{*ds} = \frac{\kappa \, \overline{U}_{night}}{a_s} \leq 0.3 \tag{2.4}$$

$$u_{*dn} = \frac{\kappa \left(\overline{U}_{night} + \overline{U}_{day}\right)/2}{a_n} \leq 0.6 \tag{2.5}$$

$$u_{*du} = \frac{\kappa \, \overline{U}_{day}}{a_u} \leq 0.95 \tag{2.6}$$

where u_{*ds} = the friction velocity for stable conditions (m/s)

u_{*dn} = the friction velocity for neutral conditions (m/s)

u_{*du} = the friction velocity for unstable conditions (m/s)

κ = von Karmen's constant (0.4)

a_n = 6.215 = ln (500)

a_s = $a_n + 7.5$

$a_u = a_n - 1.5$

$\overline{U}_{night}$ = the characteristic nighttime along-valley wind speed (m/s)

$\overline{U}_{day}$ = the characteristic daytime along-valley wind speed (m/s)

The nighttime and daytime characteristic along-long valley wind speeds are specified by the VALDRIFT user, and are winds that are typically experienced in the particular valley being simulated. The friction velocity used in Equation (2.3) for calculating the deposition velocity is the minimum of the friction velocities from Equations (2.4), (2.5) and (2.6), and the characteristic along-valley wind speed used in Equation (2.3) is the maximum of three wind speeds - $\overline{U}_{night}$, $\overline{U}_{day}$ or 1 m/s.

The amount of pesticide deposited to the ground at a grid cell adjacent to the ground is determined as

$$\omega_d = V_d C \Delta t \qquad (2.7)$$

where ω_d = the mass deposited to the ground during a model time step (g/m^2)

Δt = the model time step (s)

2.2.3 Turbulent Diffusion Source/Sink Term - Γ_{cd}

Material can move among flowtubes by turbulent diffusion. The K-theory approach is used where horizontal (lateral) and vertical diffusivities can be specified for all flowtubes. Assuming the lateral and vertical diffusivities to be a function only of the vertical coordinate Z, the source/sink term representing diffusion in Equation (2.1) is given as

$$\Gamma_{cd} = \Gamma_{cdY} + \Gamma_{cdZ} \qquad (2.8)$$

where the lateral and vertical turbulent diffusion terms are represented by

$$\Gamma_{cdY} = \overline{\dot{V}_Y' C'}\Big|_{Face\,j} - \overline{\dot{V}_Y' C'}\Big|_{Face\,j-1} \qquad (2.9)$$

$$\Gamma_{cdZ} = \overline{\dot{V}_Z' C'}\Big|_{Face\,i} - \overline{\dot{V}_Z' C'}\Big|_{Face\,i-1} \qquad (2.10)$$

and the overbar terms represent the time-averaged turbulent fluxes of C in the Y- and Z-directions as denoted, where Y is the cross-valley coordinate. The turbulent volume flow terms ($\dot{V}_Y'$ and $\dot{V}_Z'$) are flow in the Y- and Z-directions per unit length in the S-direction [m^3/s-m]. The i index refers to the vertical direction and the j index refers to the lateral direction. The index notation for a subset of flowtubes on a valley cross-section is shown in Figure 2.2. The lateral and vertical diffusion source/sink terms evaluated using Equations (2.9) and (2.10), respectively, represent the i,j computational grid cell. Equations (2.9) and (2.10) are approximated using K-theory as

$$\Gamma_{cdY} = \Delta Z_i \, K_{Y,i} \left.\frac{\partial C}{\partial Y}\right|_{\text{Face j}} - \Delta Z_i \, K_{Y,i} \left.\frac{\partial C}{\partial Y}\right|_{\text{Face j-1}} \qquad (2.11)$$

$$\Gamma_{cdZ} = \Delta Y_i \, K_{Z,i} \left.\frac{\partial C}{\partial Z}\right|_{\text{Face i}} - \Delta Y_{i-1} \, K_{Z,i-1} \left.\frac{\partial C}{\partial Z}\right|_{\text{Face i-1}} \qquad (2.12)$$

where K_Y is the lateral turbulent eddy diffusivity and K_Z is the vertical turbulent eddy diffusivity, which are both functions of height. The lengths ΔY and ΔZ are functions only of Z on each cross section, and can be easily determined. The derivatives in Equations (2.11) and (2.12) are evaluated in VALDRIFT using central finite differencing, as defined in the following equations

$$\Gamma_{cdY} = \frac{\Delta Z_i \, K_{Y,i}}{\overline{\Delta Y_i}} \left(C_{i,j-1} - 2C_{i,j} + C_{i,j+1}\right) \qquad (2.13)$$

$$\Gamma_{cdZ} = \frac{\Delta Y_i \, K_{Z,i}}{\overline{\Delta Z_{i+1}}} \left(C_{i+1,j} - C_{i,j}\right) - \frac{\Delta Y_{i-1} \, K_{Z,i-1}}{\overline{\Delta Z_i}} \left(C_{i,j} - C_{i-1,j}\right) \qquad (2.14)$$

$$\overline{\Delta Z_i} = \frac{1}{2}\left(\Delta Z_i + \Delta Z_{i-1}\right) \qquad (2.15)$$

$$\overline{\Delta Y_i} = \frac{1}{2}\left(\Delta Y_i + \Delta Y_{i-1}\right) \qquad (2.16)$$

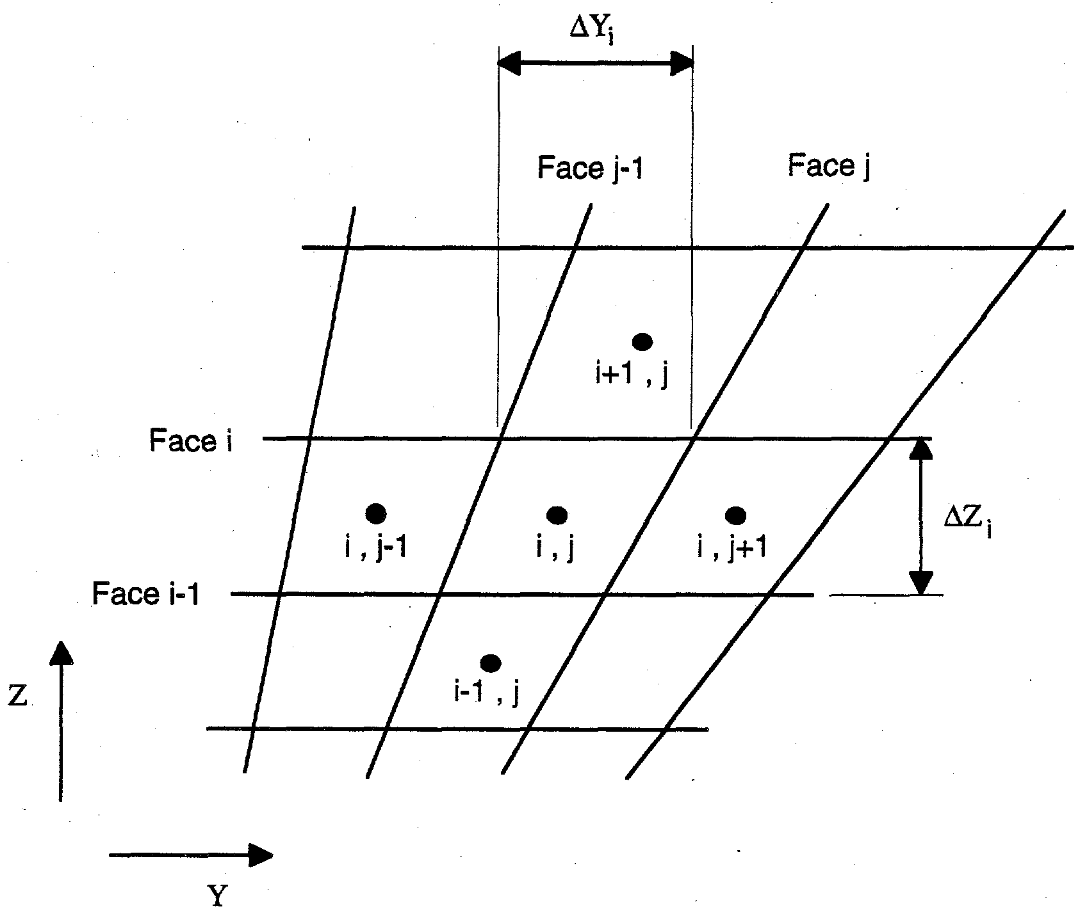


FIGURE 2.2. Index Notation Associated with the Flowtubes on a Valley Cross Section

The turbulent eddy diffusivities in Equations (2.13) and (2.14) are functions of atmospheric stability. VALDRIFT operates in three stability regimes - stable, neutral and unstable, where a stable atmosphere is assumed throughout the valley atmosphere during the nighttime and within the "stable core" during the morning transition period (refer to Appendix A for description of the stable core). The valley atmosphere is assumed to be neutral in the region above the stable core during the morning transition period. After the nocturnal temperature inversion has been destroyed and the stable core is no longer present, the valley atmosphere is considered to be unstable until sunset. After sunset, the valley atmosphere is assumed to be stable again. Within the growing convective boundary layer below the stable core during the morning transition period, the valley atmosphere is considered unstable.

The vertical and lateral turbulent eddy diffusivities are evaluated using turbulence theory formulations described by Businger et al. (1971), Panofsky and Dutton (1984) and Bian et al. (1992) as follows

$$K_Z = \frac{\kappa u_* Z}{\phi_m}\left(1 - \frac{Z}{H}\right) \tag{2.17}$$

$$K_Y = \left(\frac{\sigma_v}{\sigma_w}\right)^2 K_Z \tag{2.18}$$

$$\sigma_v = a u_* \tag{2.19}$$

$$\sigma_w = b u_* \tag{2.20}$$

$$\phi_m = 1 + 5\frac{Z}{L} \; ; \qquad Z < h \text{ , during stable conditions} \tag{2.21}$$

$$\phi_m = \left(1 - 16\frac{Z}{L}\right)^{0.25} \; ; \quad Z < h \text{ , during unstable conditions} \tag{2.22}$$

where ϕ_m = the turbulent flux-profile relationship function
H = the height of the planetary boundary layer (m)
σ_V = the standard deviation of the lateral turbulent velocity (m/s)
σ_W = the standard deviation of the vertical turbulent velocity (m/s)
h = the height of the surface layer (m)
L = the Monin-Obukov turbulent length scale (m)

and a and b are constants. The values of the turbulence parameters needed to evaluate the above equations for the turbulent eddy diffusivities are given in Table 2.2. For neutral conditions, the flux-profile relationship function has a value of one. Diffusion of the pesticide in the along-valley direction is assumed to be negligible relative to the along-valley advection, and consequently, the S-direction diffusion source/sink term is set equal to zero in VALDRIFT.

TABLE 2.2. Values of Turbulence Parameters Used in VALDRIFT as a Function of Atmospheric Stability

PARAMETER	STABLE	NEUTRAL	UNSTABLE
H (m)	400	600	1400
h (m)	10	20	30
L (m)	30	∞	-30
u_* (m/s)	0.1	0.2	0.4
a	2.29	2.00	2.28
b	0.95	1.20	1.98

2.2.4 Numerical Method

The conservation of species Equation (2.1) for each flowtube is integrated using a fully explicit finite difference scheme consisting of forward Euler differencing in time, upwind differencing for advection, and central differencing for diffusion (as discussed in Section 2.2.3). Substituting the finite differences approximations into Equation (2.1) and rearranging gives, for down-valley flows, the concentration, C_k^n, in flowtube grid cell "k" at time step "n" as

$$C_k^n = C_k^{n-1} - \frac{\Delta t}{A_k \Delta S}\left(\dot{V}_k^{n-1} C_k^{n-1} - \dot{V}_{k-1}^{n-1} C_{k-1}^{n-1}\right) + \frac{\Delta t}{A_k}\left(\Gamma_{cs,k}^{n-1} + \Gamma_{cp,k}^{n-1} + \Gamma_{cd,k}^{n-1}\right) \qquad (2.23)$$

where C_k^{n-1} = the concentration in grid cell k at the previous time step n-1 (g/m^3)

C_{k-1}^{n-1} = the concentration in grid cell k-1 at time step n-1 (g/m^3)

$\dot{V}_k^{n-1}$ = the air volume flow rate in grid cell k at time step n-1 (m^3/s)

$\dot{V}_{k-1}^{n-1}$ = the air volume flow rate in grid cell k-1 at time step n-1 (m^3/s)

$\Gamma_{cs,k}^{n-1}$ = the emission source/sink term in grid cell k at time step n-1 (g/s-m)

$\Gamma_{cp,k}^{n-1}$ = the deposition source/sink term in grid cell k at time step n-1 (g/s-m)

$\Gamma_{cd,k}^{n-1}$ = the diffusion source/sink term in grid cell k at time step n-1 (g/s-m)

A_k = the cross-sectional area of flowtube grid cell k (m^2)

n = 1, 2, ..., N_t

k = 1, 2, ..., N_S

N_t = the total number of times steps in the simulation

N_S = the total number of grid cells in the along-valley direction

For up-valley winds Equation (2.23) takes the following form because of the upwind differencing requirement on the along-valley advection term in Equation (2.1)

$$C_k^n = C_k^{n-1} - \frac{\Delta t}{A_k \Delta S}\left(\dot{V}_{k+1}^{n-1} C_{k+1}^{n-1} - \dot{V}_k^{n-1} C_k^{n-1}\right) + \frac{\Delta t}{A_k}\left(\Gamma_{cs,k}^{n-1} + \Gamma_{cp,k}^{n-1} + \Gamma_{cd,k}^{n-1}\right) \qquad (2.24)$$

Note that the concentration distribution along each flowtube described by Equations (2.23) and (2.24) is determined explicitly at the new time step, n, given the concentration distribution, the air volume flow rates and source/sink terms at the old time step, n-1. The initial conditions (concentrations at n = 0 for all k) and boundary conditions (concentrations at k = 0 for all n for down-valley winds, and concentrations at k = N_S for all n for up-valley winds) are required to solve Equations (2.23) and (2.24). The initial conditions in VALDRIFT are that the concentrations are constant for all S, where the constant value is specified by the user. The choice of boundary conditions in VALDRIFT is either of the Dirichlet type where C is specified (same value as the initial conditions), or of the von Neumann type where the gradient of C is specified. VALDRIFT only allows a zero gradient boundary condition (i.e., the inflow concentration is set equal to the concentration in the first down-wind grid cell) if the von Neumann type of boundary condition is selected.

Explicit numerical schemes are conditionally stable, that is, certain conditions apply to relationships between the grid spacing, the model time step, and model parameters such as wind speed. Numerical stability of Equations (2.23) and (2.24) is maintained by determining the modeling time step such that the following stability criteria is satisfied

$$0 < \Delta t\left(\frac{U_{Smax}}{\Delta S} + \frac{2K_{Ymax}}{\Delta Y_{min}^2} + \frac{2K_{Zmax}}{\Delta Z_{min}^2}\right) \leq 1 \qquad (2.25)$$

where

ΔY_{min} = the smallest cross-valley grid spacing (m)

ΔZ_{min} = the smallest vertical grid spacing (m)

U_{Smax} = the max. along-valley wind speed expected during a simulation (m/s)

K_{Ymax} = the max. cross-valley diffusivity expected during a simulation (m^2/s)

K_{Zmax} = the max. vertical wind diffusivity expected during a simulation (m^2/s)

The stability criteria identified by Equation (2.25) is derived by substituting Equations (2.13) and (2.14) into Equation (2.23) and regrouping terms according to the dependent variable. The stability criteria results from the condition that the coefficient of C_k^{n-1} must always be positive. The stability criteria in Equation (2.25) is similar to the well-known Courant-Friedrichs-Lewy (CFL) stability condition applying to explicit difference schemes for hyperbolic partial differential equations. The stability criteria identified by Equation (2.25) will be denoted as the Courant stability criteria, even though Equation (2.1) is actually a second-order parabolic partial differential equation (because the diffusion source/sink term is second-order and is a function of the dependent variable). Because the two diffusion terms in Equation (2.25) are typically much smaller than the advection term, Equation (2.25) is simplified as

$$0 < \Delta t \frac{U_{Smax}}{\Delta S} \leq r \qquad (2.26)$$

where r is the Courant number which is assumed to be 0.6 in VALDRIFT as a conservative number to ensure numerical stability. Solving Equation (2.26) for Δt gives

$$\Delta t = 0.6 \frac{\Delta S}{U_{Smax}} \qquad (2.27)$$

where U_{Smax} is always assumed to be greater than 1 m/s.

2.3 COMPUTATIONAL GRID

The computational domain should extend far enough down-valley and up-valley from the source to handle transport distances commensurate with the simulation period. This could be up to 100 - 200 km up-valley and down-valley (provided the valley is this long) for one diurnal cycle. Another consideration is that the domain extend enough beyond the receptor area of concern that edge effects are minimized. That is, material advected off the grid is lost and, consequently, cannot contribute to concentrations inside the model domain in the event of flow reversal. The cross-valley extent of the domain should be from ridgetop to ridgetop.

The VALDRIFT coordinate system (S,Y,Z) is nonorthogonal, with the S-axis following the valley floor centerline in the down-valley direction. The origin is at the up-valley edge of the domain at the center of the valley cross section on the valley floor. The positive S-axis points down the valley axis at the valley center; the positive Y-axis extends horizontally to the right (looking up-valley) locally perpendicular to the S-axis; and the positive Z-axis is oriented vertically. The rigorous transformation of the conservation of species equation from a Cartesian (rectangular) coordinate system into the "valley-following" coordinate system used by VALDRIFT would theoretically introduce "stretching factors" into the transformed equation. These stretching factors, however, are not included in Equation (2.1) in order to maintain computational efficiency. The errors introduced into the solution are small for valleys that do not change course by more than approximately 45° within a few kilometers.

Once the computational domain has been determined, certain terrain characteristics need to be specified. These can be determined from contour maps and/or computer-based digital elevation data sets. The terrain characteristics required at various valley cross sections are

S_{XC} = the down-valley distance of the cross-section (m),

α_L = left sidewall angle (degrees),

α_R = right sidewall angle (degrees),

z_f = elevation of the valley floor (m MSL),

z_r = elevation of the ridgetops (if the ridgetop elevations differ for the two sidewalls, use the lower of the two elevations) (m MSL).

ℓ = width of the valley floor (m),

Each valley cross section is represented as shown in Figure 2.3, and the above quantities are identified in the figure.

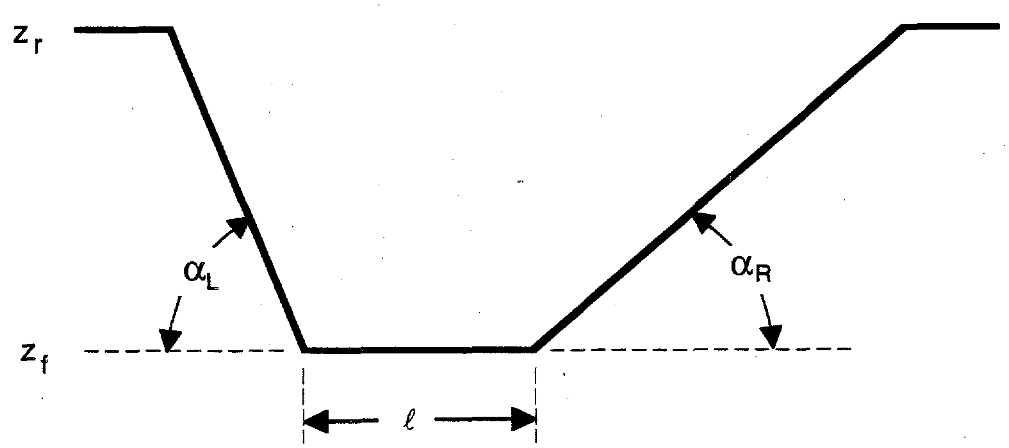


FIGURE 2.3. Simplified Valley Cross Section Showing Required Parameters

The computational grid is generated by specifying the number of flowtubes in the cross-valley and vertical directions, for example 3 x 3 as shown in Figure 2.1. A starting cross-section is divided into the desired number of layers in the vertical direction, and each layer is divided into the desired number of flowtubes in the cross-valley direction. The entire grid is generated using the condition that the flowtubes are always horizontal in the cross-valley direction, are conformal to the valley sidewalls in the vertical direction, and that the ratio of the cross-sectional area of each flowtube to the total cross-sectional area of the valley does not vary in the along-valley direction.

The number of flowtube layers in the vertical is identified by N_Z, and the number of flowtube columns across the valley by N_Y. The computational grid is then generated as follows. First, a valley cross section is initialized from which the complete grid is generated. This cross section should be "typical" and roughly the median in cross-sectional area. The i-index denotes the layers, and the j-index the columns. The layers are evenly spaced in Z, and the columns are evenly spaced in Y at each Z. The cross-sectional area, A_{ij}, of each ij flowtube at section S_0 is

$$A_{ij}(S_0) = \frac{A_i(S_0)}{N_Y} \; ; \quad j = 1, 2, ..., N_Y \tag{2.28}$$

where

$$A_i(S_0) = \int_{Z_{i-1}(S_0)}^{Z_i(S_0)} \left[Y_R(S_0, Z) - Y_L(S_0, Z) \right] dZ \tag{2.29}$$

$$Z_i(S_0) = Z_{i-1}(S_0) + \Delta Z_i(S_0) \; ; \quad i = 1, 2, ..., N_Z \; ; \quad Z_0 = 0 \tag{2.30}$$

$$\Delta Z_i(S_0) = \frac{z_r(S_0) - z_f(S_0)}{N_Z} \tag{2.31}$$

The computational flowtube grid is then generated at all S, using the following two relationships

$$A_i(S) = \frac{A_i(S_0)}{A(S_0)} A(S) \tag{2.32}$$

$$A_{ij}(S) = \frac{A_i(S)}{N_Y} \; ; \quad j = 1, 2, ..., N_Y \tag{2.33}$$

where the total cross-sectional area of the valley, A, at any location, S, is determined by

$$A(S) = \sum_{i=1}^{N_Z} A_i(S) \tag{2.34}$$

or

$$A(S) = \int_0^{Z_{N_Z}(S)} \left[Y_R(S, Z) - Y_L(S, Z) \right] dZ \tag{2.35}$$

The vertical spacing of the flowtubes on any cross section is not generally the same as at S_0 [Equation (2.31) does not necessarily hold at any arbitrary cross section, S]. Rather, the layer heights at S are determined by finding the Z_i's where

$$\int_{Z_{i-1}(S)}^{Z_i(S)} [Y_R(S,Z) - Y_L(S,Z)]\, dZ = A_i(S) \tag{2.36}$$

Equations (2.32), (2.33) and (2.36) are valid under the conditions that 1) relative fractions of the valley-cross-sectional area associated with each flowtube remains constant for all S, and 2) that the layers are always horizontal in the valley cross section. These conditions impose a flowtube configuration that is consistent with the general flow characteristics.

A_i in Equation (2.36) can be determined directly from analytical expressions for the simplified valley cross section shown in Figure 2.3. The integral on the left-hand-side of Equation (2.36) is evaluated as

$$\int_{Z_1(S)}^{Z_2(S)} [Y_R(S,Z) - Y_L(S,Z)]\, dZ = \int_{Z_1(S)}^{Z_2(S)} \left[\left(\frac{\ell}{2} + \frac{Z}{\tan \alpha_R}\right) - \left(-\frac{\ell}{2} - \frac{Z}{\tan \alpha_L}\right)\right] dZ \tag{2.37}$$

$$= \ell(Z_2 - Z_1) + \frac{\Theta}{2}\left(Z_2^2 - Z_1^2\right) \tag{2.38}$$

where
$$\Theta = \frac{1}{\tan \alpha_L} + \frac{1}{\tan \alpha_R} \tag{2.39}$$

2.4 METEOROLOGICAL MODEL

2.4.1 Wind Fields

Air mass is conserved using the continuity equation considering only the along-valley flow explicitly. Nonrecirculating lateral flows are treated as source/sink terms in the equation. Possible sources and sinks of air mass are from subsidence, regional flow intrusions, and tributary flows; currently subsidence is the only air mass source/sink term treated in VALDRIFT. Assuming the flow to be incompressible, the air mass (volume) in the valley is conserved following the equation

$$\frac{\partial \dot{V}(S,t)}{\partial S} - \Gamma_a(S,t) = 0 \tag{2.40}$$

where $\dot{V}$ is the total along-valley volume flow rate (m^3/s), and Γ_a is the air mass source/sink term ($m^3/s\text{-}m$). The air mass is also conserved for each flowtube, ij, following the equation

$$\frac{\partial \dot{V}_{ij}(S,t)}{\partial S} - \Gamma_{a,ij}(S,t) = 0 \tag{2.41}$$

where $\dot{V}_{ij}$ is the along-valley volume flow rate in flowtube ij (m^3/s), and $\Gamma_{a,ij}$ is the air mass source/sink term in flowtube ij. Integrating Equation (2.41) between a starting cross section, S_0, and some arbitrary cross section, S, gives

$$\dot{V}_{ij}(S,t) = \dot{V}_{ij}(S_0,t) - \int_{S_0}^{S} \Gamma_{a,ij}(S',t)\,dS' \tag{2.42}$$

Given the volume flow rate at S_0 and the source/sink terms for all S, the volume flow rate for each flowtube at time t can be determined for all S from Equation (2.42). The along-valley wind speed, u_{ij}, averaged over the area A_{ij} can then be calculated as

$$u_{ij}(S,t) = \frac{\dot{V}_{ij}(S,t)}{A_{ij}(S)} \tag{2.43}$$

Next consider the coordinate system (S,β,γ) which transforms from the (S,Y,Z) coordinate system using the equations

$$\beta = \frac{Y - Y_L}{Y_R - Y_L} - \frac{1}{2} \; ; \qquad -\frac{1}{2} \le \beta \le \frac{1}{2} \tag{2.44}$$

$$\gamma = \frac{z - z_f}{z_r - z_f} \; ; \qquad 0 \le \gamma \le 1 \tag{2.45}$$

Given the along-valley winds $u(S_0,\beta,\gamma,t)$ at one valley cross section (S_0), the volume flow rate for each flowtube at S_0 is determined as

$$\dot{V}_{ij}(S_0,t) = \{z_r(S_0) - z_f(S_0)\} \int_{\gamma_{i-1}}^{\gamma_i} \left[\{Y_R(S_0,\gamma) - Y_L(S_0,\gamma)\} \int_{\beta_{j-1}}^{\beta_j} u(S_0,\beta,\gamma,t)\, d\beta \right] d\gamma \quad (2.46)$$

where
$$\beta_j = \frac{j}{N_Y} - \frac{1}{2} \; ; \qquad j = 1, 2, ..., N_Y \; ; \qquad \beta_0 = -\frac{1}{2} \quad (2.47)$$

$$\gamma_i(S_0) = \frac{i}{N_Z} \; ; \qquad i = 1, 2, ..., N_Z \; ; \qquad \gamma_0 = 0 \quad (2.48)$$

If the along-valley winds are known at more than one location (e.g., two locations, S_0 and S_1), the flow field is made to conform to the measurements by incorporating a source/sink term in Equation (2.41) calculated as

$$\Gamma_{a,ij}(S : \{S_0 \rightarrow S_1\}, t) = \frac{\dot{V}_{ij}(S_0,t) - \dot{V}_{ij}(S_1,t)}{S_1 - S_0} \quad (2.49)$$

The along-valley winds are required over an entire valley cross section in order to calculate the volume flow rate for each flowtube using Equation (2.46). The approach used in VALDRIFT is to specifying the cross-valley and vertical structure of the along-valley winds using the semi-empirical parameterization given by Clements et al. (1989). They give the following semi-empirical relationship, called a "Prandtl parabolic wind profile", for mean nighttime drainage flow in a deep valley in western Colorado

$$u(Y,z,t) = U(t)\, f(Y)\, g(z) \quad (2.50)$$

where
$$f(Y) = 0.95 - 0.85\left(\frac{Y}{W(z)}\right)^2 \quad (2.51)$$

$$g(z) = 3.2\, e^{-\left[3.3\frac{z-z_f}{D-z_f}\right]} \sin\left(\pi \frac{z-z_f}{D-z_f}\right) \quad (2.52)$$

and U(t) is the maximum wind speed as a function of time observed in the valley atmosphere, W is the valley half-width (m), and D is the depth of the drainage flow (height at which wind speed goes to zero), which is typically considered to be the height of the valley sidewalls.

Transforming Equation (2.50), (2.51) and (2.52) into the (S,β,γ) coordinate system and assuming that the depth of the drainage flow extends to the ridgetop level (i.e., $D = z_r$) gives the along-valley winds at valley location S_0

$$u(S_0,\beta,\gamma,t) = U(S_0,t)\, f(\beta)\, g(\gamma) \tag{2.53}$$

where

$$f(\beta) = 0.95 - 3.4\beta^2 \tag{2.54}$$

$$g(\gamma) = 3.2\, e^{-3.3\gamma} \sin(\pi\gamma) \tag{2.55}$$

Substituting Equation (2.53) into Equation (2.46) , the along-valley volume flow rate is determined as

$$\dot{V}_{ij}(S_0,t) = U(S_0,t)\{z_r(S_0) - z_f(S_0)\} \int_{\beta_{j-1}}^{\beta_j} f(\beta)\, d\beta \;\; G(S_0,\gamma) \tag{2.56}$$

where

$$G(S_0,\gamma) = \int_{\gamma_{i-1}}^{\gamma_i} \left[\{Y_R(S_0,\gamma) - Y_L(S_0,\gamma)\}\, g(\gamma)\right] d\gamma \tag{2.57}$$

Evaluating the integral over β in Equation (2.56) gives

$$\dot{V}_{ij}(S_0,t) = U(S_0,t)\{z_r(S_0) - z_f(S_0)\}\left\{\frac{0.95}{N_\beta} - 1.13\left(\beta_j^3 - \beta_{j-1}^3\right)\right\} G(S_0,\gamma) \tag{2.58}$$

Using the relationships for the simplified valley cross section shown in Figure 2.3, and evaluating the integral over γ in Equation (2.57) gives the expression for G as

$$G(S_0,\gamma) = -\frac{3.2\, e^{-3.3\gamma}}{10.9+\pi^2}\, \Big\{ [\gamma\Theta(z_r - z_f) + \ell][3.3\sin(\gamma\pi) + \pi\cos(\gamma\pi)] + \frac{\Theta(z_r - z_f)}{10.9+\pi^2}\,[(10.9-\pi^2)\sin(\gamma\pi) + 6.6\pi\cos(\gamma\pi)] \Big\}\Big|_{\gamma_{i-1}}^{\gamma_i} \tag{2.59}$$

At each model time step, Equations (2.58) and (2.59) are used to generate the along-valley air volume flow rate for each flowtube at location S_0, given the maximum wind speed at each time step. Equation (2.42) is then used to determine the air volume flow rate at all S locations.

2.4.2 Morning Transition Period

The morning transition period is the post-sunrise period when the nocturnal down-valley flow is reversed to up-valley flow and when the nocturnal temperature inversion in the valley atmosphere is destroyed by the growth of a convective boundary layer (CBL) and the descent of the stable core remnant of the temperature inversion. Whiteman's (1990) report in Appendix A gives a more detailed description of the morning transition period. The approach for treating the morning transition period in VALDRIFT is based on the CBL growth/inversion descent thermodynamic model developed by Whiteman and McKee (1982) and implemented by Whiteman and Allwine (1985) in their valley dispersion model VALMET.

Whiteman and McKee's model is a "bulk" thermodynamic model in that it does not differentiate between sensible heat flux over different valley surfaces. The heat flux that drives the valley-inversion destruction is partitioned or distributed in a fundamentally different way from that for an inversion over homogeneous terrain. There, the sensible heat flux destroys the inversion by driving an upward growth from the ground of a convective boundary layer, which warms the inversion air mass from below until the temperature deficit is overcome and the inversion is destroyed. In contrast, in a valley the upward heat flux over valley surfaces develops a convective boundary layer, but also, as a result of sensible heat flux convergence over the slopes, causes warmed air parcels to flow up the slopes in an up-slope flow that develops within the CBL. These up-slope flows remove mass from the base of the temperature inversion in the shallow slope flows, and through mass continuity, result in a general subsiding motion over the valley center. The atmospheric energy budget approach used by Whiteman and McKee is capable of partitioning energy between these two different processes to produce inversion destruction solely by CBL growth, solely by inversion descent (assuming a nongrowing CBL is present initially in a simulation), or by a combination of the two processes. The partitioning is controlled by a single parameter, f_C, defined as the fraction of sensible heat flux going to CBL growth. The remaining fraction, 1 - f_C, is assumed to be responsible for mass transport up the valley sidewalls, which results in inversion descent. A default value for f_C of 0.15 is suggested.

