Technical Report

on

Objective Analysis (OA) Project

Prepared for : BIO
Department of Fishery and Ocean

by

Ross Hendry and **Ian He**
April, 1996

Objective analysis Technical Report

Contents

ABSTRACT

Objective analysis technique has been widely used in the field of Oceanography, meterolagy and bla bla.. ...

This technical report overviews the work carried out in the OA (Objective Analysys) project with focous on the Objective Analysis algorithms, Computer programs and user's guide. The history of the OA program development with various versions are reviewed in Appendix A. A detailed reference manual for the OA program functions is given in Appendix B.

A detailed algorithm and implementations of the objective analysis theory, the kd tree data structure and the nearest neighbor search method are described and presented with a number of examples based on the final version of the OA program oax6. A documentation on how to use OA programs and the user's guide on all OA packages including OA stand alone program (oax6), OA C library functions, OA F77 library functions, kd tree store and load programs and OA AVS modules are presented in Section 4.

A practical application problem is solved by using the OA program in Section 5. Finally, a discussion is also included.

Bibliography on objective analysis and related topics are given in the Reference List.

1 Introduction

OA applies the method of optimal interpolation (objective analysis) to estimate the values of variables at specified points in a multidimensional space. For each point, the computation is based on a weighted average of the values of a specified number of data points (i.e. "nearest neighbours") which are closest to that point. For the sake of efficiency, several dependent variables can be interpolated simultaneously using the same weights if the underlying statistical model and relative noise level in the input data are assumed to be same for each dependent variable.

The final release of OA program *oax*6 extends the capability of earlier implementations by allowing the parameters controlling the calculation of the weights to vary as a function of the independent variables, and by allowing the specification of a relative uncertainty (NOISE) for each input data record separately.

The names of variables to estimate (i.e. dependent variables) and the components of the space (i.e. independent variables) are specified with the DEPENDENT and INDEPENDENT identifiers in a deck file respectively. Overall scales for the independent variables (GLOBAL_SCALES) are used to non-dimensionalize the independent variables and define an overall metric (distance calculation) for sorting the input data and creating a data tree. This allows for the efficient selection of the NUM_CLOSEST nearest neighbours to control the interpolation of each grid point read from the GRIDFILE.

The GRIDFILE is a text file containing a list of grid points - that is, points where estimates are to be calculated (independent variables) - and parameters that define local scales and local axis orientation for each grid point. The local scales replace the initially specified GLOBAL_SCALES, and control both the selection of nearest neighbours and the calculation of the interpolation weights.

The locations of data and grid points are specified by a list of coordinates (x, y, z, t, ...). The local scales (lx, ly, lz, lt, ...) are applied to each coordinate. Provision is made for a local rotation of the coordinate system applying specifically to the first two coordinates specified (x, y), which are assumed to represent a right-handed Cartesian coordinate system. A rotation angle is specified for each grid point in the GRIDFILE. If this angle is non-zero, an anticlockwise rotation of the first two coordinates (x, y) by the specified angle is carried out to produce transformed coordinates (x', y'), and local scaling is applied to the transformed coordinates. One application of this feature would be to create horizontal maps of an oceanic field over variable bottom topography. A rotation angle at each grid point would be specified to transform to a coordinate system with along- and cross-isobath orientation, allowing the specification of local scales appropriate for variations in the along- and cross-isobath directions.

Each output data cycle contains the results of the estimation at one grid point, and includes the independent variable values (original coordinates), the computed dependent variable values and an estimate of the relative error of the estimated dependent variables in the non-dimensional output variable.

The methodology is primarily aimed at estimating the spatially or temporally varying part of a

process. Measured values typically include the contribution of an underlying mean value. This mean field may itself vary slowly in space or time, but by definition it does so only on scales which are large compared to the typical scales of variation of the dependent variable(s) with respect to the independent variable(s) as expressed in the correlation model underlying the optimal estimation. Certain details of the estimation procedure depend on whether the mean value of the dependent variable(s) is assumed to be known and has already been removed from the input dependent variable(s) or whether the input data for the dependent variable(s) still include a mean value. The METHOD identifier is used to specify which situation holds.

To speed up the selection of nearest neighbours, an internal data structure (a k-dimensional or "k-d" tree) is used to sort the input data. The structure of the k-d tree depends upon the scales specified to the GLOBAL_SCALES identifier. With the BUCKET identifier, the user may specify the number of data points to be included in the finest subdivision of the sorted data. Choice of GLOBAL_SCALES and BUCKET affect the speed of the nearest neighbours selection, but should not affect the results.

OA uses certain assumptions about the statistical nature of the data. A default covariance model is implemented in the code. Other covariance models may be introduced by the user. It uses an objective analysis method to interpolate/extrapolate data at grid points based on the values at a finite number of nearest neighbours.

The method is called "optimal", because if the covariance function is an accurate model of the spatial relationship of the data and if the assumption about noise accurately reflects the level of actual noise in the data, then the method will give the least expected error of any linear estimate.

The error computed uses a single estimate of NOISE level of the input data. Dimensional error estimates are obtained by multiplying the output OERR parameter by the standard deviation of the particular field involved (temperature, salinity, ...). Estimates of the required standard deviations must be provided by the user.

OA might be used in non-obvious ways. For example, instead of using pressure or depth as independent variables, one might use the temperature, derived from an overall average vertical profile of temperature at the input pressure or depth, which would "stretch" the space in areas of rapidly changing temperature. As another example, instead of using latitude and longitude to measure horizontal distance, one might have reason to use station number as a dimension, where the stations are not necessarily equidistant.

In addition to the OA stand alone program, OA C library and F77 library functions are also developed. The OA C library functions are based on oax6 program. It allows the user to use OA within user's C applications. An additional feature of C library is to allow the use of an user supplied covariance function to replace the default covariance function. OA F77 library functions are based on OA C library functions. It has the same features as C library functions. The difference between C Library and F77 library is that F77 library functions only support SUN system right now. C Library functions have been ported to a number of systems including SUN, DEC, IBM, SGI, HP, FreeBSD and Linuix.

An independent program for building kd tree and an independent program for loading kd tree are also developed as the by-product of the OA programs. Two AVS modules are developed to generated the AVS field files for viewing the optimal estimation results.

2 Algorithms

The optimal estimator loads data points (independent and dependent variables) and creates an internal data structure (the k-d tree). Points where the estimation is to be performed are read from the grid file one at a time and the estimated values of each dependent variables are obtained from the weighted sums of the values at the closest data points, with the weights determined by the optimal linear estimation algorithm. The k-d tree provides an efficient way of selecting a set