The thermodynamic model is composed of two coupled equations. The first equation is a prediction equation for CBL height where, in accordance with the bulk nature of the model, the CBL depth H_C is assumed not to differ over the valley floor and sidewalls. The first equation is

$$H_{C,n} = H_{C,n-1} + \Delta H_{C,n-1} \; ; \qquad N_{tsr} < n \le N_{tss} \tag{2.60}$$

where
$$\Delta H_{C,n-1} = \frac{\theta}{T}\frac{f_C}{\rho c_p}\frac{\ell + \Theta H_{C,n-1}}{\ell + \Theta (H_{C,n-1}/2)}\frac{A_0 A_1}{\gamma_\theta H_{C,n-1}}\sin[\frac{\pi}{\tau}\Delta t(n - N_{tsr})]\Delta t \tag{2.61}$$

and
$$\frac{\theta}{T} = \left(\frac{1000}{P}\right)^{.286}$$

P = the atmospheric pressure (mb)

ρ = the density of the air in the valley atmosphere (kg/m^3)

c_p = the specific heat of air at constant pressure (1005 m^2/K-s^2)

A_0 = the fraction of extraterrestrial solar flux to sensible heat flux; $0 < A_0 < 1$

A_1 = the extraterrestrial solar flux on horizontal surface at solar noon (W/m^2)

γ_θ = the valley potential temperature lapse rate at sunrise (K/m)

τ = the length of the daylight period (s)

N_{tsr} = the time step of sunrise

N_{tss} = the time step of sunset

ℓ = width of the valley floor (m),

Θ = $\cot \alpha_L + \cot \alpha_R$

Note that a non-zero initial CBL height is necessary to make Equation (2.61) tractable. In the model, this requirement is met by using an initial CBL height at sunrise of 25 m. The second equation, describing the inversion top height h_I is

$$h_{I,n} = h_{I,n-1} + \Delta h_{I,n-1} \; ; \qquad N_{tsr} < n \le N_{tss} \tag{2.62}$$

$$\Delta h_{I,n-1} = \frac{(\beta_w/2)(h_{I,N_{tsr}} - h_{I,n-1})[\ell + \Theta(h_{I,N_{tsr}} - h_{I,n-1})/2]}{h_{I,n-1}\gamma_\theta(\ell + \Theta h_{I,n-1}/2) - \Delta t(n - N_{tsr})(\beta_w/2)(\ell + \Theta h_{I,n-1})}\Delta t -$$

where (2.63)

$$\frac{\theta}{T}\frac{1}{\rho c_p}\frac{[(\ell + \Theta h_{I,n-1}) - f_c(\ell + \Theta H_{I,n-1})]A_0 A_1 \sin[\frac{\pi}{\tau}\Delta t(n - N_{tsr})]}{h_{I,n-1}\gamma_\theta(\ell + \Theta h_{I,n-1}/2) - \Delta t(n - N_{tsr})(\beta_w/2)(\ell + \Theta h_{I,n-1})}\Delta t$$

and β_W = the warm air advection rate above the valley (K/s)

The numerical simulation using the coupled Equations (2.60) and (2.62) proceeds with discrete time steps and is completed when the inversion is destroyed at the first time step at which the CBL height becomes greater than the inversion top height, such that

$$H_n \geq h_n \quad (2.64)$$

The terms in Equation (2.63) involving the warming rate for the air above the inversion allow the model to incorporate the retarding effect on temperature inversion breakup caused by warm air advection above the valley temperature inversion. Extra energy is required to destroy the valley temperature inversion if this warming occurs during the temperature inversion breakup period because the inversion cannot be broken until the entire valley atmosphere is warmed to the temperature of the air above the valley.

The time of sunrise, time of sunset, length of the daylight period, and the solar flux at solar noon used in Equations (2.61) and (2.63) are calculated using Whiteman and Allwine's (1986) solar model, given the latitude and longitude of the center of the modeling region, and the month, day and year of the simulation.

The interaction of the CBL growth/inversion descent and the VALDRIFT computational grid is shown in Figure 2.4. As the CBL grows above the center of a flowtube, that flowtube is then assumed to be fully contained within the CBL. The same approach applies to the inversion descent where the flowtube is fully contained in the flows above the inversion top once the inversion top descends below the center of the flowtube. Different turbulent diffusivities apply to diffusion dependent on the flowtubes' locations relative to the stable core, as discussed in Section 2.2.3. For example, the first two rows (i = 1 and 2) of flowtubes in Figure 2.4 would be assigned unstable diffusivities because they are contained in the CBL, and the diffusivities for the top row (i = 7) of flowtubes would be based on neutral stability.

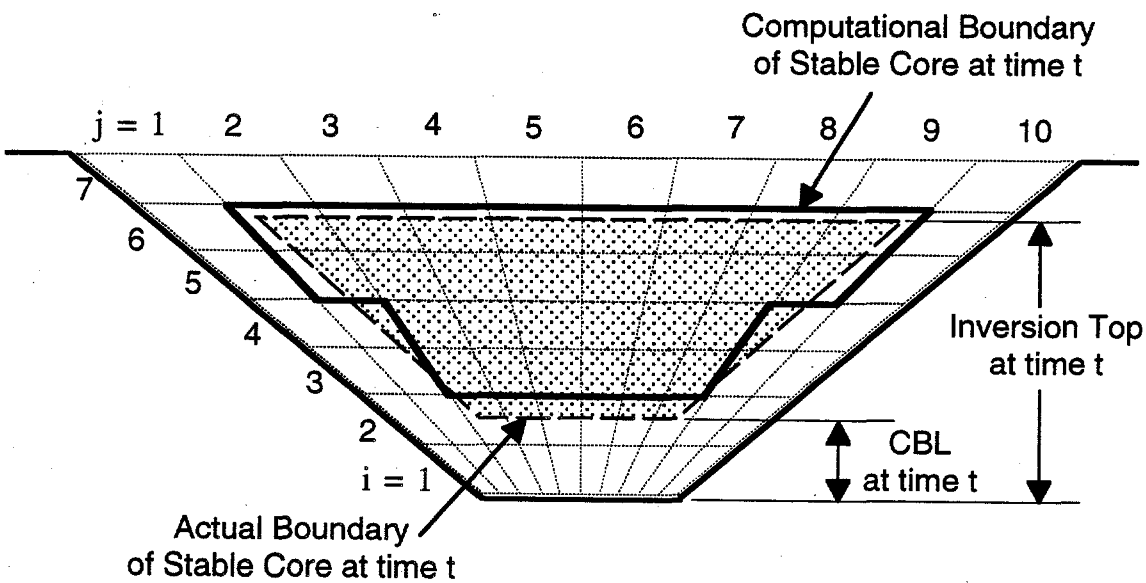


FIGURE 2.4. Illustration of the Computational Domain on a Valley Cross Section at Time t After Sunrise. The stable core, CBL, and inversion top are shown within a 7 by 10 array of valley flowtubes.

3.0 USER'S GUIDE

This section gives detailed instructions for configuring and operating VALDRIFT version 1.0. The operation of VALDRIFT is demonstrated using an example simulation where the input and output files are described. This section gives instructions for using VALDRIFT, with the instructions divided into seven major topics:

1. General Installation - the suggested directory structure and the installation procedure are described
2. Basic Operation - the basic procedures and files are described
3. Error Handling - VALDRIFT error messages are described
4. Setup Computational Domain - criteria and methods for defining the computational domain are described
5. Input Files - all input files are described
6. Output Files - all output files are described
7. Post-Processing - operation of the post-processing program is described

3.1 GENERAL INSTALLATION

VALDRIFT is written in standard Fortran and is transportable among various computers with Fortran compilers. A list of the files required by VALDRIFT and the suggested directory structure are given in Figure 3.1. This directory structure is recommended for VALDRIFT simply because the following discussion and examples are based on this structure. Other directory structures may be applied by changing the file path names in the appropriate VALDRIFT input files described in this section.

The computer memory requirements for VALDRIFT are roughly 4 Mb for a 200x31x31 computational grid. Disk space requirements for the input files are small, but the output files from VALDRIFT can be quite large. The output file characteristics are discussed in Section 3.6.

The VALDRIFT model is provided on a 3.5" DOS-formatted diskette. The procedure for installing VALDRIFT is quite simple. Insert the VALDRIFT diskette into a PC 3.5" floppy drive. Assuming that the floppy drive is labeled **A** and the internal hard disk is called **C**, enter **inst%A2C** at the **A:>** prompt followed by a return. All the files will be copied from **A** to **C** into the directory structure identified in Figure 3.1. After installation, the example case can be run by simply entering **VALDRIFT** at the **C:>** prompt, followed by a return. The provided input files will be read and the corresponding output files will be created.

C:\VALDRIFT		
• VALDRIFT.EXE	**E**	VALDRIFT executable file.
• VALDRIFT.FIL	**S**	Full path names for VALDRIFT input and output files.
• POSTPROC.EXE	**E**	POSTPROC executable file.
C:\VALDRIFT\INPUTS		
• BRUSHVAL.RS	**A**	Run specification file setting simulation conditions.
• BRUSHVAL.TER	**S**	Terrain data file giving valley cross-section information.
• BRUSHVAL.REL	**A**	Release data file giving release locations, times and rates.
• BRUSHVAL.WND	**A**	Wind data file giving wind speed and wind direction at a point in the valley.
C:\VALDRIFT\OUTPUTS		
• BRUSHVAL.TRC	**O**	Text file containing summary information from VALDRIFT.
• BRUSHVAL.BIN	**O**	Binary file containing conc. and deposition fields from VALDRIFT.
• VALDRIFT.ASC	**O**	Text file containing output from POSTPROC.
C:\VALDRIFT\SOURCE		
• (VALDRIFT Source)	**F**	FORTRAN code; 26 subroutines, 1 include file and a main program.
• (POSTPROC Source)	**F**	FORTRAN code; POSTPROC main program.

Note: **E** - Executable file; **S** - Static input file; **A** - Active input file; **O** - Output file; **F** - Fortran Source.

FIGURE 3.1. VALDRIFT Directory Structure and Files. Prefixes of the input and output file names are for the Brush Creek Valley Simulation

3.2 BASIC OPERATION

Applying VALDRIFT to estimate air concentrations and ground deposition patterns from a release in a valley is straightforward. The extent of the modeling domain, the characteristics of the valley, the characteristics of the release, and the meteorological observations must be provided. Instructions for acquiring the necessary information, and putting it into the format required by VALDRIFT are given in this section. Four input files provide the information to VALDRIFT. The files (using the prefix name BRUSHVAL for the sample simulation as an example) are

1. BRUSHVAL.RS - this file specifies the time period of the simulation, the size of the computational grid, and bulk measures of valley atmosphere characteristics.
2. BRUSHVAL.TER - this file gives the terrain data for the valley of interest, and thus specifies the geometry of the modeling domain.

3. BRUSHVAL.REL - this file specifies the location, time, duration and amount of the release.

4. BRUSHVAL.WND - this file gives the time series of the wind observations available in the valley.

A brief description of the steps required for successfully applying VALDRIFT to an atmospheric dispersion problem are given next, with more details given in subsequent sections. It is assumed that the model is installed on a computer with the appropriate directory structure in place. The steps are

1. The modeling domain is constructed from topographic maps of the spray area and a rough idea of the average wind speeds expected during a simulation period. The extent of drift of the pesticide is estimated from the winds, the time period of the release and the time period of the simulation. The length of the valley encompassing the up-valley and down-valley drift of the plume, and the valley width from ridgeline to ridgeline establishes the domain.

2. After the modeling domain has been defined, the valley characteristics can be determined from the topographic map. The valley is described by specifying the terrain characteristics, discussed in Section 2.4, in the TER input file.

3. The release information is specified by the coordinates of the starting and ending point of each flight path and the beginning and ending time for the particular flight path. The total mass of pesticide remaining in the air after each flight traverse is specified in the REL file along with the coordinates and time. The amount of mass remaining after a particular flight needs to be specified by a model such as FSCBG, where the pesticide taken up by the canopy is determined, and the amount remaining constitutes the source term used by VALDRIFT.

4. A file giving observations of wind speed and wind direction at a location in the valley is required. The wind data must be available for the duration of the simulation. The observations are assumed to be made at the center of the valley, at a down-valley coordinate and height specified in the WND file. If no wind observations are available, a default scenario should be used.

5. Once the TER, REL and WND text files are constructed using a text editor, VALDRIFT can be run using information in a "run specification" (RS) file, that is also created with a text editor.

6. VALDRIFT produces two output files. The TRC file is a text file that gives a run summary, listing all the input conditions and providing some general output results. The BIN file is a binary file written by VALDRIFT to document

all the concentration and deposition output results as a function of time. The three-dimensional concentration fields and ground-level deposition fields in the BIN file can be displayed in tabular form by using the post-processing program POSTPROC.

3.3 ERROR HANDLING

Some limited error checking is done by VALDRIFT. These error checks are limited to errors associated with the opening of files and to errors in the times of input data. Error messages are written to the TRC output file and, when an error occurs, the program terminates with an "abnormal termination" response to the screen. The error messages in the TRC file are typically one-line messages identifying the problem.

3.4 SETUP COMPUTATIONAL DOMAIN

Defining the computational domain is one of the most important steps in using VALDRIFT. Once the computational domain is configured for a particular valley, however, it does not need to be regenerated for subsequent applications for that particular valley.

The length of the domain is dictated primarily by the up-valley and down-valley transport distances expected during a simulation period. The maximum transport distance is estimated from wind observations in the valley. The duration of down-valley winds multiplied by the down-valley wind speed gives an estimate of the down-valley distance traveled, and the duration of up-valley winds multiplied by the up-valley wind speed gives an estimate of the up-valley distance traveled. Typically the winds during clear weather conditions will be down-valley throughout the night, slowing after sunrise and reversing to up-valley 2-4 hours after sunrise. The winds continue up-valley through the daytime until 1-2 h after sunset, when they slow and reverse to down-valley, beginning another diurnal cycle. With spray operations generally occurring near sunrise, the period of down-valley winds in a VALDRIFT simulation would typically be 2-4 h, and the period of up-valley winds would typically be 6-10 h, assuming simulations are continued through the late afternoon.

The length of the Brush Creek Valley simulation domain is 45 km, as shown in Figure 3.2. This is based on the spray potentially traveling 20 km down-valley (say, in a 2 m/s wind that persists for 3 h), and then reversing and being carried 45 km up-valley (say, in a 3 m/s wind which persists for 4 h). By centering the 45-km-long domain over the 2-km-square spray target area we can maintain the drift cloud within the computational domain during the entire simulation. Note in Figure 3.2 that the domain for the Brush Creek Valley simulation follows the meandering of the valley axis, and thus the along-valley coordinate S meanders with the valley. The S-coordinate origin is always located at the up-valley boundary of the domain, and values of S (in meters) increase in the down-valley direction to the down-valley domain boundary.

The width and the depth of the computational domain are determined from the valley geometry. The width would generally be the distance across the top of the valley from sidewall-to-sidewall, and can vary with S as shown in Figure 3.2 for the Brush Creek Valley modeling domain. The depth of the modeling domain is the vertical distance from the valley floor to the top of the valley sidewalls, and can also vary with S. Idealized valley cross-sections at six locations in the Brush Creek Valley are shown in Figure 3.3. The two cross-sections at the up-valley and down-valley domain boundaries are required for any VALDRIFT simulation. The intermediate cross-sections need be specified only where distinct changes occur in the valley geometry. The valley floor width, sidewall angles, valley floor elevation, ridgetop elevation and down-valley location for each cross section are specified in the TER file.

At this time, a complex system of valleys cannot be treated in VALDRIFT. The user must choose one down-valley and one up-valley path from the system of merging tributaries.

3.5 INPUT FILES

The five input files required by VALDRIFT are text files named VALDRIFT.FIL, ---.RS, ---.TER, ---.REL, and ---.WND. The later four files are named BRUSHVAL.RS, BRUSHVAL.TER, BRUSHVAL.REL, and BRUSHVAL.WND for the Brush Creek Valley sample simulation, and these files must be located in subdirectory C:\VALDRIFT\INPUTS\. Also, VALDRIFT.FIL must reside in the same directory as the VALDRIFT program (directory C:\VALDRIFT\). The formats and data required for each of these input files is described next.

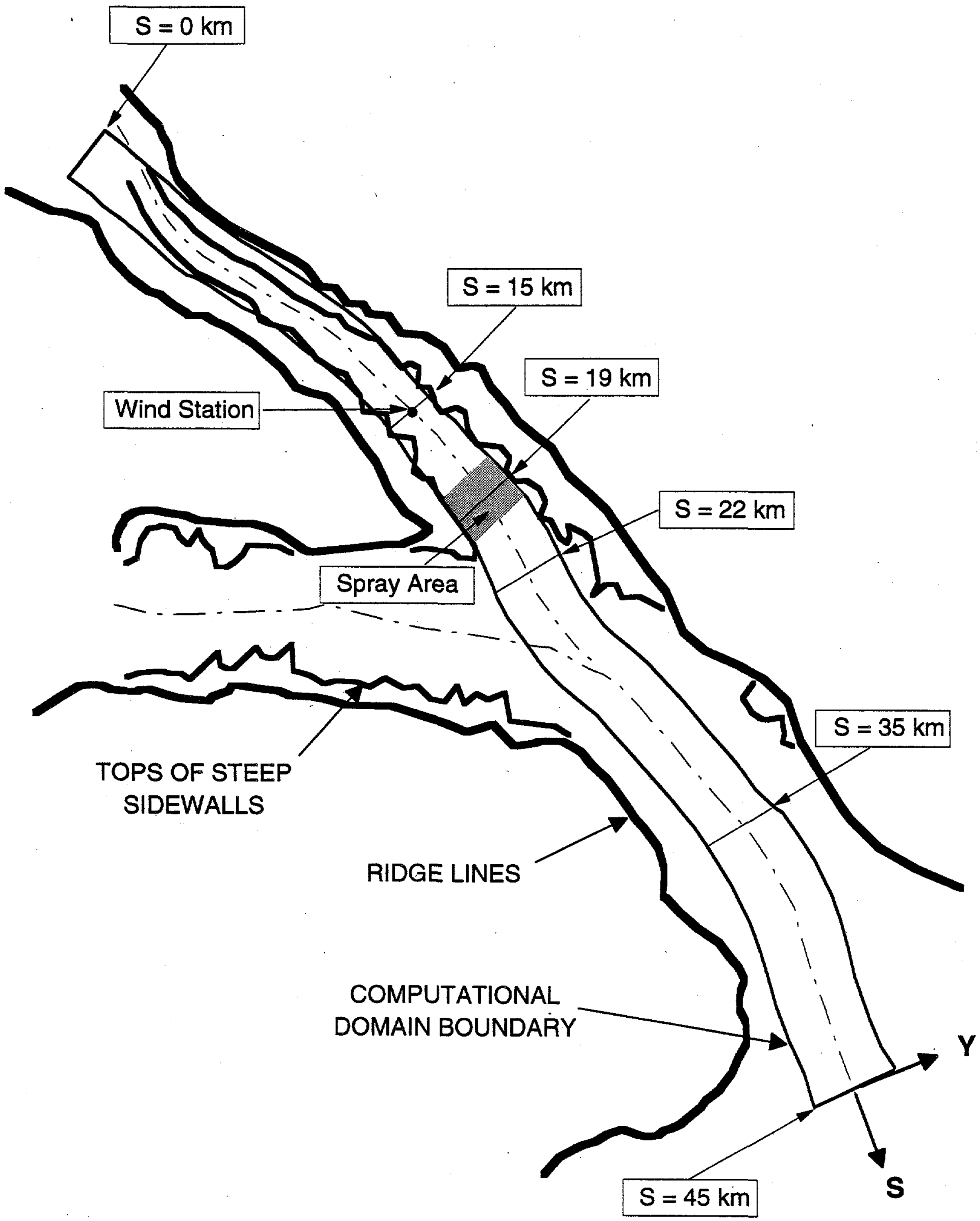


FIGURE 3.2 Computational Domain for the Brush Creek Valley Simulation

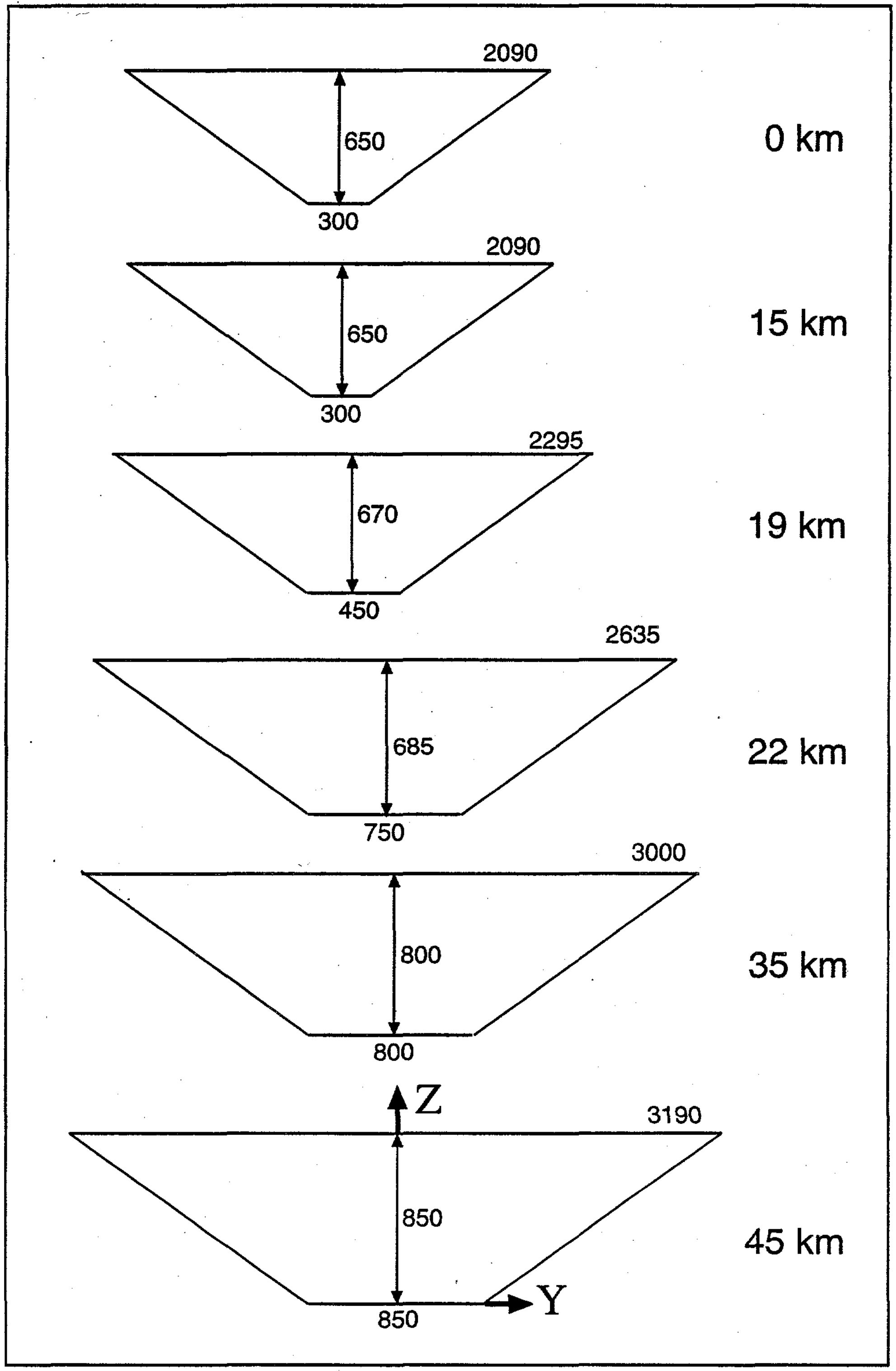


FIGURE 3.3. Valley Cross-Sections for the Brush Creek Valley Simulation

3.5.1 Pathname File - (VALDRIFT.FIL)

VALDRIFT.FIL is a text file containing the full path names of all the input and output files required by VALDRIFT. VALDRIFT.FIL is the first file read by VALDRIFT, and must reside in directory C:\VALDRIFT\ and must be named VALDRIFT.FIL. The file consists of six lines and the variable names used in VALDRIFT for each line are:

Line 1: `INFILES1`
Line 2: `INFILES2`
Line 3: `INFILES3`
Line 4: `INFILES4`
Line 5: `OUTFILE1`
Line 6: `OUTFILE2`

The definitions of the parameters in file VALDRIFT.FIL are:

`INFILES1` = Name of run specification file (---.RS)
`INFILES2` = Name of terrain data file (---.TER)
`INFILES3` = Name of release data file (---.REL)
`INFILES4` = Name of wind data file (---.WND)
`OUTFILE1` = Name of summary trace file (---.TRC)
`OUTFILE2` = Name of binary output file (---.BIN)

The VALDRIFT.FIL file for the Brush Creek Valley Simulation is given in Figure 3.4. The file names must be enclosed in single quotes and cannot be greater than 80 characters in length.

Line 1	'C:\VALDRIFT\INPUTS\BRUSHVAL.RS'
Line 2	'C:\VALDRIFT\INPUTS\BRUSHVAL.TER'
Line 3	'C:\VALDRIFT\INPUTS\BRUSHVAL.REL'
Line 4	'C:\VALDRIFT\INPUTS\BRUSHVAL.WND'
Line 5	'C:\VALDRIFT\OUTPUTS\BRUSHVAL.TRC'
Line 6	'C:\VALDRIFT\OUTPUTS\BRUSHVAL.BIN'

FIGURE 3.4. VALDRIFT.FIL for the Brush Creek Valley Simulation

3.5.2 Run Specification File - (---.RS)

The run specification file name will always have the file extension of .RS. The RS file consists of the following eight lines with a total of 28 parameters (the parameter names are used in the VALDRIFT code). Default values are suggested for several of the parameters where data may be lacking for the particular application. The parameter names used in VALDRIFT for each line are:

Line 1: `TITLE, USER, DETAIL, NEUMANN`
Line 2: `IYR, MO, IDA`
Line 3: `IHRBEG, MNBEG, IHREND, MNEND`
Line 4: `NPS, NPY, NPZ, NSTPS_MAX`
Line 5: `PRT_T, NS_PRT`
Line 6: `U_DAY, U_NIGHT, UMAX`
Line 7: `A0, PRESS, RHO, BETA_R, GAMMA, RKF`
Line 8: `DIF_COEF, C_BKGND`

The definitions of the parameters in the RS file are given next, where the bold-underlined values in brackets are recommended values.

Line 1:
- TITLE = Title of simulation (45 characters maximum)
- USER = Name of user (10 characters maximum)
- DETAIL = .TRUE. or .FALSE.; if .TRUE. then additional summary information is written to the TRC file **[.TRUE.]**
- NEUMANN = .TRUE. or .FALSE.; If .TRUE., the pesticide concentration in the air coming into the domain is set equal to the pesticide concentration in the air in the inflow boundary grid cells. When NEUMANN is set .TRUE. sources cannot be located on the inflow boundaries. If .FALSE., the concentration at the inflow boundary is set to C_BKGND **[.FALSE.]**

Line 2:
- IYR = Year of simulation (45 to 99)
- MO = Month of simulation (1 to 12)
- IDA = Day of simulation (1 to 31)

Line 3:
- IHRBEG = Beginning hour of simulation (00-23)
- MNBEG = Beginning minute of simulation (00-59)
- IHREND = Ending hour of simulation (00-23)
- MNEND = Ending minute of simulation (00-59)

Line 4:
- NPS = Number of grid points in S-direction (200 maximum) **[100]**

NPY = Number of grid points in Y-direction (21 maximum) [7]
NPZ = Number of grid points in Z-direction (21 maximum) [7]
NSTP_MX = Maximum number of grid points in the S-directions times the number of time steps (250,000 maximum) [100000]

Line 5:
PRT_T = Time step for writing output (minutes) [15]
NS_PRT = Write output every NS_PRT in S-direction [5]

Line 6:
U_DAY = Characteristic daytime along-valley winds for deposition (m/s) [4.]
U_NIGHT = Characteristic nighttime along-valley winds for dep. (m/s) [4.]
UMAX = Max. S-direction wind speed used for Courant check (m/s) [6.]

Line 7:
A0 = Fraction of extraterrestrial solar radiation at solar noon (0-0.5) [.3]
PRESS = Average air pressure of valley atmosphere (mb) [1000]
RHO = Average air density of valley atmosphere (kg/m^3) [1]
BETA_R = Rate of temperature change at top of valley atmosphere (K/s) [0.]
GAMMA = Mean potential temp. gradient in valley atmosphere (K/m) [.025]
RKF = Fraction of sensible heat flux going to CBL growth (0-1) [.15]

Line 8:
DIF_COF = Diffusion coefficient at the top of the valley atmosphere (0-1) [0.]
C_BKGND = Background concentration (g/m^3) [1.E-24]

The run specification file for the Brush Creek Valley simulation is given in Figure 3.5. Note that the title and user name in Line 1 must be enclosed in single quotes. The spacing of the fields in each line of the RS file is not significant, other than the fields be separated by at least one space or a comma (multiple spaces and a comma are acceptable).

Line 1	'Brush Creek Valley', 'KJ Allwine', .TRUE., .FALSE.
Line 2	84, 09, 26
Line 3	05, 30, 12, 30
Line 4	100, 7, 7, 250000
Line 5	30., 5
Line 6	5., 5., 5.5
Line 7	0.3, 815., 1.25, 0., .035, 0.5
Line 8	0., 1.E-24

FIGURE 3.5. BRUSHVAL.RS for the Brush Creek Valley Simulation

3.5.3 Terrain Data File - (---.TER)

The TER file is a text file containing the terrain information for the simulation, and the file name will always have the file extension of .TER. The TER file contains eight parameters, and the parameter names used in VALDRIFT for each line are

Line 1: RLAT, URLONG
Line 2: NPT
Line 3: S_TER
Line 4: Z_RIDGE
Line 5: Z_FLOOR
Line 6: ALPHA_L
Line 7: ALPHA_R
Line 8: WIDTH_FL

The definitions of the parameters in the TER file are

Line 1:
RLAT = North latitude of center of domain (degrees)
URLONG = West longitude of center of domain (degrees)

Line 2:
NPT = Number of terrain cross-sections

Line 3:
S_TER = Array of S-coordinates of locations of cross-sections (km)

Line 4:
Z_RIDGE = Array of ridge heights for cross-sections (m MSL)

Line 5:
Z_FLOOR = Array of floor elevations for cross-sections (m MSL)

Line 6:
ALPHA_L = Array of slope angle of left sidewalls for cross-sections (deg)

Line 7:
ALPHA_R = Array of slope angle of right sidewalls for cross-sections (deg)

Line 8:
WIDTH_FL = Array of valley floor widths for cross-sections (m)

Figure 2.4 gives a valley cross-section with the above requested quantities identified in the figure. Note that the left or right side of a cross-section is always identified relative to the observer looking up-valley. The TER file for the Brush Creek Valley simulation is given in Figure 3.6. The spacing of the fields in each line of the

TER file is not significant, other than the fields be separated by at least one space or a comma (multiple spaces and a comma are acceptable).

Line 1	39.5, 108.4
Line 2	6
Line 3	0., 15000., 19000., 22000., 35000., 45000.
Line 4	2550., 2550., 2510., 2480., 2450., 2400.
Line 5	1900., 1900., 1840., 1795., 1650., 1550.
Line 6	36., 36., 36., 36., 36., 36.
Line 7	36., 36., 36., 36., 36., 36.
Line 8	300., 300., 450., 750., 800., 850.

FIGURE 3.6. BRUSHVAL.TER for the Brush Creek Valley Simulation

3.5.4 Release Rate File - (---.REL)

The REL file is a text file containing the release location and release rate information for the simulation, and the file name will always have the file extension of .REL. The file consists of a variable number of lines depending on the number of times that the release rate changes and the number of release locations. The REL file contains 15 parameters, and the parameter names used in VALDRIFT for each line are

```
Line 1: NPOINTS
Line 2: S_SRC, Y_SRC, Z_SRC                       {1st Point Source}
Line 3: IHRB, MNB, IHRE, MNE
Line 4: SRC_MASS
Line 5: S_SRC, Y_SRC, Z_SRC                       {2nd Point Source}
Line 6: IHRB, MNB, IHRE, MNE
Line 7: SRC_MASS
.
.
Line  : S_SRC, Y_SRC, Z_SRC                       {"NPOINTS" Point Source}
Line  : IHRB, MNB, IHRE, MNE
Line  : SRC_MASS
Line  : NLINES
Line  : S_SB, Y_SB, Z_SB, S_SE, Y_SE, Z_SE        {1st Line Source}
Line  : IHRB, MNB, IHRE, MNE
Line  : SRC_MASS
```

```
Line   : S_SB, Y_SB, Z_SB, S_SE, Y_SE, Z_SE       {2nd Line Source}
Line   : IHRB, MNB, IHRE, MNE
Line   : SRC_MASS
.
.
Line   : S_SB, Y_SB, Z_SB, S_SE, Y_SE, Z_SE {"NLINES" Line Source}
Line   : IHRB, MNB, IHRE, MNE
Line   : SRC_MASS
```

The definitions of the parameters in the REL file are:

Line 1:
NPOINTS = Number of point sources. If no point sources are being specified, NPOINTS is set to 0 and the NLINES parameter is the next line in the file.

Line 2:
S_SRC = S-coordinate of point source (m)
Y_SRC = Y-coordinate of point source (m)
Z_SRC = Z-coordinate of point source (m above the valley floor)

Line 3:
IHRB = Hour that release began (00 to 23)
MNB = Minute that release began (00 to 59)
IHRE = Hour that release ended (00 to 23)
MNE = Minute that release ended (00 to 59)

Line 4:
SRC_MASS = Total mass released during release period (g)

Line :
NLINES = Number of line sources; or flight paths

Line :
S_SB = S-coordinate of one end of a line source (m)
Y_SB = Y-coordinate of one end of a line source (m)
Z_SB = Z-coordinate of one end of a line source (m above valley floor)
S_SE = S-coordinate of other end of a line source (m)
Y_SE = Y-coordinate of other end of a line source (m)
Z_SE = Z-coordinate of other end of a line source (m above valley floor)

The REL file for the Brush Creek Valley simulation is given in Figure 3.7. This example shows releases from six flight paths (line sources), where each 2-km-long line source is oriented in the along-valley direction. The first source is on the left sidewall (looking up-valley) and the release from this line source is for 10 minutes, beginning at 0530. The line source is 300 m off the valley-floor, roughly 50 m above the left sidewall. The remaining five line sources release progressively in 10-minute intervals, where the line sources are evenly spaced proceeding from left to right across the valley. The release rate from all sources is 1 g/s. Note that the Y-coordinate of a source is the perpendicular distance from the valley centerline. The coordinate values are negative proceeding "left" from the centerline, and the values are positive proceeding "right" from the valley centerline.

The spacing of the fields in each line of the REL file is not significant, other than the fields be separated by at least one space or a comma (multiple spaces and a comma are acceptable).

Line 1	0
Line 2	6
Line 3	18000., -750., 400., 20000., -750., 400.
Line 4	05, 30, 05, 40
Line 5	600.
Line 6	18000., -450., 300., 20000., -450., 300.
Line 7	05, 40, 05, 50
Line 8	600.
Line 9	18000., -150., 50., 20000., -150., 50.
Line 10	05, 50, 06, 00
Line 11	600.
Line 12	18000., 150., 50., 20000., 150., 50.
Line 13	06, 00, 06, 10
Line 14	600.
Line 15	18000., 450., 300., 20000., 450., 300.
Line 16	06, 10, 06, 20
Line 17	600.
Line 18	18000., 750., 400., 20000., 750., 400.
Line 19	06, 20, 06, 30
Line 20	600.

FIGURE 3.7. BRUSHVAL.REL for the Brush Creek Valley Simulation

3.5.5 Wind data file - (---.WND)

The wind inputs required are wind observations at one height from one station at the center of the valley. The wind observations must be evenly spaced in time. The

WND file is a text file containing the wind data in a VALDRIFT simulation. The wind data file name will always have the file extension of .WND. The structure of the WND file is described next using the thirteen parameters describing the wind information in VALDRIFT. The parameter names used in VALDRIFT for each line are

```
Line 1: S_MET, MET_NAM
Line 2: AZDWN, HTAGL, DTM, IYRM, MOM, IDAM, IHRM, MNM
Line 3: WD, WS, IWTIM
Line 4: WD, WS, IWTIM
.
.
Line  : WD, WS, IWTIM
```

The definitions of the parameters in the WND file are:

Line 1:
- S_MET = S-coordinate of location of wind station (m)
- MET_NAM = Name of wind station (8 characters maximum)

Line 2:
- AZDWN = Down-valley direction at the wind station (deg from True North)
- HTAGL = height of wind measurement (m AGL)
- DTM = Time interval between observations (minutes)
- IYRM = Year of wind data (45-99)
- MOM = Month of wind data (1-12)
- IDAM = Day of wind data (1-31)
- IHRM = Starting hour of wind data (00-23)
- MNM = Starting minute of wind data (00-59)

Line 3:
- WD = Wind direction (deg from True North)
- WS = Wind speed (m/s)
- IWTIM = Time of observation (expressed as HHMM)

The year, month and day in the wind file must exactly match the corresponding entries in the RS file. This year, month and day entry in the WND file is used by VALDRIFT as a check to ensure that the correct wind file is used for a particular simulation. The wind data starting time must not be later than the simulation starting time, and the period of the wind data must cover the entire simulation period. The WND file for the Brush Creek Valley simulation is given in Figure 3.8.

Line 1	15000., 'BRUSHVAL'
Line 2	120., 105., 15., 84, 09, 26, 05, 00
Line 3	5.0, 300., 0500
Line 4	5.0, 300., 0515
Line 5	5.0, 300., 0530
Line 6	5.0, 300., 0545
Line 7	5.0, 300., 0600
Line 8	5.0, 300., 0615
Line 9	5.0, 300., 0630
Line 10	5.0, 300., 0645
Line 11	5.0, 300., 0700
Line 12	4.3, 300., 0715
Line 13	3.5, 300., 0730
Line 14	2.7, 300., 0745
Line 15	2.0, 300., 0800
Line 16	1.0, 300., 0815
Line 17	0.0, 300., 0830
Line 18	1.0, 120., 0845
Line 19	2.0, 120., 0900
Line 20	2.6, 120., 0915
Line 21	3.2, 120., 0930
Line 22	3.8, 120., 0945
Line 23	4.5, 120., 1000
Line 24	5.0, 120., 1015
Line 25	5.0, 120., 1030
Line 26	5.0, 120., 1045
Line 27	5.0, 120., 1100
Line 28	5.0, 120., 1115
Line 29	5.0, 120., 1130
Line 30	5.0, 120., 1145
Line 31	5.0, 120., 1200
Line 32	5.0, 120., 1215
Line 33	5.0, 120., 1230
Line 34	5.0, 120., 1245
Line 35	5.0, 120., 1300

FIGURE 3.8. BRUSHVAL.WND for the Brush Creek Valley Simulation

3.6 OUTPUT FILES

The outputs from VALDRIFT are in the form of one formatted text file and one unformatted (binary) file written to disk. The technical descriptions of these files follows.

3.6.1 Run Summary Output File - (---.TRC)

This file is a diagnostic and summary file. It documents a particular run of VALDRIFT in addition to providing summary information about a total mass budget of all material released during the simulation. It summarizes the total mass remaining in the air, total mass deposited to the ground and the total mass advected off the grid. A portion of the TRC file for the Brush Creek Valley simulation is given in Figure 3.9. This is only a small portion of the file.

3.6.2 Concentration and Deposition Output File - (---.BIN)

This file is a sequential binary file containing a comprehensive collection of model outputs that are written to the file at each print time. This binary file, like all binary files, cannot be viewed using a text editor. Thus, a simple post-processing program, called POSTPROC, is provided to read the BIN file and allow the user to output the desired model run data in a form that can be viewed or printed with a text editor. POSTPROC is described in the next section.

The BIN file contains all the concentration and deposition results for the current simulation. The first three records of the BIN file list the characteristics of the run. Concentration and deposition results begin with record four and continue to the end of the file. The structure of the BIN file is outlined below using the parameter names from VALDRIFT.

```
Line 1: NPS, NPY, NPZ, Clk_b, Clk_e, PRT_t, dS
Line 2: dY_i, Z_i, dZ_i
Line 3: N_Gnd, i_Gnd, j_Gnd
Line 4: Clk, TOPCBL, TOPSBL, C_i_j, D_Gnd      {Data for First Time}
.
.
Line  : Clk, TOPCBL, TOPSBL, C_i_j, D_Gnd      {Data for Last Time}
```

```
**************************************************************************************************
*                                                                                                *
*                                        PROGRAM VALDRIFT                                        *
*                              ++++++VALLEY SPRAY DRIFT MODEL++++++                              *
*         PREDICTS AIR CONCENTRATIONS THROUGHOUT A VALLEY ATMOSPHERE AS A FUNCTION OF TIME.      *
*            DEPOSITION FIELDS ARE ALSO PREDICTED ON THE VALLEY FLOOR AND SIDEWALLS.            *
*            TREATS THE RELEASE OF AN INERT COMPOUND THAT CAN VARY IN TIME AND SPACE.           *
*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*
*         THE TITLE IS:Brush Creek Valley                          .       USER IS KJ Allwine    *
*            PROCESSING START TIME IS (MMDDYYHHMM) 0926840530         [ JULIAN DAY IS 270 ]      *
*            PROCESSING STOP  TIME IS (MMDDYYHHMM) 0926841230         [ JULIAN DAY IS 270 ]      *
**************************************************************************************************

 THE INPUT AND OUTPUT FILE NAMES ARE:

    C:\VALDRIFT\INPUTS\BRUSHVAL.RS
    C:\VALDRIFT\INPUTS\BRUSHVAL.TER
    C:\VALDRIFT\INPUTS\BRUSHVAL.REL
    C:\VALDRIFT\INPUTS\BRUSHVAL.WND
    C:\VALDRIFT\OUTPUTS\BRUSHVAL.TRC
    C:\VALDRIFT\OUTPUTS\BRUSHVAL.BIN

 THE RUN SPECIFICATIONS ARE:

    DETAIL = T     NEUMANN =  F   NPS = 100      NPY =   7      NPZ =   7      NSTPS_MAX =  250000
    CLK_B = 330.0  CLK_E = 750.0  PRT_T = 30.    NS_PRT =  5
    U_day_mean = 5.00              U_night_mean = 5.00
    UMAX =  5.5    VMAX =    .0   WMAX =    .0   A0 =  .30      PRESS =  815.
    RHO = 1.25     BETA_R =  .00  GAMMA = .035   RKF =  .50     NPM =  1       LUWB = 10

 THE MET STATIONS ARE:

    dTM =  15.     IYRM = 84      MOM =  9      IDAM = 26      IHRM =  0      MNM =  0
    CLK_BM =    0.

    Sta =  1       LUW = 10       S_MET =15000.  AZvly =120.   HTagl =  105.  MET_NAM = BRUSHVAL

 THE TERRAIN SPECIFICATIONS ARE:

    dS =  450.     RLAT =  39.50  LONG = 108.40  NPT =   6

    S_Ter =              0.     15000.     19000.     22000.     35000.     45000.
    z_r_Ter =         2550.      2550.      2510.      2480.      2450.      2400.
    z_f_Ter =         1900.      1900.      1840.      1795.      1650.      1550.
    alpha_l_Ter =       36.        36.        36.        36.        36.        36.
    alpha_r_Ter =       36.        36.        36.        36.        36.        36.
    l_width_Ter =      300.       300.       450.       750.       800.       850.

 THE SOLAR CHARACTERISTICS ARE:

    SR_Clk =  382.0   IHr = 06       Mn = 22.0
    SS_Clk = 1092.5   IHr = 18       Mn = 12.5
                      TAU = 710.5    A1 =  1030.5
```

FIGURE 3.9. Portion of BRUSHVAL.TRC for Brush Creek Valley Simulation

The definitions of the 18 parameters in the BIN file are

Line 1:

NPS	= Number of grid points in the S-direction
NPY	= Number of grid points in the Y-direction
NPZ	= Number of grid points in the Z-direction
Clk_b	= Starting time of simulation (minute of the day)
Clk_e	= Ending time of simulation (minute of the day)
PRT_t	= Time increment to write data (minutes)
dS	= S-direction grid spacing (m)

Line 2:

`dY_i`	=	Array of grid spacing in the cross-valley direction (m)
`Z_i`	=	Array of grid point heights (m above valley floor)
`dZ_i`	=	Array of grid spacing in the vertical direction (m)

Line 3:

`N_Gnd`	=	Number of grid points in the S-direction for deposition
`i_Gnd`	=	Number of grid points in the Z-direction on each sidewall
`j_Gnd`	=	Number of grid points in the Y-direction on the valley floor

Line 4:

`Clk`	=	Time into simulation of the data record (minute of the day)
`TOPCBL`	=	Height of the top of the CBL (m AGL)
`TOPSBL`	=	Height of the top of the stable core (m AGL)
`C_i_j`	=	3-D array of concentrations for time step (g/m^3)
`D_Gnd`	=	2-D array of ground deposition for time step (g/m^2)

3.7 POST-PROCESSING

A simple program called POSTPROC is provided with VALDRIFT to allow access to the BIN file and to document the method of reading data from the file. This program provides the user with a rudimentary means of viewing the results from VALDRIFT. We anticipate that users will write their own post-processing program to satisfy their particular needs. POSTPROC opens the BIN file, which contains all binary outputs from VALDRIFT, and selects and outputs the concentration and deposition fields for the time that the user desires. The user can specify that the base-10 logarithm of the concentration or deposition values be written, if desired, instead of the direct values. The selected data are then written to a text file, VALDRIFT.ASC. A portion of this text file is listed in Figure 3.10.

POSTPROC is an interactive program where the desired time and types of views of the data are specified. The view types possible are

1) Cross-sectional view of air concentrations - The concentrations for all flowtubes (N_Y x N_Z number of flowtubes) are written at a user-specified along-valley location.

2) Side view of air concentrations - The concentrations at all N_S along-valley locations are written for the N_Z number of flowtubes at the center of the valley.

3) Top view of air concentrations - The concentrations at all N_S along-valley locations are written for the N_Y number of flowtubes at the half-depth of the valley.

4) Ground-surface view of air concentrations - The air concentration at all N_S along-valley locations are written for the N_G number of flowtubes adjacent to the valley floor and sidewalls.

5) Ground-surface view of deposition - The deposition amount at all N_S along-valley locations are written for the N_G number of flowtubes adjacent to the valley floor and sidewalls.

Figure 3.10 shows the base-10 logarithm values of the 0800 LST concentrations for the Brush Creek Valley simulation. The cross-sectional view at S = 22 km is given first (after the general information about the particular simulation), followed by the side view of the concentrations for the first 37 S-locations proceeding down-valley from the domain origin.

Figures 3.11 and 3.12 show some of the results from the Brush Creek Valley simulation, as obtained using POSTPROC. Figure 3.11 shows the time evolution of the ground-level pesticide concentrations resulting from the six two-kilometer-long "flight paths" that occurred from 0530 through 0630 at the along-valley location centered at 19 km. The 0600 and 0800 LST panels in Figure 3.11 show the drift cloud moving down-valley in down-valley winds, and the 1000 and 1200 LST panels show the drift cloud moving up-valley after the winds shifted from down-valley to up-valley at 0830 LST. The winds remained up-valley through the remainder of the simulation. Figure 3.12 shows the time evolution of the vertical structure of the pesticide concentration at two along-valley locations, S = 15 km and S = 22 km. The 0600 LST panel at S = 22 km in Figure 3.12 is especially interesting, where the drift cloud is skewed to the left-sidewall of the valley. This simply shows the progression of the release where the first two-km-long flight (oriented along-valley) was for 10 minutes (from 0530 to 0540 LST) near the left sidewall, and the remaining five flights proceeded in equally-spaced increments across the valley, finishing with the sixth flight near the right sidewall from 0620 to 0630 LST. No material was released into the valley atmosphere after 0630 LST.

```
    ***** GENERAL INFORMATION *****

Length of domain (alongvalley/S_dir):  45000.0(m)
Number of grid points in S-direction:  100
Number of grid points in Y-direction:  7
Number of grid points in Z-direction:  7
The simulation beginning time:0530
The simulation ending time:   0800
Time step for writing concentrations to file:  30 (minutes)
------------------------------------------

The time you want is 0800 (LST), at this time:
TOPCBL =  118.222(m), TOPSBL =  695.300(m)
------------------------------------------

Cross-view at the cross-section of S =   21600.0(m)
Z,    Y: -->
   7 -.728E+01 -.731E+01 -.779E+01 -.840E+01 -.824E+01 -.775E+01 -.785E+01
   6 -.643E+01 -.647E+01 -.697E+01 -.753E+01 -.722E+01 -.671E+01 -.678E+01
   5 -.582E+01 -.591E+01 -.641E+01 -.686E+01 -.643E+01 -.593E+01 -.596E+01
   4 -.583E+01 -.601E+01 -.653E+01 -.675E+01 -.621E+01 -.573E+01 -.582E+01
   3 -.632E+01 -.665E+01 -.723E+01 -.744E+01 -.688E+01 -.636E+01 -.625E+01
   2 -.669E+01 -.711E+01 -.765E+01 -.782E+01 -.735E+01 -.681E+01 -.645E+01
   1 -.673E+01 -.706E+01 -.750E+01 -.762E+01 -.722E+01 -.676E+01 -.646E+01
------------------------------------------

Side-view along the valley center line
S,    Z: -->
   1 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   2 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   3 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   4 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   5 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   6 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   7 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   8 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
   9 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  10 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  11 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  12 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  13 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  14 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  15 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  16 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  17 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  18 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  19 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  20 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  21 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  22 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  23 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  24 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  25 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  26 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  27 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  28 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  29 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  30 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  31 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  32 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  33 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  34 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  35 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  36 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
  37 -.241E+02 -.241E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02 -.240E+02
```

FIGURE 3.10. Portion of VALDRIFT.ASC for Brush Creek Valley Simulation

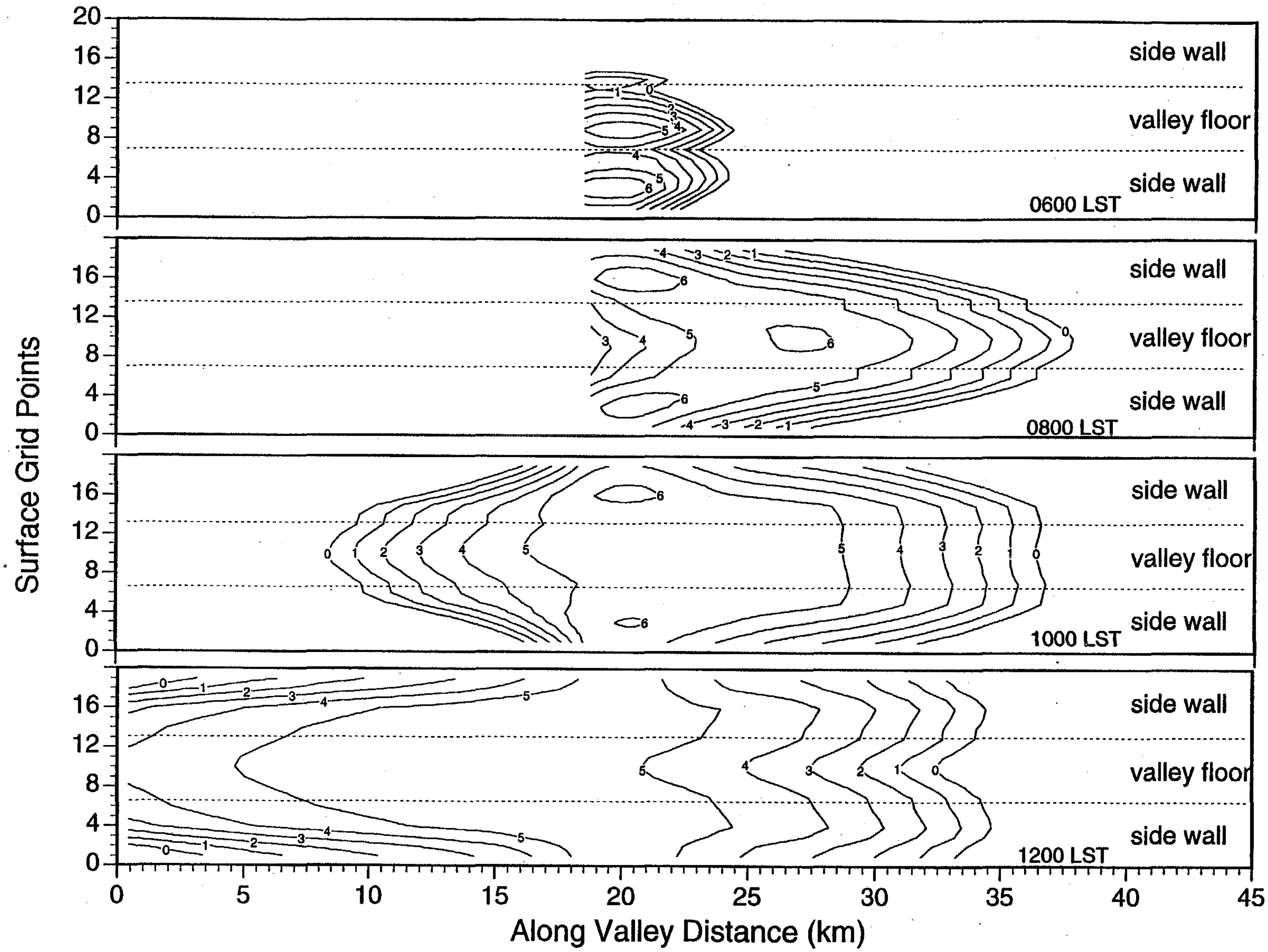


FIGURE 3.11. Concentration Contours (10^{-12} g/m^3) at the Surface at 0600, 0800, 1000, and 1200 LST. Contour labels are powers of 10.

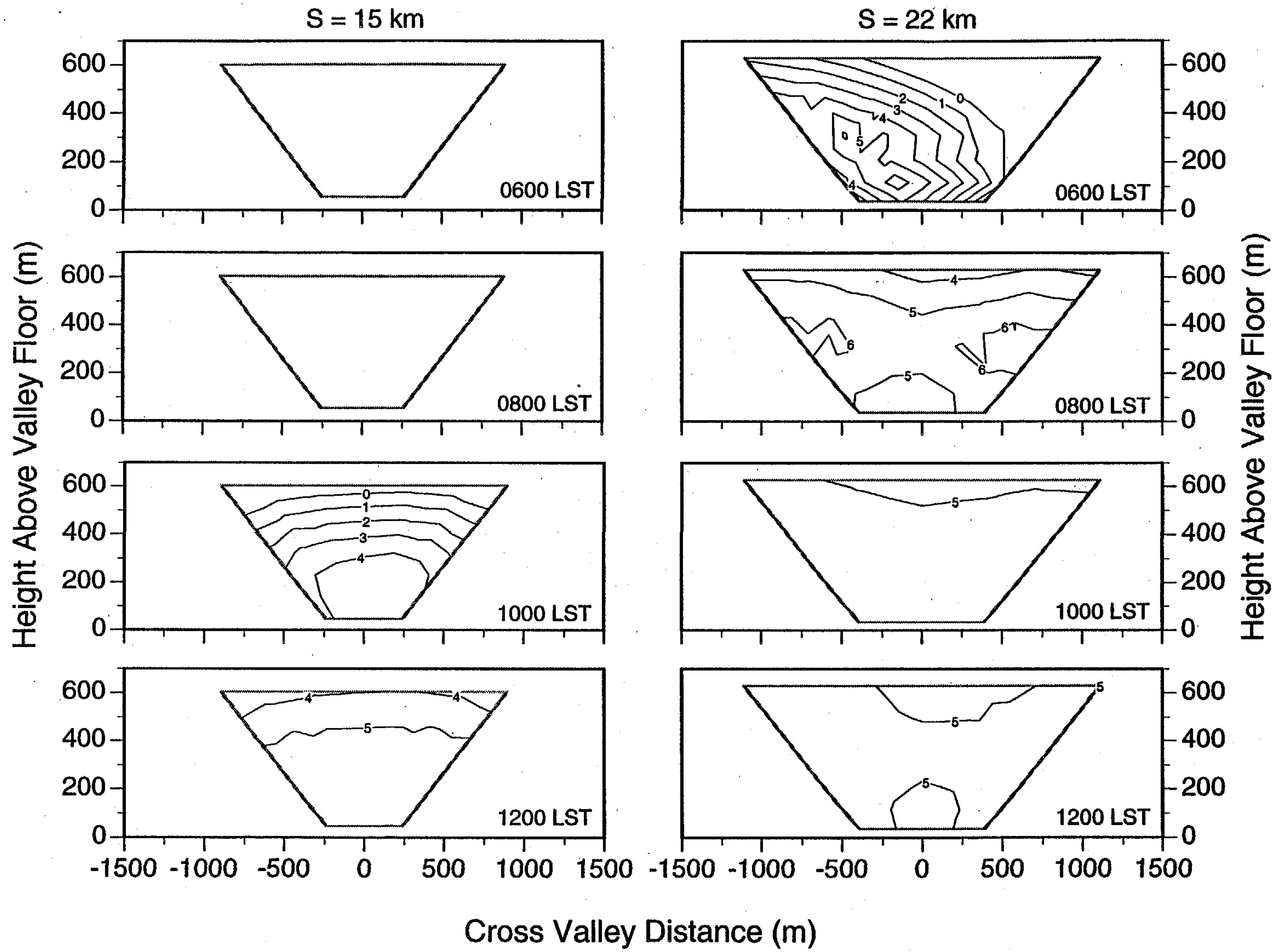


FIGURE 3.12. Concentration Contours (10^{-12} g/m^3) on Valley Cross Sections at S = 15 and 22 km at 0600, 0800, 1000, and 1200 LST. Contour labels are powers of 10.

4.0 CODE STRUCTURE

The VALDRIFT Fortran source code consists of a main program, 26 subroutines and an include file. A model flow chart is shown in Figure 4.1 and a listing of the source code is given in Appendix B. A listing of the post-processing program, POSTPROC, is also given in Appendix B, on pp. B.41 - B.45.

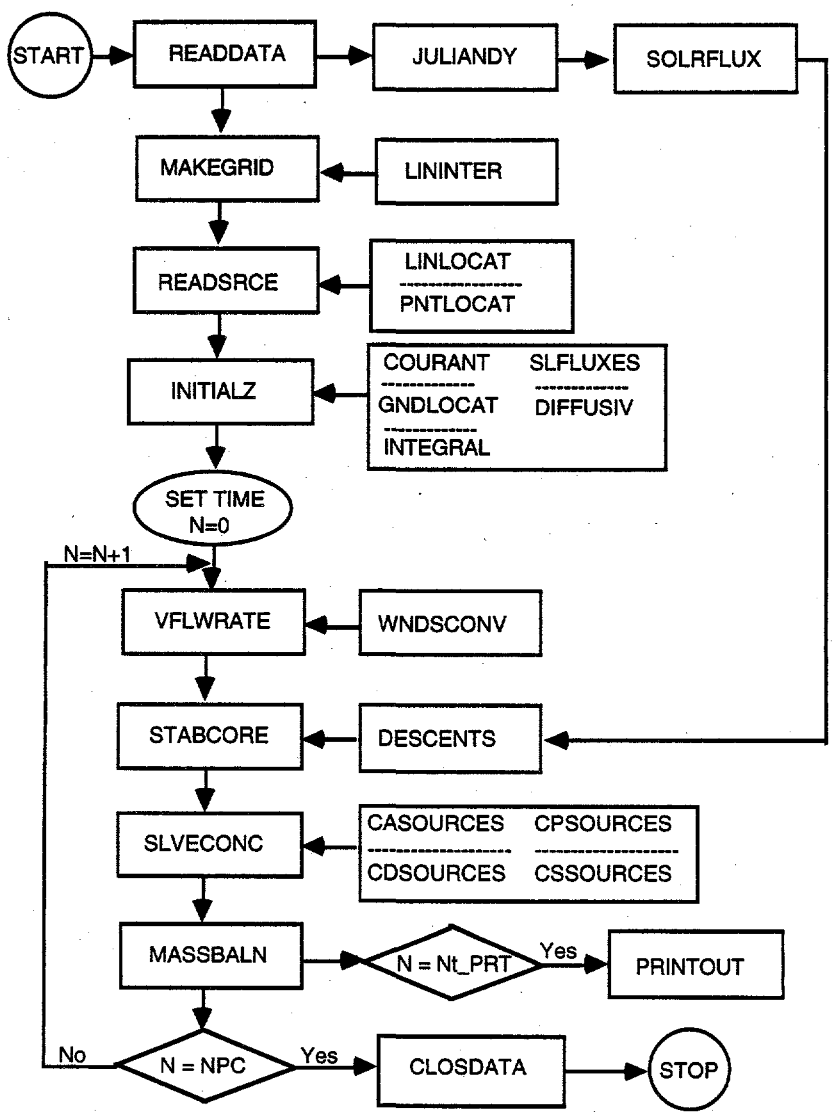


FIGURE 4.1. Flow Chart of VALDRIFT, Showing the Modular Structure of the Model

4.1 BASIC CODE FEATURES

The VALDRIFT model was written following a Fortran77 standard, and should be usable without modification on any computer system having an up-to-date Fortran compiler. The model is documented internally through the liberal use of comment statements. These comments explain the purpose of sections of code or individual FORTRAN statements, or serve as variable name definition tables.

Several special features or protocols are included in the code. The maximum dimensions of arrays are generally set using PARAMETER statements in the include file VALDRIFT.INC, and the actual dimensions (which must be less than or equal to these maximum dimensions) can be adjusted by users through the input files. If users find that the array sizes are too small for their simulation or too large for the memory capacity of a small computer they may easily change the array sizes by modifying the PARAMETER statement in the include file. The include file is listed in Appendix B on pages B.39-B.40.

The model was originally developed on Macintosh® computers (a Quadra 700 and a Mac IIci with a Radius/Rocket accelerator), and was then transferred to an IBM compatible computer (GATEWAY 2000) and a UNIX work station (SUN/690). After compilation, a single run takes roughly 2~3 minutes of central processor time per hour of simulation time on the Macintosh and IBM computers. It runs much faster on the UNIX workstation.

4.2 SUBROUTINE DESCRIPTIONS

Following is a brief technical description of the individual modules.

1. VALDRIFT - Main Program - This is the main program that controls the modules which, when taken together, form the VALDRIFT valley atmospheric drift, diffusion and deposition model. The function of the main program is to provide the basic structure of the model and to call the specific modules as required.
2. CASOURCE - Advective Effects Module - This subroutine computes the along-valley advective effects source/sink terms. The logical variable NEUMANN should be set to "TRUE" if constant flow boundary conditions at the starting point ($S = 0$) and/or at the ending point (S = NPS) are desired. If this condition is in effect, sources cannot be located on the boundaries.

3. CDSOURCE - Turbulent Diffusion Module - This subroutine computes turbulent diffusion source/sink terms using concentrations C(S,Y,Z,t). The turbulent diffusion source/sink terms include the lateral turbulent diffusion source/sink term Gama_cd_Y and the vertical turbulent diffusion source/sink term Gama_cd_Z.

4. CLOSDATA - Close Data File Module - This subroutine closes all input and output data files before the main program is terminated.

5. COURANTC - Courant Condition Module - This subroutine determines the time step for the explicit solution method using the Courant criteria for numerical stability. It also determines the number of time steps in the simulation, and the time step size required to meet the Courant numerical stability criterion for advection and diffusion calculations.

6 CPSOURCE - Dry Deposition Module - This subroutine computes dry deposition source/sink terms based on the dry deposition velocity Vd and the near-ground surface concentrations.

7. CSSOURCE - Release Rate Module - This subroutine sets the actual release rate term (Gama_cs) for all pollutant source points (point- and line-sources) in the computational domain.

8. DESCENTS - Inversion Top Descent Module - This subroutine calculates the heights of the inversion top and CBL top as a function of time after sunrise as the valley inversion is destroyed.

9. DIFFUSIV- Eddy Diffusivity Module - This subroutine uses a boundary layer parameterization method to calculate eddy diffusivities for stable, neutral, and unstable conditions.

10. GNDLOCAT - Ground Surface Location Module - This subroutine determines the indices (k,i,j) of all grid points on the ground surface including the valley floor and sidewalls.

11. INITIALZ - Initial Conditions Module - This subroutine sets the initial conditions for concentration fields C(S,Y,Z,t=0) = C_b, where C_b is the background concentration field, and sets the permeability of the lateral and vertical boundaries to advection and diffusion (0 - impermeable, 1 - permeable). It determines the mean valley topographical characteristics, and also determines the Prandtl wind profile weighting factor for estimating the peak winds on vertical profiles at the valley centerline.

12. INTEGRAL - Integral Module - This subroutine computes the integral over gamma associated with the along-valley volume flow rate in a flowtube for all cross sections.

13. JULIANDY - Julian Day Module - This subroutine computes the Julian day given the month, day, and year.

14. LININTER - Linear Interpolation Module - This subroutine calculates a value X_i ($X_1 < X_i < X_2$) by linear interpolation, where X_1 and X_2 are input data or known values.

15. LINLOCAT - Line Source Location Module - This subroutine determines the indices (k,i,j) of all grid points on a line source given the (S,Y,Z) coordinates of the beginning and ending points of the line source.

16. MAKEGRID - Making Grid Module - This subroutine takes the valley topographic characteristics and outputs each flowtube's grid in (S,Y,Z) coordinates.

17. MASSBALN - Mass Balance Module - This subroutine determines the pollutant mass balance over the computational domain for a single time step.

18. PNTLOCAT - Point Source Location Module - This subroutine determines the indices (k,i,j) of a point given the (S,Y,Z) coordinates.

19. PRINTOUT - Print Concentrations Module - This subroutine writes concentration and deposition fields to a text output file and a binary output file at specified intervals.

20. READDATA - Open/Read Data File Module - This subroutine opens data files (for example, VALDRIFT.FIL, ---.RS, and ---.TER) and reads input data from them.

21. READSRCE - Read Point /Line Source Module - This subroutine opens the point and line source file (---.REL), reads input data, and calculates the release rate in mass per minute per meter.

22. SLFLUXES - Surface Layer Fluxes Module - This subroutine calculates the surface layer (or the constant layer) fluxes using the Businger-Dyer (B-D) flux-profile relationship to determine the surface friction velocity u* and the deposition velocity V_d.

23. SLVECONC - Solve Concentrations Module - This subroutine integrates the conservation of species equation using a fully explicit scheme - Euler forward differencing in time, upwind differencing in the S-direction for the advection term, and central differencing in the Y- and Z-directions for diffusion. The Courant criteria must be satisfied in specifying dS and dt.

24. SOLRFLUX - Solar Flux Module - This subroutine calculates the time of sunrise, the day length, and the solar flux on a horizontal surface at solar noon for any site, given the latitude, longitude, and Julian Day using Whiteman and Allwine's (1986) method.

25. STABCORE - Stable Core Module - This subroutine calls Whiteman and Allwine's (1985) DESCENT module to calculate the stable core descent and determine the horizontal and vertical turbulent diffusivities K_y and K_z in the CBL or in the stable core.

26. VFLWRATE - Flow Rate Module - This subroutine inputs the wind data that drives VALDRIFT. The input winds should be from one height at the valley center for one or more stations. The winds must be evenly spaced in time (e.g., 15 minutes to 1 hour). This subroutine then solves Allwine's (1992) equation 71 on page 188 to compute the along valley volume flow rate (Vdot_S) at each cross-section using wind data. The air mass source/sink term is also computed.

27. WNDSCONV - Wind Decomposition Module - This subroutine converts the input wind speeds and directions into wind components relative to any desired direction. The positive u-component is in the specified direction. The positive v-component follows the conventional right-hand rule where the Z-axis is up. This subroutine can also do the reverse calculation, converting from components relative to any specified direction to wind direction (true north) and speed.

4.3 PARAMETER ASSIGNMENTS

Table 4.1 lists the parameter assignments in the VALDRIFT.INC include file.

TABLE 4.1. Values and Definitions of Parameters Used in VALDRIFT.

PARAMETER	VALUE	DESCRIPTION
MPS	200	Maximum number of points on S-direction
MPY	31	Maximum number of points on Y-direction
MPZ	31	Maximum number of points on Z-direction
MPT	10	Maximum number of input terrain cross-sections
MPM	1	Maximum number of input wind stations
MPR	50	Maximum number of grid cells containing sources
MPD	93	Maximum number of deposition points 2(MPZ-1) + MPY
MPS1	201	MPS + 1
MPY1	32	MPY + 1
MPZ1	32	MPZ + 1

5.0 CONCLUSIONS AND RECOMMENDATIONS

VALDRIFT version 1.0 is an atmospheric transport and diffusion model for use in well-defined mountain valleys. It is designed to determine the extent of drift from aerial pesticide spraying activities, but can also be applied to estimate the transport and diffusion of various air pollutants in valleys. The model is phenomenological - that is, the dominant meteorological processes governing the behavior of the valley atmosphere are formulated explicitly in the model, albeit in a highly parameterized fashion. The model is applicable to well-defined mountain valleys with relatively steep sidewalls (10° to 90°).

This report provides detailed technical information and a user's guide for VALDRIFT. The completion of this model is a key milestone in the U.S.D.A. Forest Service's strategic plan to improve predictions of pesticide drift from spraying operations in complex terrain. VALDRIFT will be incorporated into the Forest Service Cramer-Barry-Grim (FSCBG) modeling framework. The addition of VALDRIFT to this modeling framework will extend the range of modeling capability in a framework that is well-known to a large group of forest managers.

The completion of the VALDRIFT computer model and this accompanying user's guide is a significant step toward making a fast, scientifically-sound atmospheric dispersion model, for application in complex terrain, available to the general user community. The next step in VALDRIFT's evolution is to compare the model results with field data. PNL performed limited comparisons of VALDRIFT results with results from the U.S. Department of Energy's 1984 ASCOT (Atmospheric Studies in Complex Terrain) Brush Creek Valley tracer experiments. VALDRIFT results compared favorably with ground-level tracer concentrations several kilometers from point sources located in the deep Brush Creek Valley in Western Colorado. After comparisons with field data, VALDRIFT could be modified to include other physical processes (e.g., cross-valley flows and tributary flows) that are not currently treated in the model. The current version of VALDRIFT contains the structure to readily accommodate parameterizations of these other physical processes.

6.0 REFERENCES

Allwine, K. J. 1992. Atmospheric Dispersion in Mountain Valleys and Basins. PNL-7922, Pacific Northwest Laboratory, Richland, Washington.

Allwine, K. J. 1993. "Atmospheric Dispersion and Tracer Ventilation in a Deep Mountain Valley." J. Appl. Meteor. 32:1017-1037.

Bader, D. C., and C. D. Whiteman. 1989. "Numerical Simulation of Cross-Valley Plume Dispersion during the Morning Transition Period." J. Appl. Meteor. 28(7):652-664.

Bader, D. C., and T. B. McKee. 1985. "Effects of Shear, Stability and Valley Characteristics on the Destruction of Temperature Inversions." J. Climate Appl. Meteor. 24(8):822-832.

Bian, X., E. S. Takle, and S. Zhong. 1992. "A Theoretical Study of the Flux-Profile Relationships: Extension of the KEYPS Equation to the Tower Layer." Preprints, Tenth Symposium on Turbulence and Diffusion, 29 September-2 October 1992, Portland, Oregon, 184-187. Published by Amer. Meteor. Soc., Boston, Massachusetts.

Bilanin, A. J., M. E. Teske, J. W. Barry, and R. B. Ekblad. 1989. "AGDISP: The Aircraft Spray Dispersion Model, Code Development and Experimental Validation." Transactions of the ASAE. 32(1):327-334.

Businger, J. A., J. C. Wyngaard, Y. Izumi, and E. F. Bradley. 1971. "Flux Profile Relations in the Atmospheric Surface Layer." J. Atmos. Sci. 28:181-189.

Clements, W. E., J. A. Archuleta, and P. H. Gudiksen. 1989. "Experimental Design of the 1984 ASCOT Field Study." J. Appl. Meteor. 28(6):405-413.

Doran, J. C., T. W. Horst, and C. D. Whiteman. 1990. "The Development and Structure of Nocturnal Slope Winds in a Simple Valley." Bound.-Layer Meteor. 52(1-2):41-68.

Ekblad, R. B., and J. W. Barry. 1990. Selection and Verification of Complex Terrain Wind Flow Model for Spray Transport - Briefing Paper & Progress Report. USDA Forest Service Report No. 5E52P29 WIND.

Ekblad, R. B., C. D. Whiteman, K. J. Allwine, and J. W. Barry. 1991. "Wind Flow Model for Spray Transport in Complex Terrain." Paper presented at the 1991 International Summer Meeting of the American Society of Agricultural Engineers, June 23-26, 1991, Albuquerque, New Mexico.

Gryning, S.-E., and E. Lyck. 1983. A Tracer Investigation of the Atmospheric Dispersion in the Drynaes Valley, Greenland. RISO-R-481, RISO National Laboratory, DK-4000 Roskilde, Denmark, 67 pp.

Gudiksen, P. H., G. J. Ferber, M. M. Fowler, W. L. Eberhard, M. A. Fosberg, and W. R. Knuth. 1984. "Field Studies of Transport and Dispersion of Atmospheric Tracers in Nocturnal Drainage Flows." Atmos. Environ. 18(4):713-732.

Hewson, E. W., and G. C. Gill. 1944. "Meteorological Investigations in Columbia River Valley Near Trail, B.C." Bur. Mines Bull., U.S. Department of Interior, 453:23-228.

McNider, R. T. 1981. Investigation of the Impact of Topographic Circulations on the Transport and Dispersion of Air Pollutants. Ph.D. Thesis, Department of Environmental Science, University of Virginia. Available from University Microfilms, Ann Arbor, Michigan.

Panofsky, H. A., and J. A. Dutton. 1984. Atmospheric Turbulence. Models and Methods for Engineering Applications. John Wiley and Sons, New York, 397pp.

Segal, M., C. H. Yu, R. W. Arritt, and R. A. Pielke. 1988. "On the Impact of Valley/Ridge Thermally Induced Circulations on Regional Pollutant Transport." Atmos. Environ. 22(3):471-486.

Start, G. E., C. R. Dickson, and L. L. Wendell. 1975. "Diffusion in a Canyon Within Rough Mountainous Terrain." J. Appl. Meteor. 14:333-346.

Teske, M. E., J. F. Bowers, J. E. Rafferty, and John W. Barry. 1993. "FSCBG: An Aerial Spray Dispersion Model for Predicting the Fate of Released Material Behind Aircraft." Environ. Toxicology and Chemistry, 12:453-464.

Tyson, P. D. 1969. "Air Pollution Fumigation Conditions Associated with the Dissipation of the Mountain Wind and Onset of the Valley Wind over Pietermaritzburg." S. Afr. Geogr. J. 51:99-105.

Vergeiner, I., E. Dreiseitl, and C. D. Whiteman. 1987. Dynamics of Katabatic Winds in Colorado's Brush Creek Valley. J. Atmos. Sci. 44:148-157.

Wanner, H., and J. Hertig. 1984. "Studies of Urban Climates and Air Pollution in Switzerland." J. Climate Appl. Meteor. 23(12):1614-1625.

Whiteman, C. D. 1989. "Morning Transition Tracer Experiments in a Deep Narrow Valley." J. Appl. Meteor. 28(7):626-635.

Whiteman, C. D. 1990. "Observations of Thermally Developed Wind Systems in Mountainous Terrain." Chapter 2 in Atmospheric Processes Over Complex Terrain, (W. Blumen, Ed.), Meteor. Monogr. 23(45):5-42. Amer. Meteor. Soc., Boston, Massachusetts.

Whiteman, C. D., and K. J. Allwine. 1985. VALMET - A Valley Air Pollution Model. PNL-4728 Rev. 1, Pacific Northwest Laboratory, Richland, Washington.

Whiteman, C. D., and K. J. Allwine. 1986. "Extraterrestrial Solar Radiation on Inclined Surfaces." Environ. Software, 1:164-169.

Whiteman, C. D., and T. B. McKee. 1982: "Breakup of Temperature Inversions in Deep Mountain Valleys: Part II. Thermodynamic Model." J. Appl. Meteor. 21:290-302.

Willson, R., F. Shair, B. Reynolds, and W. Greene. 1983. "Characterization of the Transport and Diffusion of Pollutants in a Narrow Mountain Valley Region by Means of an Atmospheric Tracer." Atmos. Environ. 17(9):1633-1647.

APPENDIX A

EFFECTS OF VALLEY METEOROLOGY ON FOREST PESTICIDE SPRAYING

Reprint of PNL-7332

PNL-7332
UC-402

EFFECTS OF VALLEY METEOROLOGY ON FOREST PESTICIDE SPRAYING

C. David Whiteman

April 1990

Prepared for the
U.S. Forest Service
Missoula Technology and Development Center
under a Related Services Agreement
with the U.S. Department of Energy
Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Richland, Washington 99352

SUMMARY

Recent research has shown that many valleys, when undisturbed by external conditions such as cloudiness or high upper-level winds, undergo a regular diurnal evolution. Destruction of the nocturnal temperature inversion and reversal of the nocturnal down-valley flow occurs during a 3-1/2 to 5-hour period following sunrise, called the morning transition period. The processes responsible for temperature inversion breakup and wind system reversal are largely driven by the development of upslope flows in convective boundary layers (CBLs) that develop over the heated sidewalls. Aerial spraying of forest pesticides takes place in these growing CBLs. The important physical processes can be summarized:

- Spray released over the slopes before local sunrise will be carried down the slopes and down the valley but will remain fairly concentrated owing to reduced dispersion in the high stability atmospheric conditions. Non-deposited spray may adversely affect sensitive areas that may be located down-valley from the spray project.

- Upslope flows will form within a matter of minutes over sunlit slopes. Drift of a non-deposited plume will initially be upslope. If the spray plume escapes the boundary layer, however, or if the spraying occurs too high above the slope, it will be transported down the valley.

- The rate of development of the CBLs varies from location to location within the valley depending on time of sunrise, solar flux, surface energy budget, and temperature inversion destruction mechanism. When a boundary layer becomes deep and turbulent, spraying operations are no longer effective. This typically occurs first on the ridgetops and upper slopes. The lower slopes, because of the presence of the remnants of the nocturnal inversion above them, usually grow much more slowly and can be sprayed effectively much later in the morning. Late in the morning, the winds reverse to up-valley in the elevated remnants of the nocturnal inversion. Non-deposited pesticide will then drift up the slope and up the valley axis.

- After the temperature inversion is destroyed, the valley atmosphere will become part of a deep CBL. If coupled with strong winds aloft, the valley winds will be strong and turbulent, making conditions unsuitable for aerial spraying operations.

Recent gains in understanding of valley meteorology suggest that new modeling tools can now be applied to improve the planning and conduct of forest aerial spaying operations. Such tools may include digital topography models, solar shading algorithms, inversion breakup models, and flight path optimization models. Simple dispersion models could be modified to provide useful planning tools for forest spraying operations, and a long-term program could be initiated to apply recent boundary-layer growth models to three-dimensional topography.

1.0 INTRODUCTION

Pacific Northwest Laboratory conducted this study for the Missoula Technology and Development Center of the U. S. Department of Agriculture's Forest Service. The purpose of the study was to summarize recent research on valley meteorology during the morning transition period and to qualitatively evaluate the effects of the evolution of valley temperature inversions and wind systems on the aerial spraying of pesticides in National Forest areas of the western United States.

Aerial spraying of pesticides and herbicides in forests of the western United States is usually accomplished in the morning hours after first light, during the period known to meteorologists as the "morning transition period." The morning transition period is the post-sunrise period when the nocturnal down-valley flows are reversed to daytime up-valley flows and when the nocturnal temperature inversions are destroyed. The spraying continues until convective boundary layers and upslope flows develop sufficiently to render it difficult to place the spray where needed in the canopy. On an undisturbed clear day, the spraying must be terminated in mid to late morning. The normal sequence of meteorological events during the morning transition period is now well known, although the timing of these events varies significantly from valley to valley and from location to location within the same valley. This document describes the key physical processes that occur during the morning transition period on undisturbed days and the qualitative effects of these processes on the conduct of aerial spraying operations. Since the timing of valley meteorological events may be strongly influenced by conditions that are external to the valley, such as strong upper-level winds or the influence of clouds on the receipt of solar energy in the valley, some remarks are made on the qualitative influence of these processes. Section 4 of this report suggests ways to quantify some of the physical processes to provide useful guidance for the planning and conduct of spraying operations.

2.0 PHYSICAL PROCESSES

In this section, we summarize valley temperature inversion destruction observations collected in a number of Colorado valleys, pointing out typical characteristics of the meteorology of these valleys and the physical processes that must be included in realistic spray dispersion models simulating the inversion destruction period. In the last 10 years it has become clear that the inversion breakup patterns first observed in Colorado valleys in the 1970s are typical of patterns found in many other mountainous regions (Whiteman 1990), so that the understanding gained in the study of Colorado valleys is expected to be of widespread applicability. The following summary deals with conditions when upper winds are weak and weather conditions are undisturbed by large-scale traveling storm systems, such that the circulations within the valley are entirely locally produced. These local circulations are thermally driven, forced by pressure differences that arise owing to different rates of heating and cooling between the valley and its surroundings and between different locations within the topography. In such conditions the two main classes of local circulations are the up- and down-valley flows and the up- and down-slope flows (Figure 1) that have long been recognized in mountainous regions.

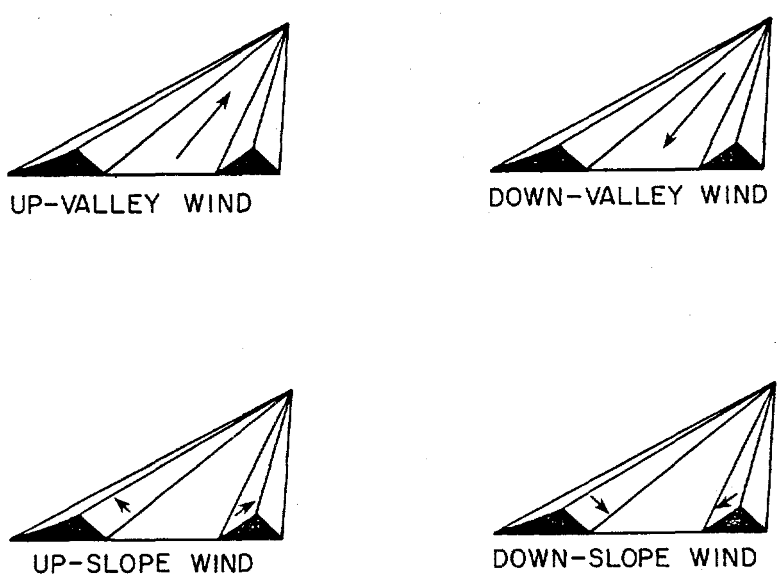


FIGURE 1. Wind System Nomenclature

2.1 Buildup of Valley Temperature Inversions

Cold air drains off the ridgetops and slopes of a valley during nighttime and collects in the valley, forming a surface-based temperature inversion (Figure 2a). In such an inversion, the coldest temperatures are found adjacent to the surface, and temperatures increase with altitude until the top of the inversion is reached. In the high-elevation, continental, mountainous western United States, the cooling power is strong enough on most clear nights to produce a surface-based temperature inversion that extends through the entire valley depth. Winds within the temperature inversion blow down the valley axis, although there are shallow downslope flows over the inclined sidewalls. The coldest temperatures and the strongest inversions generally occur at astronomical sunrise. This nocturnal inversion is generally destroyed following sunrise (Figures 2b through 2d), as solar energy provides the necessary heating at the surface. The ground or vegetative canopy absorb solar radiation and, in turn, heat the adjacent air. As the air warms it becomes less dense and eventually moves upward replacing cooler denser air in a process referred to as convection. Further details regarding the physical processes leading to the valley wind and temperature structure evolution during this period follow, illustrated by the Colorado observations.

2.2 Breakup of Valley Temperature Inversions

Temperature inversions in Colorado valleys are destroyed after sunrise following one of three patterns (Whiteman 1982) of temperature structure evolution (Figure 3). Note that Figure 3 uses potential temperature rather than actual temperature for the abscissa; potential temperature is preferred since actual temperature is a non-conservative variable under vertical motion because of the dependence of temperature on pressure. For those unfamiliar with this new variable, to a first approximation simply consider potential temperature to be actual temperature, and consider a constant potential temperature layer to be a well-mixed or convective boundary layer (CBL).

The first pattern, observed in the widest valley studied, approximates inversion destruction over flat terrain, in which the nocturnal inversion is destroyed after sunrise by the upward growth from the ground of a warming CBL. The elevated inversion or cold pool is eroded from the bottom but does not descend. The second pattern, observed in snow-covered valleys, differs significantly from the first. Here the growth of a CBL, which begins after sunrise, is arrested once the CBL has attained a depth of 25 to 50 m. The inversion is then destroyed as the top of the nocturnal inversion descends into the valley. Warming of the central region of the cross section is consistent with adiabatic subsidence heating produced by the descending motions. The third pattern of temperature structure evolution was observed in all of the valleys when snow cover was not present and describes the majority of case studies observed in field experiments. In this pattern, inversions are destroyed by a combination of two processes: the

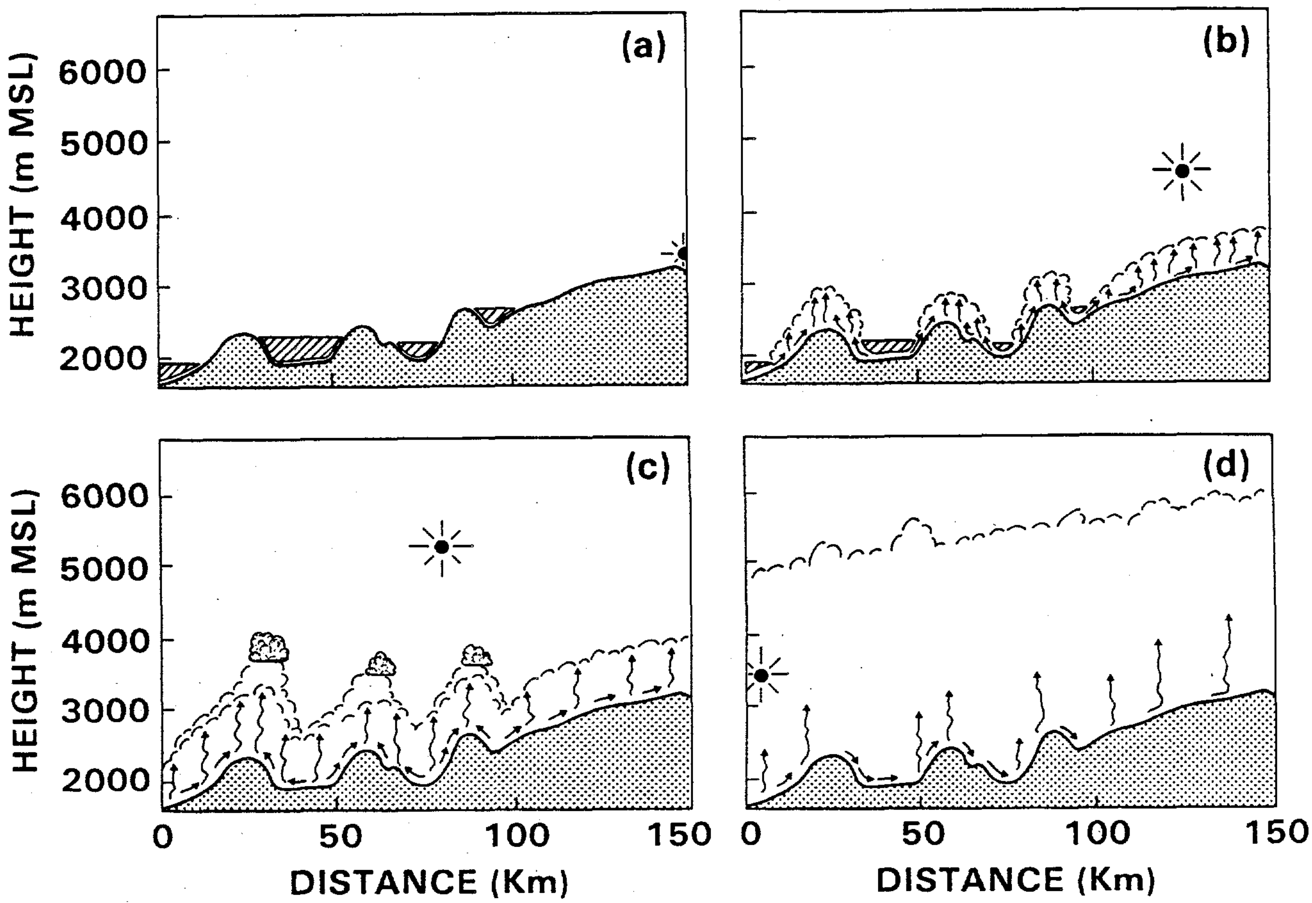
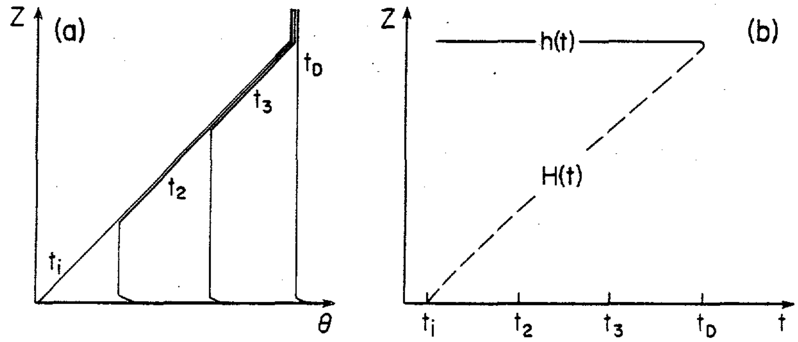
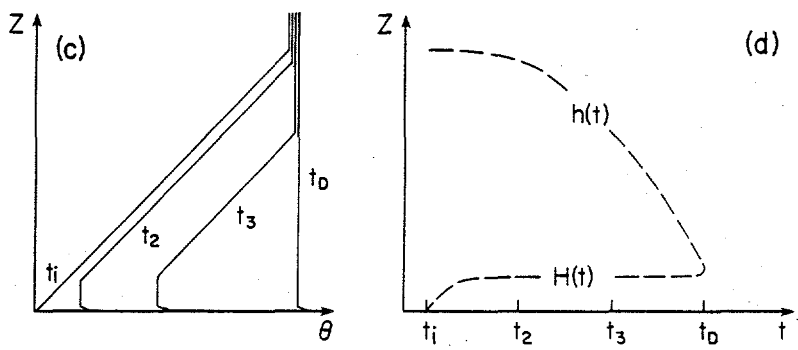


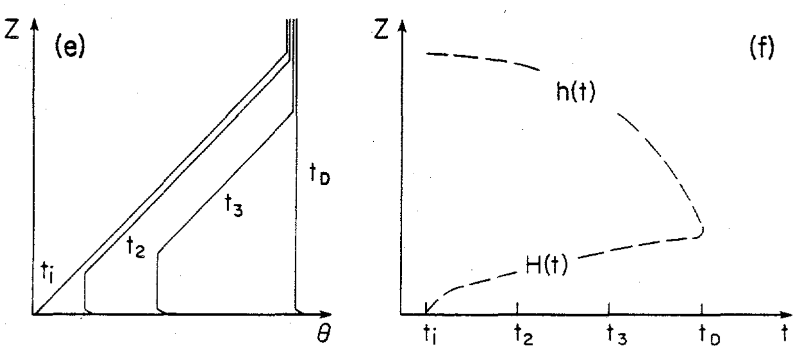
FIGURE 2. Schematic Depiction of Wind and Temperature Structure Evolution in Valleys. (a) At sunrise, the cold air has built up in the valley depressions forming surface-based temperature inversions. (b) After sunrise, convection and upslope flows begin to form over the heated slopes. Over the ridgetops, the convection results in rapid boundary layer growth, while, in the valley, the depth of convection is constrained by the overlying remnants of the nocturnal inversion. (c) By midday, the valley temperature inversions are destroyed and convection occurs in turbulent boundary layers over all heated surfaces. (d) By late in the day, the convective boundary layer reaches its maximum height over the mountain range and weak downslope flows occur over valley sidewalls that become shaded by the surrounding topography.



Pattern 1. Growth of CBL.



Pattern 2. Descent of inversion top and arrested growth of CBL.



Pattern 3. Descent of inversion top and continuous growth of CBL.

FIGURE 3. Three Patterns of Temperature Structure Evolution with Height (z) and Time (t) during the Inversion Breakup Period. Potential temperature profiles (θ) are on the left, and time-height analyses of convective boundary layer height (H) and inversion top height (h) are on the right.

continuous upward growth from the valley floor of a warming CBL and the continuous descent of the top of the nocturnal temperature inversion. Again, as for the second pattern of inversion destruction, warming of the elevated inversion layer above the CBL is caused by vertical advection, i.e., by simple sinking of warm air from aloft. In the Colorado valleys studied, the time required to break an inversion and establish a neutral atmosphere within the valley was typically 3-1/2 to 5 hours after sunrise. Temperature structure evolution during clear, undisturbed weather was surprisingly uniform from day to day and from season to season. Thus, in pesticide spraying work, one may be fairly confident of observing typical inversion breakup in a pre-spray campaign data collection program in undisturbed weather.

The common element of all three patterns of temperature structure evolution is the development of a CBL over the valley floor after it is illuminated by direct sunlight. Observations taken from the sidewalls also show the development of a CBL after direct sunlight illuminates the sidewall. Because of the shading effects of surrounding topography, the different valley surfaces can be illuminated at significantly different times, thus affecting the initiation of CBL growth. The temperature structure of the sidewall CBL is similar to that over the valley floor, but winds blow up the sidewall CBL at speeds of up to 3 m/s.

Five different temperature structure layers have been observed during the temperature inversion destruction period. Above the valley floor CBL and the sidewall CBL just mentioned is the stable core of the potential temperature inversion. This layer represents the remnants of the nocturnal surface-based valley temperature inversion. A layer above the stable core, to be called the 'neutral layer' in the subsequent text, appears to be part of a large-scale convective boundary layer that forms over the western slope of the Rocky Mountains. Such an intermediate circulation might form above any valley that is cut into the side of a mountain range. Above this layer is the free atmosphere.

Each of the five temperature structure layers, identified primarily by their potential temperature structure, can also be identified by the winds that prevail within them (see Figures 4, 5, and 6). During inversion destruction, the CBLs over the valley floor and sidewalls contain winds that blow up the floor of the valley and up the slopes. The neutral layer above the valley inversion has winds that blow up the inclined western slope of the Rocky Mountains during the day. Winds in the stable core typically continue to blow down-valley after sunrise until the stable core is nearly destroyed. Winds in the stable free atmosphere may blow from any direction with speeds determined by synoptic-scale pressure gradients. Despite variability in the strength and timing of reversal of the winds, the temperature structure evolves uniformly from day to day in individual valleys.

On the basis of the wind and temperature observations summarized above, an hypothesis has been developed to explain the temperature structure evolution (Figure 7). Since energy is required to change the temperature structure, and

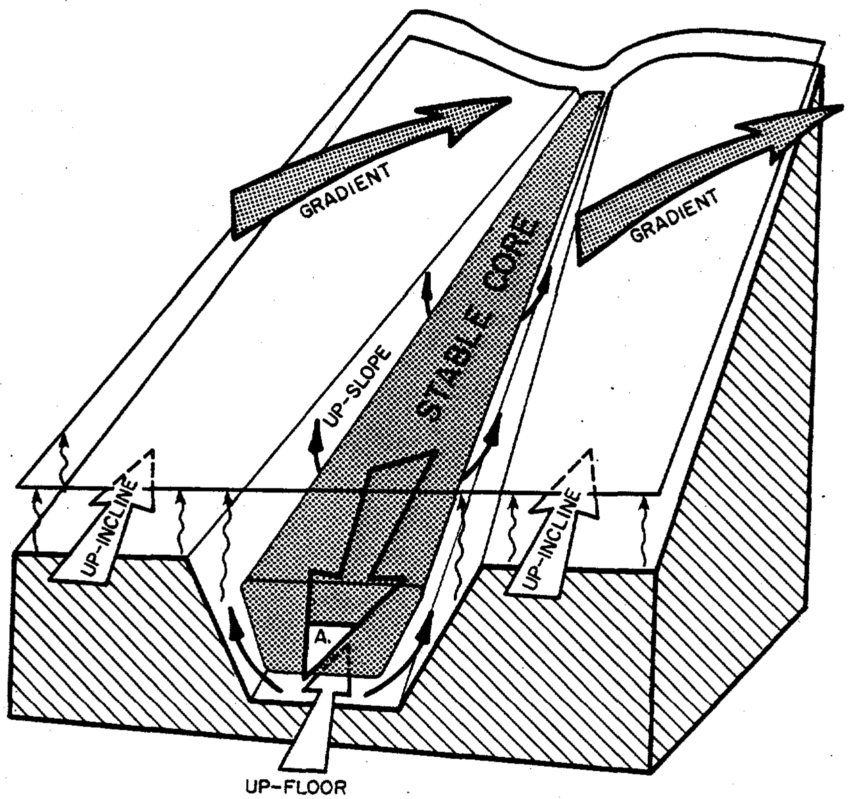


FIGURE 4. Typical Mid-morning Wind Structure Over and Within a Deep Valley on the Western Slope of the Rockies, Illustrating the Five Interrelated Wind Systems Identified in Field Studies.

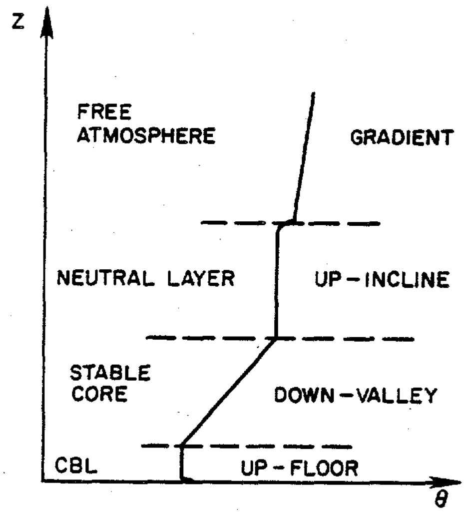


FIGURE 5. Relationship Between Temperature Structure Layers and Wind Systems. The temperature structure represents a typical mid-morning sounding from the floor of a deep valley.

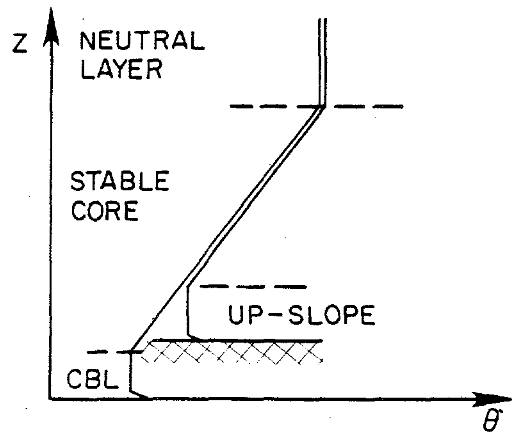


FIGURE 6. Dual Soundings from a Valley Floor and a Valley Sidewall Illustrating the Upslope Flow Found Within the CBL Over the Sidewall.

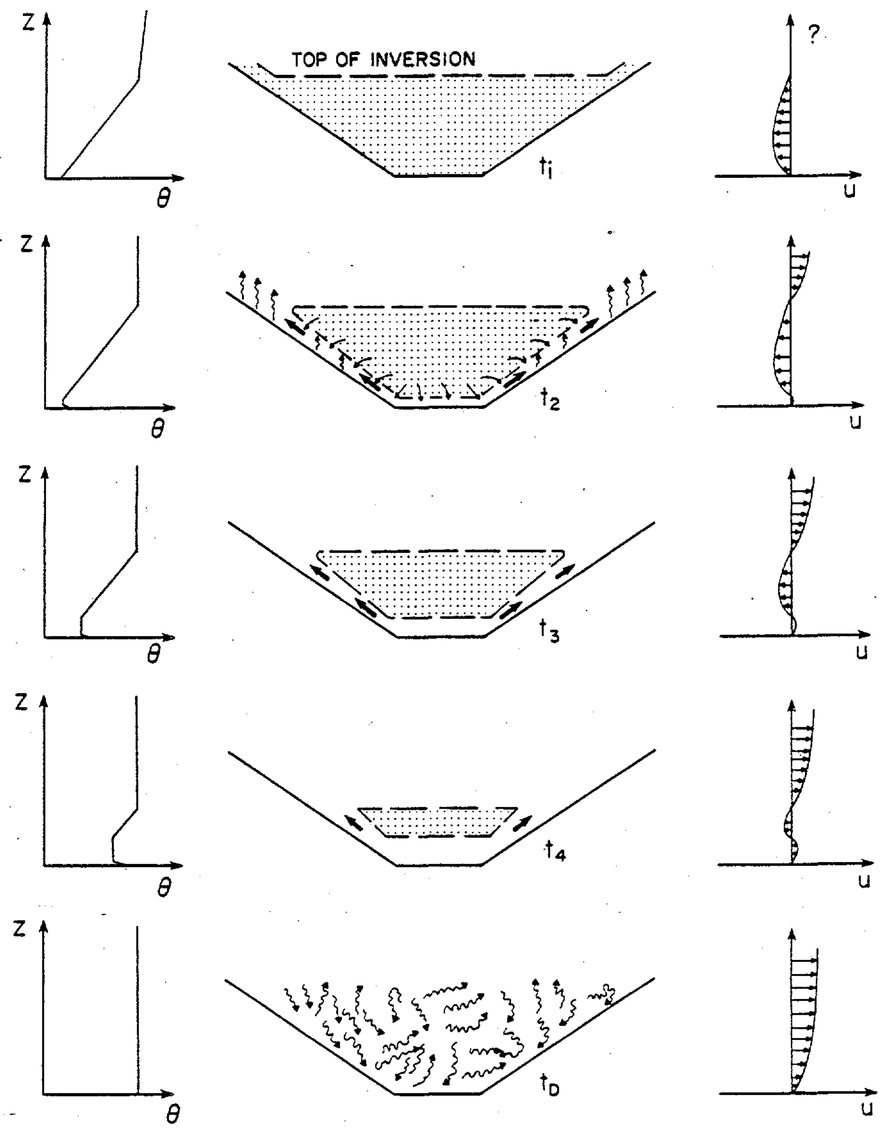


FIGURE 7. Illustration of the Hypothesis of Inversion Destruction. In the center of the diagram cross sections of a valley are shown at times t_i, t_2, t_3, t_4, and t_D. On the left are corresponding potential temperature profiles as taken from the valley center. On the right are corresponding up-valley wind components (u) as a function of height. At sunrise, t_i, an inversion is present in the valley. At t_2, a time after sunlight has illuminated the valley floor and slopes, a growing CBL is present over the valley surfaces. Mass and heat are entrained into the CBLs from the stable core above and carried up the sidewalls in the upslope flows. This results in a sinking of the stable core and growth of the CBLs (t_3 and t_4) until the inversion is broken (t_D) and a turbulent, well-mixed neutral atmosphere prevails throughout the valley depth. Down-valley winds continue to blow in the stable core during the inversion breakup period. Winds in the CBL below and in the region above the stable core often blow up-valley during this same period.

the change begins at sunrise, it is reasonable to propose that solar radiation is the driving force. A fraction of the solar radiation, received on the valley floor and sidewalls, is converted to the sensible heat flux that provides energy to the valley atmosphere. Sensible heat flux from a surface, as over flat terrain, causes a CBL to develop over the surface. Mass and heat are entrained into the CBL from the stable core above. Mass entrained into the valley floor and sidewall CBLs, however, is carried from the valley in the upslope flows that develop in the CBLs over the sidewalls. This removal of mass from the base and sides of the stable core causes the elevated inversion to sink deeper into the valley and to warm adiabatically during its subsidence and decreases the rate of growth of the underlying CBLs. The rate of warming depends directly on the rate of energy input into the valley atmosphere. This energy may be used to deepen the CBLs or to move mass up the sidewalls, allowing the stable core to sink. From this hypothesis a thermodynamic model of temperature inversion destruction has been developed (Whiteman and McKee 1982). This thermodynamic model forms the basis for parameterizations of inversion breakup in a valley air pollution model called VALMET (Whiteman and Allwine 1985).

Research in Colorado's Brush Creek valley in 1984 (Whiteman 1989, Bader and Whiteman 1989) identified a further physical process that may affect plume dispersion in narrow north-south oriented valleys. There, a cross-valley advection occurred during times of the day when one of the sidewalls was strongly illuminated by the sun and the other sidewall was in shadow or received weak solar heating. The effect of the cross-valley advection was to transport elevated plumes toward the strongly illuminated sidewall. This effect has been seen in other valleys (e.g., Urfer-Henneberger 1970) and has been investigated theoretically by Gleeson (1951).

2.3 The Valley Heat Budget

Moist surface conditions, cloudiness, or high albedo owing to snow cover may change the surface energy budget components so that sensible heat flux is reduced. In these conditions, inversion destruction will be delayed or an inversion may persist all day (see, e.g., the Yampa Valley observations of Whiteman and McKee 1982). These conditions are often advantageous for forest spraying, since the boundary layers over the sidewalls develop more slowly on such days, and acceptable spraying conditions persist much longer into the morning and afternoon than would be the case with clear skies and strong sensible heat flux.

Further, in a valley of complicated topography, the propagation of shadows from surrounding topography will ensure that some of the valley's slopes may be in shadow while others are in sunlight. Thus, we may speak of local sunrise times at points in the valley being later than the time of astronomical sunrise. Often the propagation of shadows is quite predictable to the casual observer, viz., the slow propagation of a shadow down one of the sidewalls, but in deeply dissected topography the propagation of shadows may be quite complicated, especially where the valley axis follows a sinuous course (see, e.g., Figure 8).

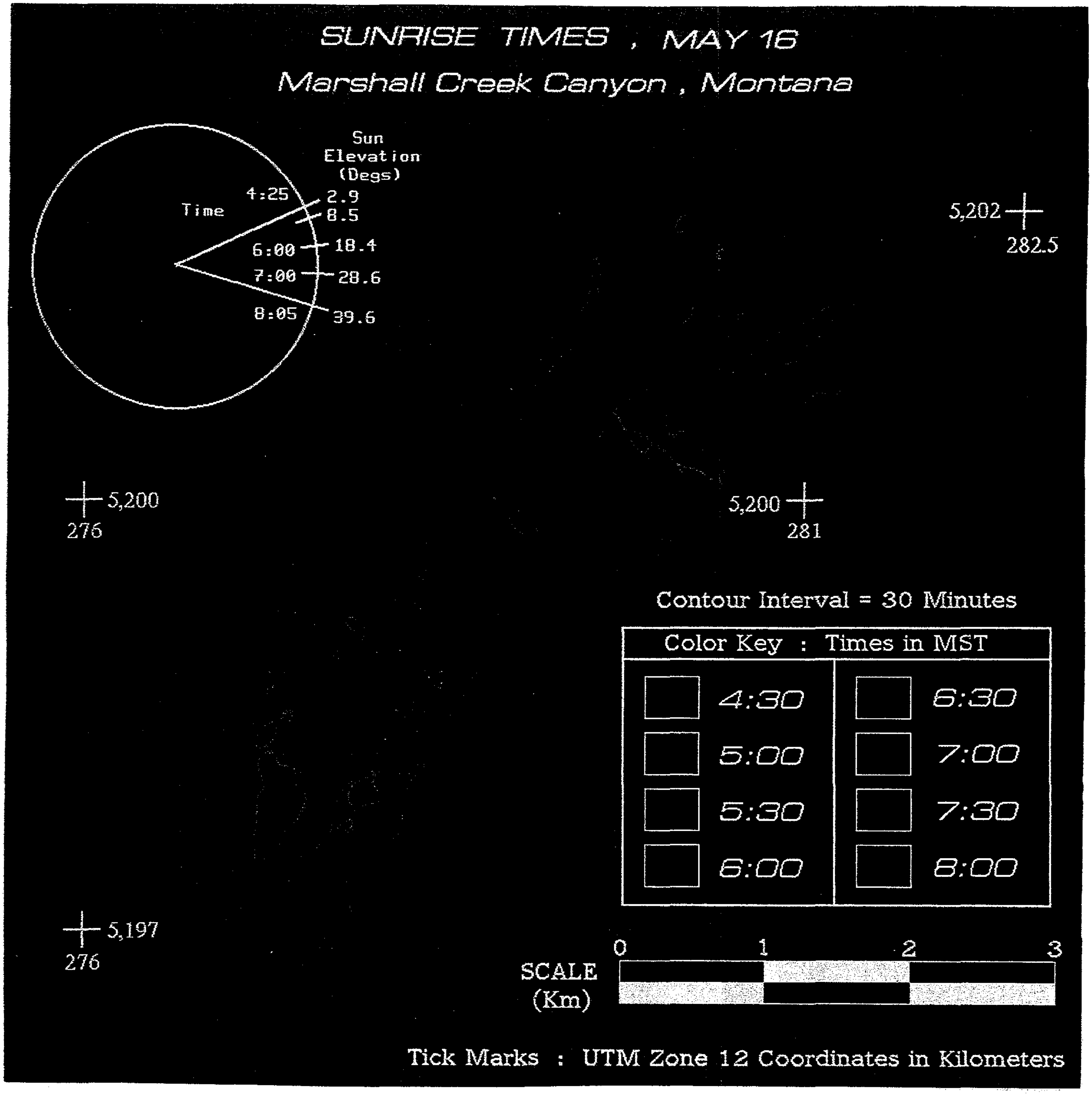


FIGURE 8. The Varying Times of Local Sunrise (Mountain Standard Time), Shown Here for Montana's Marshall Creek Canyon on May 16 Will Produce Spatial Contrasts in Boundary Layer Growth. Quantitative information from solar radiation and solar shading models such as the one shown here could be used to plan and optimize forest pesticide spraying operations in the country's national forests.

3.0 GENERAL CONSIDERATIONS: PESTICIDE DISPERSION IN FORESTED VALLEYS

The valley inversion destruction mechanism outlined above has important implications for the dispersion of pesticide sprays. These implications will be investigated first by assuming that the nocturnal inversion at sunrise contains pesticides that have been sprayed along the valley's axis in the mid-valley atmosphere. Other assumptions will be made subsequently. Typically, at sunrise, the plume would be carried down the valley in the nocturnal drainage flows, undergoing both vertical and horizontal dispersion. It is of interest that horizontal dispersion in these flows is known to be much greater than over flat terrain (Start et al., 1975) - a factor which may decrease the concentrations of a plume transported over a valley centerline relative to that over a plain. This important factor is, however, counterbalanced somewhat by the fact that the plume is channeled by the valley and that the plume centerline may directly impact an elevated terrain feature. We will focus initially on the dispersion of the pesticide on a valley cross section (Figure 9). After sunrise a convective boundary layer forms over the valley floor and sidewalls (Figure 9a). The subsequent dispersion in the stable core will be affected by two competing processes -- the sinking of the stable core and the growth of the CBL. The three inversion breakup patterns discussed above have the following dispersion implications:

1. Pattern 1 - Growth of convective boundary layer (Figure 9b): Pure growth of a CBL will result in the fumigation (Hewson et al., 1961) of a spray cloud at the valley floor as the CBL grows upward into the stable core. This process is favored when the slope flows are ineffective in removing mass from a valley and will thus occur in very wide and/or shallow valleys.

2. Pattern 2 - Sinking of stable core (Figure 9c): Failure of the CBL to grow once it has formed over the valley floor and sidewalls results in inversion destruction by sinking of the stable core. Thus, the spray cloud sinks into the top of a shallow mixed layer, producing high concentrations at the ground. The spray plume, once entrained into the CBL, is advected up the sidewalls and dispersed into the neutral layer aloft. This process is favored for narrow-to-wide valleys when sensible heat flux is weak.

3. Pattern 3 - Combination (Figure 9d): A combination of CBL growth and stable core descent results in the sinking of the spray cloud into the top of a growing mixed layer. Spray concentrations should be intermediate between the two previous cases. This pattern is the most common one in Colorado Mountain valleys.

The above discussion focuses on plume dispersion on a valley cross section. However, winds in the stable core, as mentioned above, blow down the valley until the inversion is nearly broken. Thus, pesticide sprayed into the stable core encounters a stable environment where diffusion is quite limited and will be carried down the valley, often toward populated or other sensitive areas.

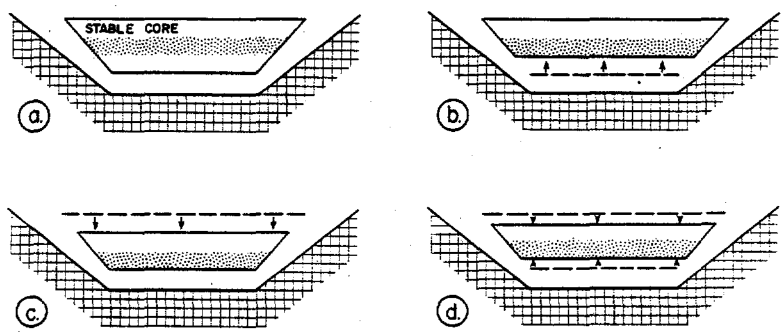


FIGURE 9. Dispersion Implications of CBL Growth and Inversion Top Descent.

Actual forest pesticide spraying, rather than occurring at an elevated position in the mid-valley atmosphere, is conducted by spray aircraft over the sidewalls above the forest canopy. Thus, after local sunrise the spray is released into the growing CBL, rather than into the stable core. When this boundary layer is shallow and the upslope winds within the layer are weak, the pesticide plume will deposit well in the canopy, aided by the downward motions from wingtip vortices. Any non-deposited chemical will generally drift up the slope and concentrations will be decreased by mixing caused by the turbulent winds within the upslope-flowing layer. A different situation arises, however, when the pesticide is sprayed later in the morning into a deep, fully-developed slope boundary layer. Then, flying conditions will be rough, and the spray plume may be broken up by strong convective eddies. A smaller proportion of the plume will be deposited and a larger drift will occur. Fortunately, however, this drift will be up-slope and pesticide will be well mixed through the deep layer so that concentrations will be low. The drift of the plume will rarely be toward sensitive population centers, since such centers are generally located down-slope on the valley floor. The most ideal spraying conditions should occur after the slope is illuminated by the sun, but before the convective boundary layer becomes deep and fully developed. The CBL on the ridgetops and at the upper levels of the sideslopes develops very rapidly after sunrise, since the stable core descends into the valley and these convective layers are not constrained by an overlying inversion layer. As these boundary layers develop, mixing may increase the exposure of the ridgetops to strong upper-level winds, further decreasing the chances of getting pesticide deposition on target vegetation.

4.0 CONCLUSIONS REGARDING FOREST PESTICIDE SPRAYING

The above description of valley meteorology during the morning transition period results in the following implications for forest spraying activities:

• Spray released before sunrise or over shaded slopes will be released into shallow stable boundary layers over the slopes. Drainage flows in these boundary layers will be downslope. The overlying down-valley flows typically superimpose a down-valley wind component on these layers, so that any drift of the spray will move downslope and down-valley. Pre-sunrise spraying may adversely affect population centers or sensitive areas that are downslope or down-valley during this time period. When plume impacts occur, concentrations may be quite high because of the poor diffusion environment along the transport path in the remnants of the strong nighttime inversion.

• Initially, shallow CBLs form over sunlit slopes. Shortly after they are sunlit (typically 10 to 15 minutes), an upslope flow will form. Drift of a non-deposited plume will then be upslope. If the plume escapes from the boundary layer (or if the spraying occurs too high above the slope), it will be transported down the valley in the remnants of the stable core.

• Convective boundary layers develop rapidly over the ridgetops and upper slopes after they become sunlit, because they are not capped by the strongly stable nocturnal inversion. Especially rapid CBL development may occur over the upper slopes that face the morning sun. When upper winds are moderate or strong, growth of the CBLs will bring the stronger winds down onto the upper slopes. The upper slopes would then become more and more turbulent, with the normal thermal convection processes being supplemented by transport of horizontal momentum from aloft. The rate of CBL development over the upper slopes will strongly affect the spraying environment there.

• Once temperature inversion destruction begins, the top of the temperature inversion will sink into the valley, exposing more of the upper slopes to intense CBL development. Winds within the stable core during this time will continue to blow down the valley, but the speed will typically be decreasing until, just before the inversion is destroyed (3-1/2 to 5 hours after astronomical sunrise on a clear day), the winds will switch to up-valley throughout the valley's volume.

• During the inversion destruction period, boundary layers on the lower slopes will be growing much more slowly because of the strongly stable layer (the remnants of the nocturnal inversion) that overlies them.

• Given the above processes, rates of CBL development depend strongly on the local amount of insolation (which varies from place to place within the valley), the surface energy budget, and the overlying temperature structure. The ridgetops and upper slopes should be sprayed first, because the dispersion environment there rapidly becomes too turbulent for continued spraying. The spraying environment on the lower slopes, in contrast, is suitable for spraying for a much longer time. Spraying conditions can be quite different on opposing slopes during the morning transition period, especially in north-south oriented valleys where the opposing slopes have quite different heating functions. The east-facing slope will receive solar input earlier, the CBL will develop earlier and more rapidly, and the differential heating of the two slopes may result in a cross-valley flow toward the heated slope.

- After the temperature inversion is destroyed, the valley becomes well coupled with the winds aloft. If these winds are weak, a thermally driven up-valley circulation will prevail within the valley. The absence of a strong capping inversion above the surfaces will result in a deep and turbulent CBL over all the valley's surfaces. If the upper winds are strong, they may superimpose their own wind direction on the valley, and winds may be correspondingly stronger and more turbulent, making conditions unsuitable for aerial spraying operations.

5.0 RECOMMENDATIONS

Our present understanding of valley meteorology during the morning transition period has progressed to the point where some useful planning tools could be constructed to optimize a forest pesticide spraying campaign. Simple models could be developed for use in the field to optimize the vectoring of spray planes so that spraying could be accomplished in the most suitable locations within a drainage area before convective boundary layers became too deep and strong. The area of effective pesticide coverage in a given spray campaign could be optimized by such a model. Digital topography models and solar shading algorithms could be developed to aid in this determination. Inversion breakup models and some simple air pollution dispersion models (Whiteman and McKee 1982, Whiteman and Allwine 1985) could be modified for support of forest spraying campaigns. Finally, some promising models of boundary layer growth on slopes have been formulated in the last several years that could be applied to the forest spraying problem. A model by Brehm (1986) was published in 1986, while two recent models, one by Dr. U. Schumann at the German Aerospace Research Establishment and one by Dr. J. Egger of the University of Munich, are presently in press. These models could be formulated in three dimensions, raising the possibility of using actual digital terrain, solar shading, and boundary layer growth models in a single package. Such boundary layer growth models have not yet been tested in real valleys, so that field experiment campaigns should be an important component of future work.

While the approaches mentioned in the previous paragraph could be applied now, several key research questions remain that will need to be addressed in order to make further progress. Some of these issues are being addressed now in ongoing research funded by the U.S. Department of Energy's Atmospheric Studies in Complex Terrain (ASCOT) program and in other federal programs. Very little work has, so far, been accomplished dealing with variations in surface energy budgets over complex terrain areas. Such energy budgets, which have a strong influence on the development of local circulations and growth of boundary layers, will be affected by clouds. Few studies have dealt with the influence of clouds. Further, much meteorological research remains to be done on forest canopies and their effects on valley heat budgets and locally developed circulations. Finally, the influence of external processes on valley circulations has not received enough attention. Key research questions involve the coupling of valley and above-valley circulations. This coupling is a quite frequent phenomenon in the valleys of the western United States.

The recent improvements in understanding of the evolution of the valley's atmosphere during the inversion destruction period could be incorporated in algorithms, planning tools, simple models, or numerical models, and should provide an improved basis for the planning and conduct of future aerial spraying operations.

6.0 REFERENCES

Bader, D. C. and C. D. Whiteman. 1989. "Numerical Simulation of Cross-Valley Plume Dispersion During the Morning Transition Period." J. Appl. Meteor. **28**:652-664.

Brehm, M. 1986. "Experimentelle und numerische Untersuchungen der Hangwindschicht und ihrer Rolle bei der Erwärmung von Tälern. [Experimental and Numerical Investigations of the Slope Wind Layer and its Role in the Warming of Valleys]." Wiss. Mitt. Nr. 54, dissertation, Universität München - Meteorologisches Institut, München, FRG.

Gleeson, T.A. 1951. "On the Theory of Cross-Valley Winds Arising From Differential Heating of the Slopes." J. Meteor. **8**:398-405.

Hewson, E.W., E.W. Bierly, and G.C. Gill. 1961. "Topographic Influences on the Behavior of Stack Effluents." Proc. Amer. Power Conference. **23**:358-370.

Start, G.E., C.R. Dickson, and L.L. Wendell. 1975. "Diffusion in a Canyon Within Rough Mountainous Terrain." J. Appl. Meteor. **14**:333-346.

Urfer-Henneberger, C. 1970. "Neure Beobachtungen über die Entwicklung des Schönwetterwindsystems in einem V-förmigen Alpental (Dischmatal bei Davos). [New Observations of the Development of a Clear Weather Wind System in a V-Shaped Mountain Valley (Dischma Valley Near Davos)]." Arch. Meteor. Geophys. Bioklim., Ser. B **18**:21-42.

Whiteman, C. D. 1982. "Breakup of Temperature Inversions in Deep Mountain Valleys: Part I. Observations." J. Appl. Meteor. **21**:270-289.

Whiteman, C. D. 1989. "Morning Transition Tracer Experiments in a Deep Narrow Valley." J. Appl. Meteor. **28**:626-635.

Whiteman, C. D. 1990. "Observations of Thermally Developed Wind Systems in Mountainous Terrain." Chapter 2 in Current Directions in Atmospheric Processes Over Complex Terrain, (W. Blumen, Ed.), Meteorological Monographs, American Meteorological Society, Boston, Massachusetts, 5-42.

Whiteman, C. D., and K. J. Allwine. 1985. VALMET - A Valley Air Pollution Model. PNL-4728, Rev. 1. [Available from NTIS as PB86-104106/AS]. Pacific Northwest Laboratory, Richland, Washington.

Whiteman, C. D., and K. J. Allwine. 1986. "Extraterrestrial Solar Radiation on Inclined Surfaces." Environ. Software, **1**:164-169.

Whiteman, C. D., and T. B. McKee. 1982. "Breakup of Temperature Inversions in Deep Mountain Valleys: Part II. Thermodynamic Model." J. Appl. Meteor., **21**:290-302.

APPENDIX B

FORTRAN SOURCE CODE

```
      Program VALDRIFT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995        K. Jerry Allwine and Xindi Bian       VALDRIFT 1.0
C                             Pacific Northwest Laboratory
C                             PO Box 999
C                             Richland, WA  99352
C
C     This program computes air concentrations and deposition fields
C     for time-varying point and line sources located in a valley.  It
C     is based on the formulation given in Allwine (1992).  This model
C     is applicable to well-defined valleys with sidewall angles
C     roughly > 10° and ≤ 90°.
C
C     The valley is approximated as a series of cross-sections
C     specifying the valley floor width, sidewall angles, elevation of
C     the valley floor and elevation of the ridgetops.  The modeling
C     domain should extend sufficiently in the upvalley and downvalley
C     directions to contain the extent of possible spray drift.
C
C     The model is 3-D (along-valley coordinate, S, cross-valley
C     coordinate, Y, and vertical coordinate Z) and the conservation
C     of species equation is integrated using a fully explicit finite
C     differences scheme consisting of forward Euler in time and upwind
C     differencing for advection, and central differencing for
C     diffusion.  Air mass is also conserved.
C
C     The meteorological inputs required are winds from one station
C     at the center of a valley cross-section at one height.  The winds
C     must be evenly spaced in time (e.g., 15 minute to 1 hour).  The
C     turbulent diffusivities need to be specified for nighttime
C     and daytime conditions.  These diffusivities are constant in time
C     and space.  The inversion characteristics of the valley atmosphere
C     before sunrise need to be specified.  The release information
C     required are the starting and ending locations of the flight
C     paths, the starting and ending times of the flights, and the
C     release rates.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Set TESTOUT to .TRUE. if want diagnostics written to .TRC file.
      TESTOUT = .FALSE.

C  ** Open output files, and open input files and read input data.
      Call Read_Data (*998)

C  ** Build computational grid.
      Call Make_Grid (*998)

C  ** Set clocks.
      clk = clk_b                         !minutes of day.
      clk_met = clk + dtM                 !time step to update met data.
      Hr_clk = clk/60.
      IHr_clk = INT(Hr_clk+.001)
      RMn_clk = (Hr_clk-Real(IHr_clk))*60.

C  ** Read source and release data.
      Call Read_Source (*998)

C  ** Set all initial fields.
      Call Initial

C  ** Determine flow field from first meteorological record.
      Call V_Flow_Rate

C  ** Determine time step to satisfy Courant condition.
      Call Courant (*998)
      dt_mn = dt/60.                      !time step in minutes.
      rts = dt/dS

C  ** Loop on time.
      Call Print_C
      nti = 0
      Write (*,100) nti, IHr_clk, RMn_clk, clk
      Do nti = 1, NPC
        clk = clk + dt_mn
        Hr_clk = clk/60.
        IHr_clk = INT(Hr_clk+.001)
        RMn_clk = (Hr_clk-Real(IHr_clk))*60.
        If (clk .gt. clk_met) Then
          clk_met = clk_met + dtM
```

```
          Call V_Flow_Rate
        EndIf
        Call KyKz_Stable_Core (nti)
        Call Solve_C (nti)
        Call Mass_Balance (nti)
        If (mod(nti,Nt_PRT).eq.0) Then
          Call Print_C
          Write (*,100) nti, IHr_clk, RMn_clk, clk
        EndIf
      EndDo

      GoTo 999
  998 Write (*,*) '  Abnormal Termination from VALDRIFT'
C  ** Close files.
  999 Call Close_Data

      Write (*,*) '  <RETURN> to exit'
      Read (*,*)
      Stop        '  VALDRIFT Completed'
  100 Format ('  Time Step =',I5,'   Hr =',I3,'   Mn =',F5.1,
     &          '   Clk =',F7.1)
      End
```

```
      Subroutine Ca_Source_Sink (k, nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian            VALDRIFT 1.0
C
C     This subroutine computes the advective effects source/sink terms
C     (g/s/m) in the conservation of species equation.  Lateral and
C     vertical advection terms are currently set to 0 throughout the
C     domain.
C
C     The longitudinal advective terms at the boundaries can be
C     determined from either: 1) assuming zero concentration gradient at
C     the boundaries, or 2) from the concentration specified on the
C     boundary.  Set NEUMANN to .TRUE. if want zero gradient boundary
C     conditions at S=0 and S=NPS.  Zero gradient means that the
C     concentration upwind from the boundary is assumed to be equal to
C     the concentration in the first downwind cell, rather than the
C     concentration specified at the boundary (typically = 0).  During
C     zero gradient boundary conditions sources cannot be located on
C     either boundary.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Determine lateral advection source/sink terms.
      Do i = 1, NPZ
        Do j = 1, NPY
          Gama_ca_Y(i,j) = 0.
        EndDo
      EndDo

C  ** Determine vertical advection source/sink terms.
      Do i = 1, NPZ
        Do j = 1, NPY
          Gama_ca_Z(i,j) = 0.
        EndDo
      EndDo

C  ** Determine the longitudinal advection source/sink terms.
      Do j = 1, NPY
        Do i = 1, NPZ
          k2 = k
          k1 = k - 1
          If (Vdot_S(k,i,j) .le. 0.) k2 = k + 1
          If (Vdot_S(k-1,i,j) .le. 0.) k1 = k
          If (NEUMANN .and. k.eq.1 .and. Vdot_S(k-1,i,j).gt.0.) Then
            Gam1 = Vdot_S(k-1,i,j)*C_i_j_old(k2,i,j)   !zero gradient BC
          Else
            Gam1 = Vdot_S(k-1,i,j)*C_i_j_old(k1,i,j)
          EndIf
          If (NEUMANN .and. k.eq.NPS .and. Vdot_S(k,i,j).lt.0.) Then
            Gam2 = Vdot_S(k,i,j)*C_i_j_old(k1,i,j)     !zero gradient BC
          Else
            Gam2 = Vdot_S(k,i,j)*C_i_j_old(k2,i,j)
          EndIf
          Gama_ca_S(i,j) = -(Gam2 - Gam1)/dS
        EndDo
      EndDo

C  ** Write results.
      If (TESTOUT) Then
        If (mod(nti,Nt_PRT).eq.0) Then
          If (Mod(k,NS_PRT) .eq. 0) Then
            Write (13,*) 'Gama_ca_S: [S: k =',k,']'
            Write (13,*) 'Z,    Y: -->'
            Do i = 1, NPZ
             ii = NPZ - i +1
             Write (13,101) ii,(Gama_ca_S(ii,j),j = 1,NPY)
            EndDo
            Write (13,*) 'Gama_ca_Y: [S: k =',k,']'
            Write (13,*) 'Z,    Y: -->'
            Do i = 1, NPZ
             ii = NPZ - i +1
             Write (13,101) ii,(Gama_ca_Y(ii,j),j = 1,NPY)
            EndDo
            Write (13,*) 'Gama_ca_Z: [S: k =',k,']'
            Write (13,*) 'Z,    Y: -->'
            Do i = 1, NPZ
             ii = NPZ - i +1
             Write (13,101) ii,(Gama_ca_Z(ii,j),j = 1,NPY)
            EndDo
          EndIf
```

```
           EndIf
         EndIf

         Return
     101 Format (i4,11(1x,E9.3))
         End
```

```
      Subroutine Cd_Source_Sink (k, nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine computes the lateral (Gama_cd_Y) and the vertical
C     (Gama_cd_Z) turbulent diffusion source/sink terms. The
C     longitudinal (Gama_cd_S) terms are set equal to zero (currently
C     no diffusion in the S-direction).  Diffusion through the
C     boundaries are controlled by the quantities BC_Y_left, BC_Y_right,
C     BC_Z_top, and BC_Z_bot.  Each quantity is set between 0 to 1,
C     which is the multiplier on the gradient diffusion through the
C     respective boundary.  If the value is zero, then no diffusion
C     occurs through the boundary.
C
C     The top boundary is the only boundary that is currently
C     permeable to diffusion.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Determine longitudinal diffusion source/sink terms.
      Do j = 1, NPY
        Do i = 1, NPZ
          Gama_cd_S(i,j) = 0.
        EndDo
      EndDo

C  ** Determine lateral diffusion source/sink terms.
      Do i = 1, NPZ
        factr = dZ_i(k,i)/dY_i_bar(k,i)
C  **   Left boundary.
        Gama_cd_Y(i,1) = factr * K_y(i,1) *
     &       ( (C_i_j_old(k,i,2)-C_i_j_old(k,i,1)) +
     &         (C_i_j_old(k,i,0)-C_i_j_old(k,i,1))*BC_Y_left(k,i) )
C  **   Left to right.
        Do j = 2, NPY-1
          Gama_cd_Y(i,j) = factr * K_y(i,j) *
     &    (C_i_j_old(k,i,j-1)-2.*C_i_j_old(k,i,j)+C_i_j_old(k,i,j+1))
        EndDo
C  **   Right boundary.
        Gama_cd_Y(i,NPY) = factr * K_y(i,NPY) *
     &      ( (C_i_j_old(k,i,NPY+1)-C_i_j_old(k,i,NPY))*BC_Y_right(k,i) +
     &        (C_i_j_old(k,i,NPY-1)-C_i_j_old(k,i,NPY)) )
      EndDo

C  ** Determine vertical diffusion source/sink terms.
      Do j = 1, NPY
C  **   Bottom boundary.
        Gama_cd_Z(1,j) = dY_i(k,1)*K_z(1,j)/dZ_i_bar(k,2)*
     &       (C_i_j_old(k,2,j)-C_i_j_old(k,1,j)) - dY_i(k,0)
     &       *K_z(0,j)/dZ_i_bar(k,1)*(C_i_j_old(k,1,j)
     &       - C_i_j_old(k,0,j)) * BC_Z_bot(k,j)
C  **   Bottom to top boundary.
        Do i = 2, NPZ-1
          Gama_cd_Z(i,j) = dY_i(k,i)*K_z(i,j)/dZ_i_bar(k,i+1)*
     &       (C_i_j_old(k,i+1,j)-C_i_j_old(k,i,j)) - dY_i(k,i-1)
     &       *K_z(i-1,j)/dZ_i_bar(k,i)*(C_i_j_old(k,i,j)
     &       - C_i_j_old(k,i-1,j))
        EndDo
C  **   Top boundary.
          Gama_cd_Z(NPZ,j) = dY_i(k,NPZ)*K_z(NPZ,j)/dZ_i_bar(k,NPZ)*
     &       (C_i_j_old(k,NPZ+1,j)-C_i_j_old(k,NPZ,j)) * BC_Z_top(k,j)
     &       - dY_i(k,NPZ-1)*K_z(NPZ-1,j)/dZ_i_bar(k,NPZ)*
     &       (C_i_j_old(k,NPZ,j) - C_i_j_old(k,NPZ-1,j))
      EndDo

C  ** Write results.
      If (TESTOUT) Then
        If (mod(nti,Nt_PRT).eq.0) Then
          If (Mod(k,NS_PRT) .eq. 0) Then
            Write (13,*) 'Gama_cd_S: [S: k =',k,']'
            Write (13,*) 'Z,    Y: -->'
            Do i = 1, NPZ
             ii = NPZ - i +1
             Write (13,101) ii,(Gama_cd_S(ii,j),j = 1,NPY)
            EndDo
            Write (13,*) 'Gama_cd_Y: [S: k =',k,']'
            Write (13,*) 'Z,    Y: -->'
            Do i = 1, NPZ
             ii = NPZ - i +1
             Write (13,101) ii,(Gama_cd_Y(ii,j),j = 1,NPY)
```

```
                EndDo
                Write (13,*) 'Gama_cd_Z: [S: k =',k,']'
                Write (13,*) 'Z,    Y: -->'
                Do i = 1, NPZ
                 ii = NPZ - i +1
                 Write (13,101) ii,(Gama_cd_Z(ii,j),j = 1,NPY)
                EndDo
              EndIf
            EndIf
          EndIf

          Return
101       Format (i4,11(1x,E9.3))
          End
```

```
      Subroutine Close_Data
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian               VALDRIFT 1.0
C
C     This subroutine closes all files.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      Do ifile = 13, 15
        Close(ifile)
      EndDo
      Do n = 1, NPM
        LUW(n) = LUWB + n-1
        Close(LUW(n))
      EndDo

      Return
      End
```

```
      Subroutine Courant (*)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995              KJ Allwine and X Bian               VALDRIFT 1.0
C
C     This subroutine determines the time step for the explicit
C     solution method using the Courant criteria for numerical
C     stability.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"

C  ** Set range of acceptable time step.
      PRT_t_s = PRT_t*60.
      dt_max = PRT_t_s                         !Printing Frequency

C  ** Determine time step from Courant criteria for advection
C  ** (Courant # < 0.6).
      Crnt_A = 0.6
      dt_S = dt_max
      dt_Y = dt_max
      dt_Z = dt_max
      If (Umax .gt. 0.) dt_S = Crnt_A*dSmin/Umax
      If (Vmax .gt. 0.) dt_Y = Crnt_A*dYmin/Vmax
      If (Wmax .gt. 0.) dt_Z = Crnt_A*dZmin/Wmax
      dt_A = Min (dt_S, dt_Y, dt_Z)

C  ** Determine time step from Courant criteria for diffusion
C  ** (Courant # < 0.1).
      Crnt_D = 0.1
      dt_S = dt_max
      dt_Y = dt_max
      dt_Z = dt_max
      If (KSmax .gt. 0.) dt_S = Crnt_D*dSmin*dSmin/KSmax
      If (KYmax .gt. 0.) dt_Y = Crnt_D*dYmin*dYmin/KYmax
      If (KZmax .gt. 0.) dt_Z = Crnt_D*dZmin*dZmin/KZmax
      dt_D = Min (dt_S, dt_Y, dt_Z)

C  ** Time step is minimum of advection and diffusion criteria.
      dt = Min (dt_A, dt_D, PRT_t_s)
      dt = PRT_t_s/Real(NINT((PRT_t_s/dt)+0.4999)) !PRT_t multiple of dt

C  ** Determine number of time steps in simulation.
      NPC = 60.*(clk_e-clk_b)/dt
      Nt_PRT = NINT(PRT_t_s/dt)
      Nstps = NPS*NPC

C  ** Write out results.
      Write (13,'(//A)') '+++++RESULTS FROM SUBROUTINE COURANT+++++'
      Write (13,100) dt, NPC, Nt_PRT, Nstps, dt_A, dt_D

      If (Nstps .gt. Nstps_max) Then
        Write (13,*) '  The number of time steps times the number of',
     &       ' space steps is',Nstps, '.'
        Write (13,*) '  This will take more than 1/2 h to run.  You',
     &        ' may want to reconsider the configuration.'
        Write (13,*) '  If want to run, resubmit the run with',
     &        ' Nstps_max greater than',Nstps,'  .'
        Return 1
      EndIf

      Return
  100 Format (/,10X,'DT =',F6.1,5X,'NPC =',I4,6X,'NT_PRT =',I3,4x,
     &  'NSTPS =',I6,2X,'DT_A =',F6.1,3X,'DT_D =',F6.1)
      End
```

```
      Subroutine Cp_Source_Sink (k, nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine sets the dry deposition source/sinks (Gama_cp).
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Set the dry deposition for all points on cross-section to zero.
      Do i = 1, NPZ
        Do j = 1, NPY
          Gama_cp(i,j) = 0.
        EndDo
      EndDo

C  ** Determine the dry deposition at ground-level on cross-section.
      If (Vd .ne. 0.) Then
        Do iG = 1, N_Gnd
          DDinst = Vd * C_i_j_old(k,i_Gnd(iG),j_Gnd(iG))
          D_Gnd(k,iG) = D_Gnd(k,iG) + DDinst*dt
          Gama_cp(i_Gnd(iG),j_Gnd(iG)) = - DDinst * D_Sfc(k,iG)
        EndDo
      EndIf

      Return
      End
```

```
      Subroutine Cs_Source_Sink (k, nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian            VALDRIFT 1.0
C
C     This subroutine sets the release rate term (Gama_cs).
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Set the release rate for all points on cross-section to zero.
      Do i = 1, NPZ
        Do j = 1, NPY
          Gama_cs(i,j) = 0.
        EndDo
      EndDo

C  ** Set the actual release rate for all sources on cross-section.
      Do n = 1, N_Src
        If (clk.ge.clk_b_Src(n) .and.
     &      clk.le.clk_e_Src(n). and. k_Src(n).eq.k ) Then
          Gama_cs(i_Src(n),j_Src(n)) = Rate_Src(n)
        EndIf
      EndDo

      Return
      End
```

```
      Subroutine Descent
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995              KJ Allwine and X Bian            VALDRIFT 1.0
C
C     This subroutine calculates the heights of the inversion top
C     and CBL top as a function of time after sunrise as valley
C     inversions are destroyed, using Whiteman's (1980)
C     numerical method.
C
C**** DEFINITIONS ***************************************************
C     BETA_R      RATE OF TEMP CHANGE AT INVERSION TOP (K/SEC)        *
C     dt          TIME STEP (SEC)                                     *
C     GAMMA       VERT POT TEMPERATURE GRADIENT IN INVERSION (K/M)    *
C     TOPCBL      HEIGHT OF CBL TOP (M)                               *
C     TOPSBL      HEIGHT OF INVERSION TOP (M)                         *
C     HZERO       INITIAL HEIGHT OF INVERSION TOP (M)                 *
C     RKF         FRACTION (0-1) OF SENS HT FLX GOING INTO CBL GROWTH*
C     PI          TRIGONOMETRIC CONSTANT                              *
C     TAU         LENGTH OF DAYLIGHT PERIOD (SEC)                     *
C     EL_T        ELAPSED TIME SINCE SUNRISE (SEC)                    *
C     WDTH        VALLEY FLOOR WIDTH (M)                              *
C     CSHP        VALLEY CROSS-SECTION SHAPE FACTOR                   *
C     FACT1       FACTOR USED IN CALCULATIONS                         *
C     THTOT       POTENTIAL TEMPERATURE                               *
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Define local variables.
      HZERO = zmean(NPZ)
      WDTH = wdth_mean

C  ** Stop calculations when inversion top sinks below CBL top.
      If (TOPSBL .LE. TOPCBL) Return

C  ** Do calculations.
      EL_T = EL_T + dt
      FACT2 = FACT1*SIN(PI*(EL_T)/TAU)
      DHDT = 0.
      If (RKF.NE.0.) DHDT = FACT2*THTOT*RKF*(WDTH+TOPCBL*CSHP)/
     &                     (GAMMA*TOPCBL*(WDTH+(TOPCBL*CSHP)/2.))
      FNUM = (WDTH+TOPSBL*CSHP-RKF*(WDTH+TOPCBL*CSHP))*FACT2-((BETA_R*
     &        (HZERO-TOPSBL)*(WDTH+((HZERO+TOPSBL)/2.)*CSHP))/(2.*THTOT))
              FDENOM=TOPSBL*GAMMA*(WDTH+(TOPSBL*CSHP)/2.)
     &        +(BETA_R*(EL_T)*(WDTH+TOPSBL*CSHP))/2.
      DHTDT = -THTOT*FNUM/FDENOM
      DHT = DHTDT*dt
      TOPSBL = TOPSBL + DHT
      DHC = DHDT*dt
      TOPCBL = TOPCBL + DHC

      Return
      End
```

```
      Subroutine Diffusivities
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian            VALDRIFT 1.0
C
C     This subroutine uses boundary layer parameterizations to calculate
C     eddy diffusivities for stable, neutral and unstable conditions.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"
      Real*4    L_sta, L_uns

      Ak = 0.4              !  Von Karmen Constant
      C_sta = 5.0           !  Coefficient in stabl B_D formula
      C_uns = 16.0          !  Coefficient in unstabl B_D formula

C **  Set the surface friction velocities U*.
      U_star_sta =.1        !  (m/s)  For stable condition
      U_star_neu =.2        !  (m/s)  For neutral condition
      U_star_uns =.4        !  (m/s)  For unstable condition

C **  Set the M-O scale length, L = (U*^2 T)/(Ak g T*).
      L_sta = 30.0          !  (m)   For stable condition
      L_uns = -30.0         !  (m)   For unstable condition

C **  Set the mean PBL heights.
      H_pbl_sta = 400.0   !  (m)  For stable condition
      H_pbl_neu = 600.0   !  (m)  For neutral condition
      H_pbl_uns = 1400.0  !  (m)  For unstable condition

C **  Set the mean surface layer heights.
      h_s_sta = 10.0        !  (m)  For stable condition
      h_s_neu = 20.0        !  (m)  For neutral condition
      h_s_uns = 30.0        !  (m)  For unstable condition

C **  Set the half of mean depth of drainage flow.
      H_limit = H_pbl_sta - 200.
      D_half = zmean(NPZ)/4.
      If ( D_half .ge. H_limit) D_half = H_limit

C **  Calculate the mean flux-decrease rate with Z above the Constant
C **  Fluxes Layer [See Bian et al, 1992, Equation(10)].
      Flux_dec_r_sta = (1. - D_half / H_pbl_sta)
      Flux_dec_r_neu = (1. - D_half / H_pbl_neu)
      Flux_dec_r_uns = (1. - D_half / H_pbl_uns)

C **  Calculate mean dimensionless values of B-D flux-profile formula.
      Zs_sta = D_half
      Zs_neu = D_half
      Zs_uns = D_half
      If ( Zs_sta .ge. h_s_sta) Zs_sta = h_s_sta
      If ( Zs_neu .ge. h_s_neu) Zs_neu = h_s_neu
      If ( Zs_uns .ge. h_s_uns) Zs_uns = h_s_uns

      F_sta = (1.0 + C_sta * Zs_sta/L_sta)            ! For stable
      F_neu = 1.0                                     ! For neutral
      F_uns = 1./(1.0 - C_uns * Zs_uns/L_uns)**.25    ! For unstable

C **  Calculate the mean Z-dir eddy diffusivity (m^2/s).
      Z_sta = (D_half + h_s_sta)/2.5
      Z_neu = (D_half + h_s_neu)/6.0
      Z_uns = (D_half + h_s_uns)/14.5
      K_z_stable   = Ak*U_star_sta*Z_sta*Flux_dec_r_sta/F_sta
      K_z_neutral  = Ak*U_star_neu*Z_neu*Flux_dec_r_neu/F_neu
      K_z_unstable = Ak*U_star_uns*Z_uns*Flux_dec_r_uns/F_uns
      write(13,*) ' '
      write(13,*) ' +++++ RESULTS FROM SUBROUTINE DIFFUSIV +++++ '
      write(13,*) ' '
      write(13,*) '        K_z_stable   = ',K_z_stable,' (m2/s)'
      write(13,*) '        K_z_neutral  = ',K_z_neutral,' (m2/s)'
      write(13,*) '        K_z_unstable = ',K_z_unstable,' (m2/s)'
      write(13,*) ' '

C **  Set/input/Read Sigma_v and Sigma_w.
      Sigma_w_sta = 0.95*U_star_sta         ! (m/s)
      Sigma_w_neu = 1.20*U_star_neu
      Sigma_w_uns = 1.98*U_star_uns
      Sigma_v_sta = 2.29*U_star_sta
      Sigma_v_neu = 2.00*U_star_neu
      Sigma_v_uns = 2.28*U_star_uns

C **  Calculate the mean Y-dir eddy diffusivity (m^2/s).
```

```
      K_y_stable   = (Sigma_v_sta/Sigma_w_sta)**2*K_z_stable
      K_y_neutral  = (Sigma_v_neu/Sigma_w_neu)**2*K_z_neutral
      K_y_unstable = (Sigma_v_uns/Sigma_w_uns)**2*K_z_unstable
      write(13,*) '         K_y_stable   = ',K_y_stable ,' (m2/s)'
      write(13,*) '         K_y_neutral  = ',K_y_neutral,' (m2/s)'
      write(13,*) '         K_y_unstable = ',K_y_unstable,' (m2/s)'
      write(13,*) '  '

C **  Calculate the mean S-dir eddy diffusivity (m^2/s).
      K_s_unstable = 0.
      K_s_neutral  = 0.
      K_s_stable   = 0.
      write(13,*) '         K_s_stable   = ',K_s_stable  ,' (m2/s)'
      write(13,*) '         K_s_neutral  = ',K_s_neutral ,' (m2/s)'
      write(13,*) '         K_s_unstable = ',K_s_unstable,' (m2/s)'
      write(13,*) '  '

      Return
      End
```

```
      Subroutine Ground_Loc
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995              KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine determines the indicies (i,j) of the grid
C     points on the sidewalls and the floor of each cross-section.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      N_Gnd = 0
C  ** Down left sidewall.
      Do i = NPZ, 1, -1
        N_Gnd = N_Gnd + 1
        i_Gnd(N_Gnd) = i
        j_Gnd(N_Gnd) = 1
      EndDo
C  ** Across valley floor.
      Do j = 2, NPY-1
        N_Gnd = N_Gnd + 1
        i_Gnd(N_Gnd) = 1
        j_Gnd(N_Gnd) = j
      EndDo
C  ** Up right sidewall.
      Do i = 1, NPZ
        N_Gnd = N_Gnd + 1
        i_Gnd(N_Gnd) = i
        j_Gnd(N_Gnd) = NPY
      EndDo

      Return
      End
```

```
      Subroutine Initial
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine sets the initial conditions.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Determine the maximum values of diffusivities.
      KSmax =  MAX (K_s_unstable,K_s_neutral,K_s_stable)
      KYmax =  MAX (K_y_unstable,K_y_neutral,K_y_stable)
      KZmax =  MAX (K_z_unstable,K_z_neutral,K_z_stable)

C  ** Set the initial conditions for Cij's.
      Do  k = 1, NPS
        Do i = 1, NPZ
          Do j = 1, NPY
            C_i_j(k,i,j) = C_Bkgnd
            C_i_j_old(k,i,j) = C_i_j(k,i,j)
          EndDo
        EndDo
      EndDo

C  ** Set constant Dirichlet BC conditions for Cij's at all boundaries.
      Do  k = 0, NPS+1
C  **   Valley floor.
        Do j = 0, NPY+1
          C_i_j(k,0,j) = C_Bkgnd
          C_i_j_old(k,0,j) = C_i_j(k,0,j)
        EndDo
C  **   Top of valley atmosphere.
        Do j = 0, NPY+1
          C_i_j(k,NPZ+1,j) = C_Bkgnd
          C_i_j_old(k,NPZ+1,j) = C_i_j(k,NPZ+1,j)
        EndDo
C  **   Left sidewall.
        Do i = 0, NPZ+1
          C_i_j(k,i,0) = C_Bkgnd
          C_i_j_old(k,i,0) = C_i_j(k,i,0)
        EndDo
C  **   Right sidewall.
        Do i = 0, NPZ+1
          C_i_j(k,i,NPY+1) = C_Bkgnd
          C_i_j_old(k,i,NPY+1) = C_i_j(k,i,NPY+1)
        EndDo
      EndDo
C  ** S=0 plane.
      Do i = 0, NPZ+1
        Do j = 0, NPY+1
          C_i_j(0,i,j) = C_Bkgnd
          C_i_j_old(0,i,j) = C_i_j(0,i,j)
        EndDo
      EndDo
C  ** S=NPS+1 plane.
      Do i = 0, NPZ+1
        Do j = 0, NPY+1
          C_i_j(NPS+1,i,j) = C_Bkgnd
          C_i_j_old(NPS+1,i,j) = C_i_j(NPS+1,i,j)
        EndDo
      EndDo

C  ** Set the permeability of the lateral and vertical boundaries
C  ** to diffusion (0 - impermeable, 1 - permeable).
      Do k = 1, NPS
        Do i = 1, NPZ
          BC_Y_left(k,i) = 0.
          BC_Y_right(k,i) = 0.
        EndDo
        Do j = 1, NPY
          BC_Z_bot(k,j) = 0.
          BC_Z_top(k,j) = Dif_Coef
        EndDo
      EndDo

C  ** Determine the mean valley characteristics used for valley
C  ** inversion destruction.
      zmean(0) = 0.
      Do i = 1, NPZ
        zz = 0.
        Do k = 0, NPS
```

```
            zz = zz + Z_i(k,i)
          EndDo
          zmean(i) = zz/Real(NPS+1)
        EndDo
        CSHP = 0.
        wdth_mean = 0.
        Do k = 0, NPS
          wdth_mean = wdth_mean + l_width(k)
          CSHP = CSHP + theta(k)
        EndDo
        wdth_mean = wdth_mean/Real(NPS+1)
        CSHP = CSHP/Real(NPS+1)
        TOPCBL = 1.
        TOPSBL = zmean(NPZ)
        EL_T = 0.
        CP = 1005.
        FACT1 = A0*A1/(RHO*CP)
        THTOT = (1000./PRESS)**.286

C   ** Initialize mass balance variables.
        TOT_RELE = 0.
        FLUX_NET = 0.

C   ** Determine indicies of ground-level cells.
        Call Ground_Loc
        Write (14) NPS, NPY, NPZ, clk_b, clk_e, PRT_t, dS
        Write (14) dY_i, Z_i, dZ_i
        Write (14) N_Gnd, i_Gnd, j_Gnd
        Do  k = 1, NPS
          Do iG = 1, N_Gnd
            D_Gnd(k,iG) = 0.0
          EndDo
        EndDo

C   ** Determine quantities used to calculate the volume flow.
        Call Integral

C   ** Determine Prandtl wind profile weighting factor for
C   ** estimating peak winds on vertical profile.
        Do n = 1, NPM
          Met_Sta_Idx(n) = NINT(Real(NPS)*S_Met(n)/S_Ter(NPT))
          fac = HTagl(n)/Z_i(Met_Sta_Idx(n),NPZ)
          facMet(n) = 3.2*SIN(pi*fac)/EXP(3.3*fac)
        EndDo

C **    Determine surface friction velocity and dry deposition Vd
         Call Surface_fluxes
C **    Determine difusivities
         Call Diffusivities

C   ** Write out results.
        If (DETAIL) Then
          Write (13,'(//A)') '+++++RESULTS FROM SUBROUTINE INITIAL+++++'
          Write (13,'(/A/)') '   ** Zmean **'
          Write (13,'(A,12(4x,I2,4x))') 'Z --> ',(i,i=1,NPZ)
          Write (13,'(6x,12E10.3)') (zmean(i),i=1,NPZ)
          Write(13,'(/2(5x,A,E10.3))') '  Wdth_Mean=', wdth_mean,
     &                                 '  CSHP=', CSHP
          Write (13,'(/A/)') '   ** Integral **'
          Write (13,'(A,12(4x,I2,4x))') 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,'(I6,12E10.3)') k,(integ(k,i),i=1,NPZ)
          EndDo
        EndIf

        Return
        End
```

```
      Subroutine Integral
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine computes the integral over gamma associated
C     with the along-valley volume flow rate in flowtube ij for all
C     cross sections.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      qq2 = 3.3*3.3
      pi2 = pi*pi
      cc0 = qq2 + pi2
      cc1 = - 3.2/cc0
      cf1 = cc1*3.3
      cf2 = cc1*pi
      cc3 = cc1*(qq2-pi2)/cc0
      cf4 = cc1*2.*3.3*pi/cc0
      Do k = 0, NPS
        aa = theta(k)*dZ_i(k,NPZ)
        arg0 = cf2*l_width(k) + cf4*aa
        Do i = 1, NPZ
          cc2 = pi*gama(k,i)
          cc3 = aa*gama(k,i) + l_width(k)
          arg1 = Exp(-3.3*gama(k,i)) *
     &         ( (cc3*cf1+aa*cf3)*Sin(cc2) + (cc3*cf2+aa*cf4)*Cos(cc2) )
          integ(k,i) = arg1 - arg0
          arg0 = arg1
        EndDo
      EndDo

      Return
      End
```

```
      Subroutine Julian_Day (IYR, MO, IDA, JULDAY)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine computes the Julian day given the month, day and
C     year.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Integer NDAY(12)
      Data NDAY /0,31,59,90;120,151,181,212,243,273,304,334/

      JULDAY = IDA + NDAY(MO)
C  ** Adjust for leap year.
      If ( (MOD(IYR,4).eq.0) .and. (MO.ge.3) )  JULDAY = JULDAY + 1

      Return
      End
```

```
      Subroutine LinInt (N, X, Y, Xv, Yv, *)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine determines a value Yv, at some coordinate Xv,
C     given an array of coordinates X and corresponding values Y.  The
C     value at the desired coordinate is determined by linear
C     interpolation.  If the coordinate Xv is outside the limits of
C     the array X, the value Yv is set to the respective endpoint value.
C
C     The inputs are:
C        N  = number of inputs points (>1).
C        X  = array of input point coordinates.
C        Y  = array of input point values.
C        Xv = coordinate where value needed.
C
C     The outputs are:
C        Yv = value at desired coordinate.
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Real*4 X(1), Y(1)

      If (N.le.1) Then
        Write (13,*) '  Not enough input points for subroutine LinInt'
        Return 1
      EndIf
C  ** Endpoint checks.
      If (Xv .le. X(1)) Then
        Yv = Y(1)
        Return
      EndIf
      If (Xv .gt. X(N)) Then
        Yv = Y(N)
        Return
      EndIf
C  ** Linear interpolation.
      Do J = 2, N
        If (Xv .le. X(J)) Then
          Slp = (Y(J) - Y(J-1)) / (X(J) - X(J-1))
          Cpt = Y(J) - Slp*X(j)
          Yv = Slp*Xv + Cpt
          Return
        EndIf
      EndDo

      End
```

```
      Subroutine Line_Loc (S_Sb, Y_Sb, Z_Sb, S_Se, Y_Se, Z_Se, *)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine determines the indicies (k,i,j) of all the grid
C     point on a line source given the (S,Y,Z) coordinates of the
C     beginning and ending points of the line source.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C **  Determine number of grid points on this line.
      Call Point_Loc (S_Sb, Y_Sb, Z_Sb, k_c_b, j_c_b, i_c_b)
      Call Point_Loc (S_Se, Y_Se, Z_Se, k_c_e, j_c_e, i_c_e)
      Ns_c = 1 + k_c_e - k_c_b
      Ny_c = 1 + j_c_e - j_c_b
      Nz_c = 1 + i_c_e - i_c_b
      Npts = Max (Ns_c, Ny_c, Nz_c)
      If (Npts .gt. MPR) Then
        Write (13,*) '  Too many grid points in line source'
        Return 1
      EndIf
      RNpts = Real(Npts)
      dNs_c = Real(Ns_c)/RNpts
      dNy_c = Real(Ny_c)/RNpts
      dNz_c = Real(Nz_c)/RNpts
      Do ip = 1, Npts
        Rip = Real(ip-1)
        k_c(ip) = k_c_b + Int(Rip*dNs_c)
        i_c(ip) = i_c_b + Int(Rip*dNz_c)
        j_c(ip) = j_c_b + Int(Rip*dNy_c)
      Enddo

      Return
      End
```

```
      Subroutine Make_Grid (*)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine takes the terrain input data and outputs valley
C     characteristics.  The valley coordinate system is (S,Z,Y) where
C     S is along the valley with the origin at the upvalley end of the
C     domain, Y is across the valley from left to right looking up the
C     valley where the origin is at the center of the valley, and Z is
C     vertical above the valley floor with the origin at the floor.
C
C     The input terrain characteristics at NPS cross-sections are the
C     elevation of the ridge, z_r, and valley floor, z_f, the sidewall
C     angles, alpha_l and alpha_r, and the width of the valley floor,
C     l_width.  The most up-valley cross-section (at S=0) must be given
C     as a minimum.
C
C     The outputs are:  Z_i(S,Z), dZ_i(S,Z), A_i(S,Z), A_i_j(S,Z),
C     dZ_i_bar(S,Z), dY_i(S,Z), dY_i_bar(S,Z), gama(S,Z), beta(Y), and
C     Vdot_Fac(Y)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Define constants.
      RNPS = Real(NPS)
      RNPZ = Real(NPZ)
      RNPY = Real(NPY)
      DtoR = pi/180.

C  ** Fill-up the S array with computed S values.
      S(0) = S_Ter(1)
      Do k = 1, NPS
        S(k) = S(k-1) + dS
      EndDo

C  ** Interpolate z_r, z_f, alpha_l, alpha_r and l_width in S.
      Do k = 0, NPS
        Call LinInt (NPT, S_Ter, z_r_Ter, S(k), z_r(k), *998)
        Call LinInt (NPT, S_Ter, z_f_Ter, S(k), z_f(k), *998)
        Call LinInt (NPT, S_Ter, alpha_l_Ter, S(k), alpha_l(k), *998)
        Call LinInt (NPT, S_Ter, alpha_r_Ter, S(k), alpha_r(k), *998)
        Call LinInt (NPT, S_Ter, l_width_Ter, S(k), l_width(k), *998)
      EndDo

C  ** Initialize cross section A(S0).
      Do k = 0, NPS
        Th_l = 0.
        Th_r = 0.
        If (alpha_l(k).lt.90.) Th_l = 1./Tan(DtoR*alpha_l(k))
        If (alpha_r(k).lt.90.) Th_r = 1./Tan(DtoR*alpha_r(k))
        theta(k) = Th_l + Th_r
      EndDo
      Z_i(0,0) = 0.
      A_Cum = 0.
      cc1 = (z_r(0)-z_f(0))/RNPZ
      Do i = 1, NPZ
        dZ_i(0,i) = cc1
        Z_i(0,i) = Z_i(0,i-1) + dZ_i(0,i)
        A_i(0,i) = Z_i(0,i)*(l_width(0)+theta(0)*Z_i(0,i)/2.) - A_cum
        A_cum = A_cum + A_i(0,i)
      EndDo

C  ** Initialize all cross sections A(S).
      A_0 = A_cum
      Do k = 0, NPS
        Z_i(k,NPZ) = z_r(k)-z_f(k)
        A_S = Z_i(k,NPZ)*(l_width(k) + theta(k)/2. * Z_i(k,NPZ))
        cc2 = A_S/A_0
        Do i = 1, NPZ
          A_i(k,i) = A_i(0,i)*cc2
          A_i_j(k,i) = A_i(k,i)/RNPY
        EndDo
      EndDo

C  ** Compute Z_i(S) [Eq10 p169], dZ_i(S,i), dZ_i_bar(S,i)
      dZmin = 10000.
      Do k = 0, NPS
        Z_i(k,0) = 0.
        Do i = 1, NPZ
          If (theta(k) .eq. 0.) Then          ! Vertical Walls
```

```
              dZ_i(k,i) = A_i(k,i)/l_width(k)
              Z_i(k,i) = Z_i(k,i-1) + dZ_i(k,i)
            Else
              Arg = Z_i(k,i-1)*(l_width(k)+theta(k)*Z_i(k,i-1)/2.)
     &                                                  + A_i(k,i)
              Z_i(k,i) = (sqrt(l_width(k)*l_width(k) +
     &                    2.*theta(k)*Arg) - l_width(k))/theta(k)
              dZ_i(k,i) = Z_i(k,i) - Z_i(k,i-1)
            EndIf
            If (dZ_i(k,i) .lt. dZmin) dZmin = dZ_i(k,i)
            ii = i-1
            If (i .eq. 1) ii = i
            dZ_i_bar(k,i) = (dZ_i(k,i) + dZ_i(k,ii))/2.
          EndDo
        EndDo

C    ** Compute dY_i(S) and dY_i_bar.
        dYmin = 10000.
        Do k = 0, NPS
          dY_i(k,0) = l_width(k)/RNPY
          If (dY_i(k,0) .lt. dYmin) dYmin = dY_i(k,0)
          dtheta = theta(k)/RNPY
          Do i = 1, NPZ
            dY_i(k,i) = dY_i(k,0) + dtheta*Z_i(k,i)
            dY_i_bar(k,i) = (dY_i(k,i) + dY_i(k,i-1))/2.
          EndDo
        EndDo

C    ** Compute deposition surfaces.
        Do k = 1, NPS
          iG = 0
C    **   Down left sidewall.
          fac_l = Sin(DtoR*alpha_l(k))
          Do i = NPZ, 2, -1
            iG = iG + 1
            D_Sfc(k,iG) = dZ_i(k,i)/fac_l
          EndDo
C    **   Lower left corner.
          iG = iG + 1
          D_Sfc(k,iG) = dZ_i(k,1)/fac_l + dY_i(k,0)
C    **   Across valley floor.
          Do j = 2, NPY-1
            iG = iG + 1
            D_Sfc(k,iG) = dY_i(k,0)
          EndDo
C    **   Lower right corner.
          fac_r = Sin(DtoR*alpha_r(k))
          iG = iG + 1
          D_Sfc(k,iG) = dZ_i(k,1)/fac_r + dY_i(k,0)
C   ***   Up right sidewall.
          Do i = 2, NPZ
            iG = iG + 1
            D_Sfc(k,iG) = dZ_i(k,i)/fac_r
          EndDo
        EndDo

C    ** Compute beta and gama coordinates.
        beta(0) = -0.5
        Do j = 1, NPY
          beta(j) = Real(j)/RNPY - 0.5
        EndDo
        Do k = 0, NPS
          Do i = 0, NPZ
            gama(k,i) = Z_i(k,i)/(z_r(k) - z_f(k))
          EndDo
        EndDo

C    ** Compute beta function for vol. flow calculation [Eq71 p188].
        cc1 = 0.95/RNPY
        Do j = 1, NPY
          Vdot_Fac(j) = cc1 - 1.13*(beta(j)*beta(j)*beta(j)
     &                        - beta(j-1)*beta(j-1)*beta(j-1))
        EndDo

C    ** Write out results.
        If (DETAIL) Then
          Write (13,'(//A)') '+++++RESULTS FROM SUBROUTINE MAKE_GRID+++++'
          Write (13,100) '  ** S (M) **'
          Write (13,101) 'k --> ',(k,k=0,NPS,NS_PRT)
          Write (13,103) (S(k),k=0,NPS,NS_PRT)
```

```
          Write (13,100) '  ** A_i_j (M^2) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(A_i_j(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** Z_i (M) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(Z_i(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** dZ_i (M) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(dZ_i(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** dZ_i_bar (M) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(dZ_i_bar(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** dY_i (M) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(dY_i(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** dY_i_bar (M) **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(dY_i_bar(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** gamma **'
          Write (13,101) 'Z --> ',(i,i=1,NPZ)
          Do k = 0, NPS, NS_PRT
            Write (13,102) k,(gama(k,i),i=1,NPZ)
          EndDo
          Write (13,100) '  ** beta **'
          Write (13,101) 'Y --> ',(j,j=1,NPY)
          Write (13,103) (beta(j),j=1,NPY)
        EndIf

  998 Return
  100 Format (/A/)
  101 Format (A,11(4x,I2,4x))
  102 Format (I6,11E10.3)
  103 Format (6X,11E10.3)
      End
```

```
      Subroutine Mass_Balance (nti)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine determines the pollutant mass balance over the
C     computational domain for a time step.
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Check mass in domain and release to domain for this time.
      TOT_R = TOT_R*dt*dS
      TOT_C = 0.
      TOT_dep =0.
      Do k = 1, NPS
        Do i = 1, NPZ
          cc1 = A_i_j(k,i)*dS
          Do j = 1, NPY
            TOT_C = TOT_C + C_i_j(k,i,j)*cc1
          EndDo
        EndDo

C ** Calculate total deposition mass.
        Do iG = 1, N_Gnd
          TOT_dep = TOT_dep + D_Gnd(k,iG) * D_Sfc(k,iG) *dS
        EndDo
      EndDo

      TOT_RELE = TOT_RELE + TOT_R
      TOT_MASS = TOT_C + TOT_dep - FLUX_NET  !flux_net is previous time
      residual = TOT_RELE - TOT_MASS
      ratio = 1.
      If (TOT_RELE .ne. 0.) ratio = TOT_MASS/TOT_RELE

C  ** Write out mass balance.
      If (mod(nti,Nt_PRT) .eq. 0) Then
        Write (13,99)
        Write (13,100) nti, clk, IHr_clk, RMn_clk
        Write (13,101)
        Write (13,102) TOT_RELE, TOT_MASS, residual, ratio, TOT_C,
     &                 TOT_dep, FLUX_NET
      EndIf

C  ** Compute net mass advective flux in S-dir.
      SumA0 = 0.
      SumA1 = 0.
      Do i = 1, NPZ
        Do j = 1, NPY
          If (NEUMANN) Then
            adv0 =  C_i_j(1,i,j)*Vdot_S(0,i,j)*dt
            adv1 =  C_i_j(NPS,i,j)*Vdot_S(NPS,i,j)*dt
          Else
            k0 = 0
            If (Vdot_S(0,i,j) .lt. 0.) k0 = 1
            adv0 =  C_i_j(k0,i,j)*Vdot_S(0,i,j)*dt
            k1 = NPS
            If (Vdot_S(NPS,i,j) .lt. 0.) k1 = NPS+1
            adv1 =  C_i_j(k1,i,j)*Vdot_S(NPS,i,j)*dt
          EndIf
          SumA0 = SumA0 + adv0
          SumA1 = SumA1 + adv1
        EndDo
      EndDo
      S_adv = SumA0 - SumA1
      FLUX_NET = FLUX_NET + S_adv

      Return
   99 Format (//'+++++RESULTS OF SUBROUTINE MASS_BALANCE+++++')
  100 Format (/'  AFTER ',I4,' TIMESTEPS     CLK = ',F7.1,
     &        '                HOUR =',I3,'  MINUTE =',F5.1)
  101 Format (/'        TOT_RELE    TOT_MASS     residual      ratio ',
     &        '          TOT_C    TOT_dep     FLUX_NET')
  102 Format (2x,3(2X,E10.4),F10.5,5x,4(2X,E10.4)/)
      End
```

```
      Subroutine Point_Loc (S_c, Y_c, Z_c, KK, JJ, II)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995              KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine determines the indicies (k,i,j) of a point given
C     the (S,Y,Z) coordinates.
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      Do k = 0, NPS-1
        If ( S_c.ge.S(k) .and. S_c.lt.(S(k+1)) ) Then
          KK = k + 1
          GoTo 10
        EndIf
      EndDo
      KK = NPS
   10 Continue

      Do i = 0, NPZ-1
        If ( Z_c.ge.Z_i(KK,i) .and. Z_c.lt.Z_i(KK,i+1) ) Then
          II = i + 1
          GoTo 20
        EndIf
      EndDo
      II = NPZ
   20 Continue

      Do j = 0, NPY-1
        Half_Y = 0.5*dY_i(KK,II)*Real(NPY)
        YYmin = Real(j)*dY_i(KK,II) - Half_Y
        YYmax = Real(j+1)*dY_i(KK,II) - Half_Y
        If ( Y_c.ge.YYmin .and. Y_c.lt.YYmax ) Then
          JJ = j + 1
          GoTo 30
        EndIf
      Enddo
      JJ = NPY

   30 Return
      End
```

```
      Subroutine Print_C
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine writes concentration and deposition fields.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Write concentration and deposition arrays to .BIN file.
      Write (14) clk, TOPCBL, TOPSBL,
     &           (((C_i_j(k,i,j),k=1,NPS),i=1,NPZ),j=1,NPY),
     &           ((D_Gnd(k,iG),k=1,NPS),iG=1,N_Gnd)

      If (DATDMP) Then
C  **   Write tables to .DAT file.
        ac = 1.E-30
        Write (15,*) '  '
        Write (15,100) 'clk:',clk,'    HOUR = ',IHr_clk,'  MINUTE =',
     &             RMn_clk,'  TOPCBL=',TOPCBL,'  TOPSBL=',TOPSBL
        Write (15,*) '  '
        Do k = 0, NPS, NS_PRT
          Write (15,*) 'CROSS_VIEW AT: [S: k =',k,']'
          Write (15,*) 'Z,    Y: -->'
          Do i = 1, NPZ
            ii = NPZ - i + 1
            If (LOGPRT) Then
              Write (15,102) ii,(LOG10(C_i_j(k,ii,j)+ac),j = 1,NPY)
            Else
              Write (15,101) ii,(C_i_j(k,ii,j),j = 1,NPY)
            EndIf
          EndDo
          Write (15,*) '^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^'
        EndDo
        Write (15,*) '-----------------------------------'
        Write (15,*) 'TOP_VIEW AT: [Z: i = NZcs =',i_Src(1),']'
        Write (15,*) 'S,    Y: -->'
        Do k = 1, NPS
          If (LOGPRT) Then
            Write (15,102) k,(LOG10(C_i_j(k,i_Src(1),j)+ac),j = 1,NPY)
          Else
            Write (15,101) k,(C_i_j(k,i_Src(1),j),j = 1,NPY)
          EndIf
        EndDo
        Write (15,*) '-----------------------------------'
        Write (15,*) 'LATERAL_VIEW AT: [Y: j = NYcs =',j_Src(1),']'
        Write (15,*) 'S,    Z: -->'
        Do k = 1, NPS
          If (LOGPRT) Then
            Write (15,102) k,(LOG10(C_i_j(k,i,j_Src(1))+ac),i = 1,NPZ)
          Else
            Write (15,101) k,(C_i_j(k,i,j_Src(1)),i = 1,NPZ)
          EndIf
        EndDo
        Write (15,*) '==================================='
      EndIf

      Return
100   Format (A,F7.1,A,I2.2,A,F5.1,A,F7.1,A,F7.1)
101   Format (i4,11(1x,E9.3))
102   Format (i4,11(1x,F9.3))
      End
```

```
      Subroutine Read_Data (*)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine opens file VALDRIFT.FIL and reads the names of
C     the four input and two output files.  Files VALDRIFT.RS and
C     VALDRIFT.TER are opened, read and closed.  The wind data file is
C     opened and read to the starting record.  The solar parameters are
C     determined for the day and location.  The output run trace file,
C     VALDIRFT.TRC is opened.and initial run conditions are written.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"

C  ** Define constants.
      pi = 3.141593

C  ** Set DATDMP to .TRUE. if want concentrations written to .DAT file.
      DATDMP = .FALSE.
      LOGPRT = .FALSE.       !Concentrations in powers of 10 to DAT file

C  ** Open file and read in names of input and output files.
      Open (Unit=1, File='VALDRIFT.FIL', Status='OLD', Form='FORMATTED')
      Do L = 1, 6
        Read (1,*) FILEN(L)
      EndDo
      Close (Unit=1)

C  ** Open Output Files - .TRC and .BIN.
      OPEN(Unit=13, File=FILEN(5), Status ='UNKNOWN',
     &                                         FORM='FORMATTED')
      OPEN(Unit=14, File=FILEN(6), Status ='UNKNOWN',
     &                                         FORM='UNFORMATTED')

C  ** Open Output .DAT File.
      If (DATDMP) OPEN(Unit=15, File='VALDRIFT.DAT',
     &                           Status ='UNKNOWN', FORM='FORMATTED')

C  ** Read run specification file.
      Open (Unit=1, File=FILEN(1), Status='OLD', Form='FORMATTED')
      TITLE = '    '
      USER = '    '
      Read (1,*) TITLE, USER, DETAIL, NEUMANN
      Read (1,*) IYR, MO, IDA
      Call Julian_Day (IYR, MO, IDA, JUL)
      Read (1,*) IHRBeg,MNBeg, IHREnd,MNEnd
      clk_b = IHRBeg*60 + MNBeg
      clk_e = IHREnd*60 + MNEnd
      Read (1,*) NPS, NPY, NPZ, Nstps_max
      If (NPS+1 .gt. MPS1) Then
        Write (13,*) '  Too many points in the S-direction'
        Return 1
      EndIf
      If (NPY+1 .gt. MPY1) Then
        Write (13,*) '  Too many points in the Y-direction'
        Return 1
      EndIf
      If (NPZ+1 .gt. MPZ1) Then
        Write (13,*) '  Too many points in the Z-direction'
        Return 1
      EndIf
      Read (1,*) PRT_t, NS_PRT
      Read (1,*) U_day_mean, U_night_mean, Umax
      Vmax = 0.
      Wmax = 0.
      Read (1,*) A0, PRESS, RHO, BETA_R, GAMMA, RKF
      Read (1,*) Dif_Coef, C_Bkgnd

C  ** Note that all met stations must cover same period of record and
C  ** have the same time step.
      NPM = 1 .                    ! Number of met stations
      LUWB = 10                    ! Logical Unit of 1st met station
      Do n = 1, NPM
        LUW(n) = LUWB + n-1
        OPEN (Unit=LUW(n),File=FILEN(4),Status='OLD',Form='FORMATTED')
        Read (LUW(n),*) S_Met(n), Met_Nam(n)
        Read (LUW(n),*) AZdwnvly(n),HTagl(n),dtM,IYRM,MOM,IDAM,IHRM,MNM
        If (IYRM.ne.IYR .or. MOM.ne.MO .or. IDAM.ne.IDA) Then
          Write (13,*) '  Met file not on right year/month/day.'
          Return 1
        EndIf
```

```
          clk_bM = IHRM*60 + MNM
          If (clk_b .lt. clk_bM) Then
            Write (13,*) '  Met file starts too late.'
            Return 1
          EndIf
C  **     Read to starting wind record.
          NSKP =  NINT((clk_b - clk_bM)/dtM)
          Do nn = 1, NSKP
            Read (LUW(n),*) WS, WD, IWTIM
          EndDo
        EndDo
        Close (Unit=1)

C  ** Read terrain file.
        Open (Unit=1, File=FILEN(2), Status='OLD', Form='FORMATTED')
        Read (1,*) RLAT, URLONG
        Read (1,*)  NPT
        Read (1,*) (S_Ter(k), k = 1,NPT)
        Read (1,*) (z_r_Ter(k), k = 1,NPT)
        Read (1,*) (z_f_Ter(k), k = 1,NPT)
        Read (1,*) (alpha_l_Ter(k), k = 1,NPT)
        Read (1,*) (alpha_r_Ter(k), k = 1,NPT)
        Read (1,*) (l_width_Ter(k), k = 1,NPT)
        Close (Unit=1)
C  ** Calculate space step in S.
        dS = (S_Ter(NPT) - S_Ter(1))/Real(NPS)
        dSmin = dS
C  ** Calculate time of sunrise, time of sunset, length of daylight,
C  ** and solar flux at solar noon.
        Call Solar_Inp (RLAT, URLONG, JUL, tsrh, tssh, Tau, A1)
        clk_sr = tsrh*60.      ! minute of day.
        clk_ss = tssh*60.      ! minute of day.

C  ** Write inputs to .TRC file.
        Write (13,5)
        Write (13,6)
        Write (13,7)
        Write (13,8)
        Write (13,9) TITLE, USER
        Write (13,10) MO, IDA, IYR, IHRBeg, MNBeg, JUL
        Write (13,11) MO, IDA, IYR, IHREnd, MNEnd, JUL
        Write (13,12)
        Write (13,53)
        Do L = 1, 6
          Write (13,54) FILEN(L)
        EndDo
        Write (13,13)
        Write (13,14) DETAIL, NEUMANN, NPS, NPY, NPZ, Nstps_max,
     &           clk_b, clk_e, PRT_t, NS_PRT, U_day_mean,U_night_mean,
     &           Umax, Vmax, Wmax, A0, PRESS,
     &           RHO, BETA_R, GAMMA, RKF, NPM, LUWB
        Write (13,15)
        Write (13,16)  dtM, IYRM, MOM, IDAM, IHRM, MNM, clk_bM
        Do n = 1, NPM
          Write (13,17) n,LUW(n),S_Met(n),AZdwnvly(n),HTagl(n),Met_Nam(n)
        EndDo
        Write (13,18)
        Write (13,19) dS, RLAT, URLONG, NPT
        Write (13,20) 'S_Ter =       ',(S_Ter(k), k = 1,NPT)
        Write (13,20) 'z_r_Ter =     ',(z_r_Ter(k), k = 1,NPT)
        Write (13,20) 'z_f_Ter =     ',(z_f_Ter(k), k = 1,NPT)
        Write (13,20) 'alpha_l_Ter =',(alpha_l_Ter(k), k = 1,NPT)
        Write (13,20) 'alpha_r_Ter =',(alpha_r_Ter(k), k = 1,NPT)
        Write (13,20) 'l_width_Ter =',(l_width_Ter(k), k = 1,NPT)
        Write (13,21)
        Hr_clk = clk_sr/60.
        IHr_clk = INT(Hr_clk+.001)
        RMn_clk = (Hr_clk-Real(IHr_clk))*60.
        Write (13,22) clk_sr, IHr_clk, RMn_clk
        Hr_clk = clk_ss/60.
        IHr_clk = INT(Hr_clk+.001)
        RMn_clk = (Hr_clk-Real(IHr_clk))*60.
        Write (13,23) clk_ss, IHr_clk, RMn_clk
        Write (13,24) Tau/60., A1

        Return

      5 Format (5X,10('**********'))
      6 Format (5X,'*',98X,'*')
      7 Format (5X,'*',41X,'PROGRAM VALDRIFT',41X,'*',/,
```

```
     &    5X,'*',31X,'++++++VALLEY SPRAY DRIFT MODEL++++++',31X,'*',/,
     &    5X,'*',8X,'PREDICTS AIR CONCENTRATIONS THROUGHOUT A VALLEY ',
     &    'ATMOSPHERE AS A FUNCTION OF TIME.',9X,'*',/,
     &    5X,'*',11X,'DEPOSITION FIELDS ARE ALSO PREDICTED ON THE ',
     &    'VALLEY FLOOR AND SIDEWALLS.',16X,'*',/,
     &    5X,'*',11X,'TREATS THE RELEASE OF AN INERT COMPOUND THAT ',
     &    'CAN VARY IN TIME AND SPACE.',15X,'*')
    8 Format (5X,'*',98('+'),'*')
    9 Format (5X,'*',8X,'THE TITLE IS:',A,'.',5X,'USER IS ',A10,8X,'*')
   10 Format (5X,'*',11X,'PROCESSING START TIME IS (MMDDYYHHMM) ',
     &    5I2.2,10X,'[ JULIAN DAY IS',I4,' ]',8X,'*')
   11 Format (5X,'*',11X,'PROCESSING STOP  TIME IS (MMDDYYHHMM) ',
     &    5I2.2,10X,'[ JULIAN DAY IS',I4,' ]',8X,'*')
   12 Format (1H ,4X,10('**********'))
   53 Format (/,7X,'THE INPUT AND OUTPUT FILE NAMES ARE:'/)
   54 Format (10X,A)
   13 Format (/,7X,'THE RUN SPECIFICATIONS ARE:')
   14 Format (/,10X,'DETAIL = ',L2,4X,'NEUMANN = ',L2,3X,'NPS =',
     &  I4,6X,'NPY =',I4,6X,'NPZ =',I4,6X,'NSTPS_MAX =',I12,/
     &  10X,'CLK_B =',F6.1,2X,'CLK_E =',F6.1,2X,'PRT_T =',F4.0,4X,
     &  'NS_PRT =',I3,4X,/,10x,'U_day_mean =',F5.2,14X,'U_night_mean =',
     &  F5.2,/,10X,
     &  'UMAX =',F5.1,4X,'VMAX =',F5.1,4X,
     &  'WMAX =',F5.1,4X,'A0 =',F5.2,6X,'PRESS =',F6.0,/,10X,
     &  'RHO =',F5.2,5X,'BETA_R =',F5.2,2X,'GAMMA =',F5.3,3X,
     &  'RKF =',F5.2,5X,'NPM =',I3,7X,'LUWB =',I3)
   15 Format (/,7X,'THE MET STATIONS ARE:')
   16 Format (/,10X,'dTM =',F5.0,5X,'IYRM =',I3,6X,'MOM =',I3,6x,
     &  'IDAM =',I3,6X,'IHRM =',I3,6X,'MNM =',I3,/
     &  10X,'CLK_BM =',F6.0)
   17 Format (/,10X,'Sta =',I3,7X,'LUW =',I3,7X,'S_MET =',F6.0,2x,
     &  'AZvly =',F4.0,4X,'HTagl =',F6.0,2x,'MET_NAM = ',A)
   18 Format (/,7X,'THE TERRAIN SPECIFICATIONS ARE:')
   19 Format (/,10X,'dS =',F6.0,5X,'RLAT =',F7.2,2X,'LONG =',F7.2,2x,
     &  'NPT = ',I3/)
   20 Format  (10X,A,10F10.0)
   21 Format (/,7X,'THE SOLAR CHARACTERISTICS ARE:'/)
   22 Format (10X,'SR_Clk =',F7.1,4X,'IHr = ',I2.2,7X,'Mn =',F5.1)
   23 Format (10X,'SS_Clk =',F7.1,4X,'IHr = ',I2.2,7X,'Mn =',F5.1)
   24 Format (29X,'TAU =',F6.1,4x,'A1 = ',F7.1/)
      End
```

```
      Subroutine Read_Source (*)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine opens the source file, reads the input data and
C     closes the file.
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"

C  ** Read release file.
      N_Src = 0
      Open (Unit=1, File=FILEN(3), Status='OLD', Form='FORMATTED')
      Read (1,*) NPoints
      If (NPoints .ne. 0) Then
        Do N = 1, NPoints
          N_Src = N_Src + 1
            If ( (N_Src + 1) .gt. MPR) Then
              Write (13,*) 'Too many source points in the domain'
              Return 1
            EndIf
          Read (1,*) S_Src, Y_Src, Z_Src
          Call Point_Loc (S_Src, Y_Src, Z_Src,
     &                     k_Src(N_Src), j_Src(N_Src), i_Src(N_Src))
          Read (1,*) IHRB,MNB, IHRE,MNE
          clk_b_Src(N_Src) = IHRB*60 + MNB
          clk_e_Src(N_Src) = IHRE*60 + MNE
          Read (1,*) Src_Mass
          Rate_Src(N_Src) = Src_Mass/
     &              (dS*(clk_e_Src(N_Src)-clk_b_Src(N_Src))*60.)
        EndDo
      EndIf
      Read (1,*) NLines
      If (Nlines .ne. 0) Then
        Do N = 1, NLines
          Read (1,*) S_Sb, Y_Sb, Z_Sb, S_Se, Y_Se, Z_Se
          Call Line_Loc (S_Sb, Y_Sb, Z_Sb, S_Se, Y_Se, Z_Se, *998)
          Read (1,*) IHRB,MNB, IHRE, MNE
          clk_mn_b = IHRB*60 + MNB
          clk_mn_e = IHRE*60 + MNE
          Read (1,*) Src_Mass
C  **     release rate in mass per minute per meter.
          Rate = (Src_Mass/(clk_mn_e-clk_mn_b))/(Real(Npts)*dS*60.)
          Do NN = 1, Npts
            N_Src = N_Src + 1
             If ( (N_Src + 1) .gt. MPR) Then
               Write (13,*) 'Too many source points in the domain'
               Return 1
             EndIf
            k_Src(N_Src) = k_C(NN)
            i_Src(N_Src) = i_C(NN)
            j_Src(N_Src) = j_C(NN)
            clk_b_Src(N_Src) = clk_mn_b
            clk_e_Src(N_Src) = clk_mn_e
            Rate_Src(N_Src) = Rate
          EndDo
        EndDo
      EndIf
      Close (Unit=1)

C  ** Write inputs to .TRC file.
      Write (13,'(//A)') '+++++RESULTS FROM SUBROUTINE READ_SOURCE+++++'
      Write (13,16) NPoints, NLines, N_Src
      Write (13,17)
      Do N = 1, N_Src
        Write (13,18) N, k_Src(N), i_Src(N), j_Src(N),
     &                 clk_b_Src(N), clk_e_Src(N), Rate_Src(N)*dS
      EndDo

  998 Return
   16 Format (/,10X,'NPOINTS =',I3,3X,'NLINES =',I3,4X,'N_SRC =',I4)
   17 Format (/,13X,'NSRC',6X,'K_SRC',5X,'I_SRC',5X,
     &                'J_SRC',5X,'CLK_B',5X,'CLK_E',5X,'RATE (g/s)')
   18 Format (14X,I3,6X,I4,7X,I3,7X,I3,5X,F6.1,4X,F6.1,2X,E10.3)
      End
```

```
      Subroutine Surface_fluxes
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine calculates U* and Vd.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"

      Ak = 0.4              !  Van Karmen Constant
      a_neu = log(500.)
      a_sta = a_neu + 7.5
      a_uns = a_neu - 1.5

C **  Calculate the surface friction velocity, U*.
      If (U_night_mean .le. 0.1) U_night_mean = 0.1
      If (U_day_mean .le. 0.1)   U_day_mean   = 0.1
      U_star_sta = U_night_mean*Ak/a_sta
      If (U_star_sta .ge. 0.3) U_star_sta = 0.3
      U_star_neu = (U_night_mean + U_day_mean)*Ak/a_neu/2.
      If (U_star_neu .ge. 0.6) U_star_neu = 0.6
      U_star_uns = U_day_mean*Ak/a_uns
      If (U_star_uns .ge. 0.95) U_star_uns = 0.95

C **  Calculate the dry deposition velocity, Vd.
      U_mx1 = max(1.,U_night_mean ,U_day_mean)
      U_s_min = min(U_star_sta,U_star_neu,U_star_uns)
      Vd = U_s_min*U_s_min/U_mx1
      Write (13,*) ' '
      Write (13,*) ' +++++ RESULTS FROM SUBROUTINE SLFLUXES +++++ '
      Write (13,*) ' '
      Write (13,*) '        U_star_sta = ',U_star_sta,' (m/s)'
      Write (13,*) '        U_star_neu = ',U_star_neu,' (m/s)'
      Write (13,*) '        U_star_uns = ',U_star_uns,' (m/s)'
      Write (13,*) '        Vd         = ',Vd        ,' (m/s)'
      Write (13,*) ' '

      Return
      End
```

```
      Subroutine Solve_C (nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian            VALDRIFT 1.0
C
C     This subroutine integrates the conservation of species equation
C     using a fully explicit scheme - Euler forward differencing in
C     time, upwind differencing in S-dir for the advection term, and
C     central differencing in Y-dir and Z-dir for diffusion.  The
C     Courant criteria must be satisfied in specifying dS and dt.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      TOT_R = 0.                !Accumulate total release for this time.
      Do k = 1, NPS
        Call Cs_Source_Sink (k, nti)
        Call Cd_Source_Sink (k, nti)
        Call Ca_Source_Sink (k, nti)
        Call Cp_Source_Sink (k, nti)
        Do i = 1, NPZ
          cc1 = dt/A_i_j(k,i)
          Do j = 1, NPY
            F = ( Gama_cs(i,j)    + Gama_cd_S(i,j) + Gama_cd_Y(i,j) +
     &            Gama_cd_Z(i,j) + Gama_ca_S(i,j) + Gama_ca_Y(i,j) +
     &            Gama_ca_Z(i,j) + Gama_cp(i,j)   )*cc1
            C_i_j(k,i,j) = C_i_j_old(k,i,j) + F
            TOT_R = TOT_R + Gama_cs(i,j)
          EndDo
        EndDo
      EndDo

C  ** Replace old C array with current C array.
      Do k = 1, NPS
        Do i = 1, NPZ
          Do j = 1, NPY
            C_i_j_old(k,i,j) = C_i_j(k,i,j)
          EndDo
        EndDo
      EndDo

      Return
      End
```

```
      Subroutine Solar_Inp (RLAT1, URLONG, JULDAY, tsr, tss, Tau, A1)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine calculates time of sunrise. length of day, and
C     solar flux on horizontal surfaces at solar noon for any site
C     given latitude, longitude and day-of-year.  This subroutine uses
C     McCullough's 1968 approximations as given in Arch. Meteor.
C     Geophys. Bioklimat., Ser. B, 16, 129-143.
C
C     The inputs are:
C        RLAT1 - Latitude (Deg)
C        URLONG - Longitude (Deg; West/positive)
C        JULDAY - Julian Day
C
C     The outputs are:
C        tsr - time of sunrise (hour)
C        tss - time of sunset (hour)
C        Tau - Length of day (sec)
C        A1 - solar flux at solar noon (W/m^2)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Real*4 EQNTIM(25), LONG, LONCOR

C  ** Equation of time in hrs fast or slow.  The values in array EQNTIM
C  ** are HHMM for every 15 Julian days (e.g., 1, 16, 31, 46, ..., 360).
      Data EQNTIM/ -0.0533, -0.1589, -0.2233, -0.2378, -0.2064, -0.1442,
     1             -0.0689, +0.0003, +0.0475, +0.0622, +0.0425, -0.0028,
     2             -0.0558, -0.0961, -0.1058, -0.0789, -0.0186, +0.0639,
     3             +0.1517, +0.2258, +0.2683, +0.2647, +0.2094, +0.1094,
     4             -0.0131 /

C  ** Specify constants.
      PI = 3.141593
      CONV = PI/180.                       ! Degrees to radians
      DZERO = 80.                          ! Julian day of vernal equinox.
      ECCENT = .0167                       ! Eccentricity of earth's orbit
      DECMAX = (23. + 27./60.)*PI/180.     ! Maximum declination.
      OMEGA = 2.*PI/365.                   ! Revolution rate of earth.
      SC = 1367.                           ! Solar Constant (W/m^2)
      STDLON = 15.*NINT(URLONG/15.)        ! Long. of std. meridian
      ONEHR = 15.*PI/180.
      RLAT = RLAT1*CONV

C  ** Compute length of daylight period.
      D = JULDAY
      OMD = OMEGA*D
      OMDZRO = OMEGA*DZERO
      RDVCSQ = 1./(1.-ECCENT*COS(OMD))**2
      LONG = OMEGA*(D-DZERO) + 2.*ECCENT*(SIN(OMD)-SIN(OMDZRO))
      DECLIN = ASIN(SIN(DECMAX)*SIN(LONG))
      SR = -ABS(ACOS(-TAN(RLAT)*TAN(DECLIN)))
      TAU = -SR*2./ONEHR

C  ** Calculate time of solar noon as a preliminary to determining
C  ** the local time for any given hour angle.
      LONCOR = (STDLON-URLONG)*1./15.
      If (D .GT. 361.) GoTo 1
      ID = ((D-1.)/15.) + 1.
      D2 = (ID-1)*15 + 1
      TIMCOR = (EQNTIM(ID+1)-EQNTIM(ID))*(D-D2)/15. + EQNTIM(ID)
    1 If (D .EQ. 362.) TIMCOR = -0.0211
      If (D .EQ. 363.) TIMCOR = -0.0292
      If (D .EQ. 364.) TIMCOR = -0.0372
      If (D .EQ. 365.) TIMCOR = -0.0453
      TMNOON = 12.00 - LONCOR - TIMCOR

C  ** Calculate solar flux at solar noon.
      COSZ = SIN(RLAT)*SIN(DECLIN) + COS(RLAT)*COS(DECLIN)*1
      A1 = SC*RDVCSQ*COSZ
      If (A1 .LT. 0.) A1 = 0.

C  ** Calculate time-of-sunrise and time-of-sunset.
      T1 = (12. + (SR/ONEHR) + (TMNOON-12.00))
      IT = T1
      T2 = IT
      NTSR = (T1-T2)*60. + (T2*100.)
      tsr = T2*1. + (T1-T2)*1.
      tss = tsr + TAU
      TAU = TAU*3600.
```

```
      Return
      End
```

```
      Subroutine KyKz_Stable_Core (nti)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian              VALDRIFT 1.0
C
C     This subroutine calculates the stable core, and Ky and Kz.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

      If ( (clk.ge.clk_sr) .and. (clk.le.clk_ss) ) Then
C  **   Diffusivities during daytime are for unstable conditions in CBL
C  **   and neutral conditions above stable core.
        Call Descent
        Do i = 0, NPZ
          If ( TOPCBL .ge. zmean(i) ) Then
            Do j = 0, NPY
              K_s(i,j) = K_s_unstable
              K_y(i,j) = K_y_unstable
              K_z(i,j) = K_z_unstable
            EndDo
          EndIf
          If ( TOPSBL .le. zmean(i) ) Then
            Do j = 0, NPY
              K_s(i,j) = K_s_neutral
              K_y(i,j) = K_y_neutral
              K_z(i,j) = K_z_neutral
            EndDo
          EndIf
        EndDo
      Else
C  **   Diffusivities during nighttime are for stable conditions.
        Do i = 0, NPZ
          Do j = 0, NPY
            K_s(i,j) = K_s_stable
            K_y(i,j) = K_y_stable
            K_z(i,j) = K_z_stable
          EndDo
        EndDo
      EndIf

      Return
      End
```

```
      Subroutine V_Flow_Rate
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine computes the along valley volume flow rate
C     (Vdot_S) at each cross-section with wind data.  The air mass
C     source/sink term is also computed.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include "VALDRIFT.INC"

C  ** Determine along-valley peak wind speed at desired time from obs.
      Do n = 1, NPM
        Read (LUW(n),*)  WS, WD, IWTIM
        Call WNDCNV (AZdwnvly(n), WD, WS, Umet, A4, 2)
        winds(n) = Umet/facMet(n)
      EndDo

C  ** Compute V_dot at each wind observation location.
      Do n = 1, NPM
        kkk = Met_Sta_Idx(n)
        ww = winds(n)*Z_i(kkk,NPZ)
        Do i = 1, NPZ
          cc1 = ww*integ(kkk,i)
          Do j = 1, NPY
            Vdot_S(kkk,i,j) = cc1*Vdot_Fac(j)
          EndDo
        EndDo
      EndDo

C  ** Compute Source/Sink terms for all S between the wind station.
      If (NPM .eq. 1) Then
        Do k = 0, NPS
          Do i = 1, NPZ
            Do j = 1, NPY
              Gama_a(k,i,j) = 0.
            EndDo
          EndDo
        EndDo
      Else
        Do n = 1, NPM-1
          kkk1 = Met_Sta_Idx(n)
          kkk2 = Met_Sta_Idx(n+1)
          DeltaS = S(kkk2) - S(kkk1)
          IB = kkk1 + 1
          If (n .eq. 1) IB = 0
          IE = kkk2
          If (n .eq. NPM-1) IE = NPS
          Do i = 1, NPZ
            Do j = 1, NPY
              S_S = (Vdot_S(kkk1,i,j) - Vdot_S(kkk2,i,j)) / DeltaS
              Do k = IB, IE
                Gama_a(k,i,j) = S_S
              EndDo
            EndDo
          EndDo
        EndDo
      EndIf

C  ** Compute Vol Flow and velocity for all S.
      If (NPM .eq. 1) Then
        kkk = Met_Sta_Idx(1)
        Do k = 0, NPS
          Do i = 1, NPZ
            Do j = 1, NPY
              Vdot_S(k,i,j) = Vdot_S(kkk,i,j)             ! real winds
c             Vdot_S(k,i,j) = A_i_j(k,i)                  ! 1 m/s winds
c             Vdot_S(k,i,j) = 5.*A_i_j(k,i)               ! 5 m/s winds
c             Vdot_S(k,i,j) = 0.                          ! 0 m/s winds
            EndDo
          EndDo
        EndDo
      Else
        Do n = 1, NPM-1
          kkk1 = Met_Sta_Idx(n)
          kkk2 = Met_Sta_Idx(n+1)
          IB = kkk1 + 1
          IE = kkk2
          If (n .eq. NPM-1) IE = NPS
          Do k = IB, IE
            Do i = 1, NPZ
```

```
              Do j = 1, NPY
                Vdot_S(k,i,j) = Vdot_S(k-1,i,j) + Gama_a(k,i,j) * dS
              EndDo
            EndDo
          EndDo
        EndDo
        kkk1 = Met_Sta_Idx(n) - 1
        Do k = kkk1, 0, -1
          Do i = 1, NPZ
            Do j = 1, NPY
              Vdot_S(k,i,j) = Vdot_S(k+1,i,j) - Gama_a(k,i,j) * dS
            EndDo
          EndDo
        EndDo
      EndIf

C  ** Write out results.
      If (DETAIL) Then
        Write (13,'(//A)') '+++++RESULTS OF SUBROUTINE V_FLOW_RATE+++++'
        Write (13,'(/A,F5.1,A/)') '  ** Vdot for time ',clk, ' **'
        Do k = 0, 0  !NPS, NS_PRT
          Write (13,*) '  k = ',k
          Write (13,'(A,12(4x,I2,4x))') 'Y --> ',(j,j=1,NPY)
          Do i = NPZ, 1, -1
            Write (13,'(I6,12E10.3)') i,(Vdot_S(k,i,j),j=1,NPY)
          EndDo
        EndDo
C  **   Write out alongvalley wind speeds.
        Write (13,'(/A,F5.1,A/)') '  ** U for time ',clk, ' **'
        Do k = 0, 0  !NPS, NS_PRT
          Write (13,*) '  k = ',k
          Write (13,'(A,12(4x,I2,4x))') 'Y --> ',(j,j=1,NPY)
          Do i = NPZ, 1, -1
            Write (13,'(I6,12E10.3)') i,
     &                       ((Vdot_S(k,i,j)/A_i_j(k,i)),j=1,NPY)
          EndDo
        EndDo
      EndIf

      Return
      End
```

```
      Subroutine WNDCNV (AZ, A1, A2, A3, A4, IC)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995             KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This subroutine computes the components of winds relative
C     to any desired direction given the wind speed and wind direction.
C     The positive u-component is in the specified direction.  The
C     positive v-component follows the conventional right-hand rule
C     where the z-axis is up.
C
C     This subroutine also converts from components relative to any
C     specified direction, to wind direction (True N) and speed.
C
C     AZ = direction of the positive u-component (deg true North).
C     WD= wind direction (deg true North)
C     WS = wind speed (m/s)
C     U  = u-component of winds (m/s)
C     V  = v-component of winds (m/s)
C
C     IC = 1 --> u,v   to WD,WS --> A1=u,  A2=v,  A3=WD, A4=WS
C     IC = 2 --> WD,WS to u,v   --> A1=WD, A2=WS, A3=u,  A4=v
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DTOR = 3.14159/180.       ! Degrees to radians
      If (IC .eq. 1) Then       ! u,v to WD,WS
        U = A2
        V = -A1
        WS = SQRT(U*U + V*V)
        If (WS .EQ. 0.) Then
          WD = 0.
        Else
          If (U.GE.0. .AND. V.GE.0.) WD = 270. - ASIN(V/WS)/DTOR
          If (U.LT.0. .AND. V.LE.0.) WD = ASIN(ABS(U)/WS)/DTOR
          If (U.EQ.0. .AND. V.LE.0.) WD = 360.
          If (U.GE.0. .AND. V.LE.0.) WD = 270. + ACOS(U/WS)/DTOR
          If (U.LE.0. .AND. V.GE.0.) WD = 90.+ ASIN(V/WS)/DTOR
        EndIf
        WD = WD - (180.-AZ)
        If (WD .LT. 0.)   WD = 360. + WD
        If (WD .GT. 360.) WD = WD - 360.
        A3 = WD
        A4 = WS
      EndIf
      If (IC .eq. 2) Then    ! WD,WS to u,v
        WD = A1
        WS = A2
        WDP = WD + (180.-AZ)
        If (WDP .LT. 0.) WDP = 360. + WDP
        THET = (270.-WDP)*DTOR
        U = -WS*SIN(THET)
        V = WS*COS(THET)
        A3 = U
        A4 = V
      EndIf
      Return
      End
```

```
C     VALDRIFT.INC
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995            KJ Allwine and X Bian                VALDRIFT 1.0
C
C     This is the include file for VALDRIFT.
C
C     MPS  = maximum number of points in S-dirction.
C     MPY  = maximum number of points in Y-dirction.
C     MPZ  = maximum number of points in Z-dirction.
C     MPT  = maximum number of input terrain cross-sections.
C     MPM  = maximum number of input met stations.
C     MPR  = maximum number of source grid points.
C     MPD  = maximum number of deposition points = 2(MPZ-1) + MPY
C     MPS1 = MPS + 1
C     MPY1 = MPY + 1
C     MPZ1 = MPZ + 1
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Parameter( MPS = 200,      MPY = 31,            MPZ = 31,
     &           MPT = 10,       MPM =  1,            MPR = 50,
     &           MPD = 93,       MPS1= 201,           MPY1= 32,
     &           MPZ1= 32  )

C ** Typing Statements.

      Character  Met_Nam(MPM)*8,   TITLE*45,          USER*10,
     &           FILEN(6)*80

      Logical DETAIL,               NEUMANN,           TESTOUT,
     &        LOGPRT,               DATDMP

      Integer Met_Sta_Idx(MPM),   LUW(MPM)

      Integer k_c(MPR),           j_c(MPR),          i_c(MPR)

      Integer k_Src(MPR),         i_Src(MPR),        j_Src(MPR)

      Integer i_Gnd(MPD),         j_Gnd(MPD)

C ** Input diffusivities.
      Real*4  K_s_unstable,      K_s_neutral,        K_s_stable,
     &        K_y_unstable,      K_y_neutral,        K_y_stable,
     &        K_z_unstable,      K_z_neutral,        K_z_stable,
     &        KSmax,             KYmax,              KZmax

C ** Input wind data.
      Real*4  S_Met(MPM),        AZdwnvly(MPM),      HTagl(MPM),
     &        winds(MPM),        facMet(MPM)

C ** Input source data.
      Real*4  Clk_b_Src(MPR),    Clk_e_Src(MPR),     Rate_Src(MPR)

C ** Input terrain data.
      Real*4  S_Ter(MPT),        alpha_l_Ter(MPT),   alpha_r_Ter(MPT),
     &        z_r_Ter(MPT),      z_f_Ter(MPT),       l_width_Ter(MPT)

C ** Terrain data interpolated in S.
      Real*4  alpha_l(0:MPS),    alpha_r(0:MPS),     theta(0:MPS),
     &        z_r(0:MPS),        z_f(0:MPS),         l_width(0:MPS),
     &        S(0:MPS),          Z_i(0:MPS,0:MPZ),   dY_i(0:MPS,0:MPZ),
     &        gama(0:MPS,0:MPZ), dZ_i(0:MPS,MPZ),    A_i(0:MPS,MPZ),
     &        A_i_j(0:MPS,MPZ),  dY_i_bar(0:MPS,MPZ), zmean(0:MPZ),
     &        beta(0:MPY),       dZ_i_bar(0:MPS,MPZ), D_Sfc(MPS,MPD)

C ** Air volume flow arrays.
      Real*4  integ(0:MPS,MPZ),  Vdot_S(0:MPS,MPZ,MPY),
     &        Vdot_Fac(MPY),     Gama_a(0:MPS,MPZ,MPY)

C ** Release and deposition source/sink terms at location S.
      Real*4  Gama_cs(MPZ,MPY), Gama_cp(MPZ,MPY)

C ** Diffusion source/sink arrays at location S.
      Real*4  Gama_cd_Z(MPZ,MPY),Gama_cd_Y(MPZ,MPY), Gama_cd_S(MPZ,MPY)

C ** Advection source/sink arrays at location S.
      Real*4  Gama_ca_Z(MPZ,MPY),Gama_ca_Y(MPZ,MPY), Gama_ca_S(MPZ,MPY)

C ** Diffusivity arrays at location S.
      Real*4  K_y(0:MPZ,0:MPY),  K_z(0:MPZ,0:MPY),   K_s(0:MPZ,0:MPY)

C ** Lateral and vertical diffusion permeability arrays at location S.
```

```
      Real*4  BC_Z_top(MPS,MPY), BC_Y_left(MPS,MPZ),
     &        BC_Z_bot(MPS,MPY), BC_Y_right(MPS,MPZ), Dif_Coef

C  ** Concentration arrays.
      Real*4  C_i_j     (0:MPS1,0:MPZ1,0:MPY1),
     &        C_i_j_old(0:MPS1,0:MPZ1,0:MPY1),
     &        C_Bkgnd

C  ** Ground deposition array.
      Real*4  D_Gnd(MPS,MPD)

C  ** Common Blocks.

      Common /INPUT/ TITLE,      USER,       DETAIL,     NEUMANN,   NPS,
     &               NPY,        NPZ,        NPT,        NPM,       NPC,
     &               pi,         Nstps_max,TESTOUT,      FILEN, DATDMP

      Common /GRIDM/ alpha_l,    alpha_r,    theta,      z_r,       z_f,
     &               l_width,    S,          dS,         Z_i,       A_i,
     &               A_i_j,      dZ_i,       dY_i,       dZ_i_bar,  dY_i_bar,
     &               beta,       gama

      Common /TIMES/ IYR,        MO,         IDA,        JUL,       clk_b,
     &               clk_e,      IHRBeg,     MNBeg,      IHREnd,    MNEnd,
     &               clk_sr,     clk_ss,     clk,        clk_bM,    rts,
     &               Hr_clk,     IHr_clk,    RMn_clk

      Common /FLOWS/ Vdot_S,     Gama_a,     integ,      Vdot_Fac,  S_Met,
     &               winds,      AZdwnvly,   HTagl,      Met_Nam,   LUWB,
     &                 dtM,        facMet,     LUW,      Met_Sta_Idx

      Common /SOURC/ N_Src,      k_Src,      i_Src,      j_Src,     clk_b_Src,
     &               Npts,       k_c,        j_c ,       i_c,       clk_e_Src,
     &               Rate_Src,   Gama_cs

      Common /CONCE/ C_i_j,      C_i_j_old,C_Bkgnd

      Common /MASSB/ TOT_C,      TOT_RELE,   FLUX_NET,   TOT_R,     residual

      Common /SOLAR/ RLAT,       URLONG,     TAU,        A1

      Common /DESCE/ TOPCBL,     TOPSBL,     BETA_R,     GAMMA,     RKF,
     &               FACT1,      THTOT,      CSHP,       zmean,     wdth_mean,
     &               EL_T,       A0,         RHO,        PRESS

      Common /PRINT/ PRT_t,      Nt_PRT,     NS_PRT,     LOGPRT

      Common /COURA/ dSmin,      dYmin,      dZmin,      Umax,      Vmax,
     &               Wmax,       KSmax,      KYmax,      KZmax,     dt

      Common /TERAN/ alpha_l_Ter,     alpha_r_Ter,    z_r_Ter,
     &               z_f_Ter,         l_width_Ter,    S_Ter

      Common /BOUND/ BC_Y_left,       BC_Y_right,     BC_Z_bot,
     &               BC_Z_top,        Dif_Coef

      Common /TURBU/ K_s_unstable,    K_s_neutral,    K_s_stable,
     &               K_y_unstable,    K_y_neutral,    K_y_stable,
     &               K_z_unstable,    K_z_neutral,    K_z_stable,
     &               K_s,             K_z,            K_y,
     &               Gama_cd_Z,       Gama_cd_Y,      Gama_cd_S,
     &               Gama_ca_Z,       Gama_ca_Y,      Gama_ca_S

      Common /GROND/ N_Gnd,           i_Gnd,          j_Gnd,
     &               D_Gnd,           Vd,             Gama_cp,    D_Sfc


      Common /SLFLX/ U_star_sta ,     U_star_neu,     U_star_uns,
     &               U_day_mean ; U_night_mean
```

```
      Program PostProcess
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     May 1995           KJ Allwine and X Bian             VALDRIFT 1.0
C
C     This routine opens '.bin' file and writes text files of VALDRIFT
C     outputs.
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      Include   "VALDRIFT.INC"
      Parameter (N=50)
      Integer   i_gp(N), j_gp(N)
      Character bin_file*60, asc_file*60, PATH*60, BINNAM*8

C  ** Set file path of input BIN file and output ASC file.
      Nch = 20
      PATH = 'HD:VALDRIFT:OUTPUTS:'       ! MAC
c     PATH = 'C:\VALDRIFT\OUTPUTS\'       ! DOS

C  ** Open input BIN file.
   10 BINNAM = '  '
      Write (*,*) 'Enter name of BIN file (e.g., BRUSHVAL; ≤ 8 char.)'
      Read (*,'(a)') BINNAM
      Ncr = Index(BINNAM,' ') - 1
      If (Ncr .le. 0) Ncr = 8
      bin_file = PATH(1:Nch)//BINNAM(1:Ncr)//'.BIN'
      Open (11, File=bin_file, Status ='OLD', Form='UNFORMATTED',
     &                                                    Iostat=IER)
      If (IER .ne. 0) Then
        Write (*,*)
        Write (*,102) bin_file
        Write (*,*) 'Do you want to enter a new BIN file name?'
        Write (*,*) 'If yes, please enter 1; if not, enter 2'
        Read (*,*) icnt
        If (Icnt .eq. 1) GoTo 10
        GoTo 800
      EndIf

C  ** Open output ASC file.
      asc_file = PATH(1:Nch)//'VALDRIFT.ASC'
      Open (12, File=asc_file, Status ='UNKNOWN', Form='FORMATTED')

C **  Read model computational domain parameters from BIN file.
      Read (11) NPS,NPY,NPZ,clk_b,clk_e,PRT_t,dS
      S_l = dS*real(NPS)
      Ihr_b = int(clk_b/60.)
      min_b = int(clk_b - Ihr_b*60)
      Ihr_e = int(clk_e/60.)
      min_e = int(clk_e - Ihr_e*60)
      nptrt = int(PRT_t)
C **  Read model grid points from BIN file.
      Read (11) dY_i,Z_i,dZ_i
C **  Read the ground surface grid points from BIN file.
      Read (11) N_Gnd,i_Gnd,j_Gnd

C **  Write general information to the sreen.
      Write (*,*) '  '
      Write (*,*) '    ***** GENERAL INFORMATION *****'
      Write (*,*) '  '
      Write (*,*) 'Length of domain (alongvalley/S_dir):',S_l,'(m)'
      Write (*,*) 'Number of grid points in S-direction:',NPS
      Write (*,*) 'Number of grid points in Y-direction:',NPY
      Write (*,*) 'Number of grid points in Z-direction:',NPZ
      Write (*,'(a,2i2.2)') 'The simulation beginning time:',Ihr_b,min_b
      Write (*,'(a,2i2.2)') 'The simulation ending time:    ',Ihr_e,min_e
      Write (*,*) 'Time step for writing concentrations to file:',nptrt,
     &            ' (minutes)'
      Write (*,*) '  '
      Write (*,*)'____________________________________________________'
      Write (*,*)'  '
      Write (*,*) '  '

C **  Write general information into the ASC file.
      Write (12,*) '  '
      Write (12,*) '    ***** GENERAL INFORMATION *****'
      Write (12,*) '  '
      Write (12,*) 'Length of domain (alongvalley/S_dir):',S_l,'(m)'
      Write (12,*) 'Number of grid points in S-direction:',NPS
      Write (12,*) 'Number of grid points in Y-direction:',NPY
      Write (12,*) 'Number of grid points in Z-direction:',NPZ
      Write (12,'(a,2i2.2)')'The simulation beginning time:',Ihr_b,min_b
      Write (12,'(a,2i2.2)')'The simulation ending time:    ',Ihr_e,min_e
```

```
      Write (12,*)'Time step for writing concentrations to file:',nptrt,
     &            ' (minutes)'
      Write (12,*) '-----------------------------------------'
      Write (12,*)' '

C **  Specify the time for the desired data.
  199 Write (*,*) 'Please input hour (0-23) you want:'
      Read (*,*) Ihr_u
      Write (*,*)
      Write (*,*) 'Please input minute (0-59) you want:'
      Read (*,*) min_u
      time = Ihr_u*60 + min_u
      Write (*,*)' '
      Write (*,'(a,2i2.2)') 'The time you input is :',Ihr_u,min_u
      Write (*,*) 'If it is right, please enter 1; if not, enter 2'
      Read (*,*) nlg
      If (nlg .ne.1 ) GoTo 199
      If ( (time.lt.clk_b) .or. (time.gt.clk_e) ) Then
        Write (*,*)' '
        Write (*,*) 'Sorry, the time you want is not in the VALDRIF'
     &  ,' output data file,'
        Write (*,*) 'please redo !'
        Write (*,'(a,2i2.2)')'Information: simulation beginning time =',
     &  Ihr_b,min_b
        Write (*,'(a,2i2.2)')'Information: simulation ending time =   ',
     &   Ihr_e,min_e
        GoTo 199
      Endif

C **  Extract the data for the desired time from the BIN file.
      Write (*,*) ' '
      Write (*,*) 'Starting to extract the concentration data from',
     &            ' the BIN file,'
      Write (*,*) ' '
      Write (*,*) 'Please wait ...'
      Do iread = 1,10000
C **    Read time, CPL/SBL and concentrations from BIN file.
        Read (11,end=901) clk, TOPCBL, TOPSBL,
     &                    (((C_i_j(k,i,j),k=1,NPS),i=1,NPZ),j=1,NPY),
     &                    ((D_Gnd(k,iG),k=1,NPS),iG=1,N_Gnd)
        If (clk .ge. (time - PRT_t/2.)) Then
          Write (12,'(a,2i2.2,a)') 'The time you want is ',Ihr_u,min_u,
     &                          ' (LST), at this time:'
          Write (12,*) 'TOPCBL =',TOPCBL,'(m),',' TOPSBL =',TOPSBL,'(m)'
          Write (12,*) '-----------------------------------------'
          GoTo 900
        Endif
      Enddo
  900 Write (*,*) ' '
      Write (*,*) 'The concentration data you want has been extracted.'

C **  Specify every nth along-valley location to write.
      Write (*,*) ' '
      Write (*,*) 'Specify every nth along-valley location to output'
      Read (*,*) NS_PRT
      If ( (NS_PRT.le.0) .or. (NS_PRT.gt.NPS) ) NS_PRT = 1

C **  Specify if want Log10 to output.
      Write (*,*) ' '
      Write (*,*) 'Do you want to have Log10 of the output?'
      Write (*,*) 'If yes, please enter 1; if not, enter 2'
      Read (*,*) ILYes
      If (ILYes .eq. 1) Then
        Do k = 1, NPS
          Do i = 1, NPZ
            Do j = 1, NPY
              If (C_i_j(k,i,j) .gt. 0.) Then
                C_i_j(k,i,j) = LOG10(C_i_j(k,i,j))
              Else
                C_i_j(k,i,j) = -99.
              EndIf
            EndDo
          EndDo
        EndDo
        Do k = 1, NPS
          Do iG = 1, N_Gnd
            If (D_Gnd(k,iG) .gt. 0.) Then
              D_Gnd(k,iG) = LOG10(D_Gnd(k,iG))
            Else
              D_Gnd(k,iG) = -99.
```

```
              EndIf
            EndDo
          EndDo
        EndIf

C **    Process the cross view information.
        Write (*,*) ' '
        Write (*,*) 'Do you want to have a cross-section view?'
        Write (*,*) 'If yes, please enter 1; if not, enter 2'
        Read (*,*) nlg
        If (nlg .eq. 1) Then
          Write (*,*)' '
  299     Write(*,*)'please enter location in S_dir of the cross section,'
          Write (*,*) 'which is far from the origin in meters:'
          Read (*,*) S_crs
          k = Nint(S_crs/dS)
          If ( (k.lt.1) .or. (k.gt.NPS) ) Then
            Write (*,*)' '
            Write (*,*) 'Sorry, the location of cross section you want is'
            Write (*,*) 'out of the computational domain, please redo !'
            Write (*,*)'Infomation: Length of domain (alongvalley/S_dir):'
     &                ,S_l,'(m)'
            GoTo 299
          Endif
C **      Write the cross view into the ASC file.
          Write (12,*) ' '
          Write (12,*) 'Cross-View Concentrations (g/m^3) at the ',
     &                 'Cross-Section of S = ', k*dS, '(m) '
          Write (12,*) 'Z,    Y: -->'
          Do i = 1, NPZ
            ii = NPZ - i +1
             Write (12,101) ii,(C_i_j(k,ii,j),j = 1,NPY)
          EndDo
          Write (12,*) '------------------------------------------'
        Endif

C **    Process the side view information.
        Write (*,*) ' '
        Write (*,*) 'Do you want to have a side view?'
        Write (*,*) 'If yes, please enter 1; if not, enter 2'
        Read (*,*) nlg
        If (nlg .eq. 1) Then
          j = int((NPY+1)/2)
C **      Write the side view (at Y=(NPY+1)/2) into the ASC file.
          Write (12,*) ' '
          Write (12,*) 'Side-View Concentrations (g/m^3) Along the ',
     &                 'Valley Center Line'
          Write (12,*) 'S,    Z: -->'
          Do k = NS_PRT, NPS, NS_PRT
            Write (12,101) k,(C_i_j(k,i,j),i = 1,NPZ)
          EndDo
          Write (12,*) '------------------------------------------'
        Endif

C **    Process the top view information.
        Write (*,*)' '
        Write (*,*) 'Do you want to have a top view?'
        Write (*,*) 'If yes, please enter 1; if not, enter 2'
        Read (*,*) nlg
        If (nlg .eq. 1) Then
          i = int((NPZ+1)/2)
C **      Write the top view (at Z=(NPZ+1)/2) into the ASC file.
          Write (12,*) ' '
          Write (12,*) 'Top-view Concentrations (g/m^3) at the Level ',
     &                 'of Half-Mean-Valley Depth'
          Write (12,*) 'S,    Y: -->'
          Do k = NS_PRT, NPS, NS_PRT
             Write (12,101) k,(C_i_j(k,i,j),j = 1,NPY)
          EndDo
          Write (12,*) '------------------------------------'
        Endif

C **    Process the ground-surface view information.
        Write (*,*) ' '
        Write (*,*) 'Do you want to have a ground surface view?'
        Write (*,*) 'If yes, please enter 1; if not, enter 2'
        Read (*,*) nlg
        If (nlg .eq. 1) Then
C **      Write the ground view into the ASC file.
          Write (12,*) ' '
```

```
          Write (12,*) 'Ground-level Concentrations (m/m^3)'
          Write (12,*)  ' S  !       On the left sidewall      ! ',
     &                   '           On the valley floor       ! ',
     &                   '           On the right sidewall     ! '
          If (N_gnd .le. 9) Then
            Do k = NS_PRT, NPS, NS_PRT
               Write (12,101) k,(C_i_j(k,i_Gnd(j),j_Gnd(j)),
     &                                                  j = 1,N_Gnd)
            EndDo
          Else
C **        Left sidewall plot grid.
            i_gp(1) = i_Gnd(1)
            j_gp(1) = j_Gnd(1)
            i_gp(2) = i_Gnd(int((NPZ+1)/2))
            j_gp(2) = j_Gnd(int((NPZ+1)/2))
            i_gp(3) = i_Gnd(NPZ)
            j_gp(3) = j_Gnd(NPZ)
C **        Valley floor plot grid.
            ig1 = NPZ + int((N_Gnd/2 - NPZ)/2)
            i_gp(4) = i_Gnd(ig1)
            j_gp(4) = j_Gnd(ig1)
            i_gp(5) = i_Gnd(int(N_Gnd/2))
            j_gp(5) = j_Gnd(int(N_Gnd/2))
            ig2 = N_Gnd - NPZ - int((N_Gnd/2 - NPZ)/2) +1
            i_gp(6) = i_Gnd(ig2)
            j_gp(6) = j_Gnd(ig2)
C **        Right sidewall plot grid.
            i_gp(7) = i_Gnd(N_Gnd- NPZ + 1)
            j_gp(7) = j_Gnd(N_Gnd- NPZ + 1)
            i_gp(8) = i_Gnd(N_Gnd-int(NPZ/2))
            j_gp(8) = j_Gnd(N_Gnd-int(NPZ/2))
            i_gp(9) = i_Gnd(N_Gnd)
            j_gp(9) = j_Gnd(N_Gnd)
            Do k = NS_PRT, NPS, NS_PRT
              Write (12,'(i3,3(a,3(E9.3,1x)),a)') k,' !  ',
     &         (C_i_j(k,i_gp(j),j_gp(j)),j = 1,3),
     &         ' !  ',
     &         (C_i_j(k,i_gp(j),j_gp(j)),j = 4,6),
     &         ' !  ',
     &         (C_i_j(k,i_gp(j),j_gp(j)),j = 7,9),
     &         ' ! '
            EndDo
          Endif
          Write (12,*) '-----------------------------------'
        Endif

C **  Write all ground deposition values to the ASC file.
        Write (12,*) '  '
        Write (12,*) 'Ground Deposition Values (g/m^2) '
        Write (12,*) 'S,   Ground Index: -->'
        Do k = NS_PRT, NPS, NS_PRT
           Write (12,'(i3,a,30(E9.3,1x))') k,' ! ',
     &                                   (D_Gnd(k,iG),iG=1,N_Gnd)
        EndDo
        Write (12,*) '-----------------------------------'

C **  Write all ground-level concentration values to the ASC file.
        Write (12,*) '  '
        Write (12,*)'Ground-Level Concentration Values (g/m^3)'
        Write (12,*) 'S,   Ground Index: -->'
        Do k = NS_PRT, NPS, NS_PRT
           Write (12,'(i3,a,30(E9.3,1x))') k,' ! ',
     &                   (C_i_j(k,i_Gnd(j),j_Gnd(j)),j=1,N_Gnd)
        EndDo
        Write (12,*) '-----------------------------------'

  800 Write (*,*)
      Write (*,*) 'Post-Processing Completed, <Return> to exit'
      Read (*,*)
      GoTo 911
  901 Write (*,*)
      Write (*,*) 'Read past end of BIN file, <Return> to exit'
      Read (*,*)
C **  Close files.
  911 Close (11)
      Close (12)

      Stop
  102 Format ('***Error Opening BIN file***: ',A)
  101 Format (i4,50(1x,E9.3))
```

```
  112 Format (i4,3(3X,E11.3))
      End
```

DISTRIBUTION

No. of Copies

OFFSITE

2 DOE Office of Scientific and Technical Information

2 Dr. Harold Thistle
U.S. Forest Service
Missoula Technology and Development Center
Fort Missoula, Building 1
Missoula, MT 59801

Mr. John W. Barry
USDA Forest Service
Forest Health Protection
2121C Second St., Suite 102
Davis, CA 95616

Dr. Jim Hadfield
Forestry Sciences
1133 N. Western Ave.
Wenatchee, WA 98801

Mr. Bob Ekblad
814 Whitaker
Missoula, MT 59803

Dr. Milt Teske
Continuum Dynamics, Inc.
P.O. Box 3073
Princeton, NJ 08543

5 Dr. K. Jerry Allwine
Allwine Environmental Services
P. O. Box 904
Richland, WA 99352

No. of Copies

Mr. Iral Ragenovich
USDA Forest Service
Forest Pest Management
P. O. Box 3623
Portland, OR 97208

ONSITE

13 Pacific Northwest Laboratory

C. D. Whiteman (5)
X. Bian
W. R. Barchet
Publishing Coordination
Technical Report Files (5)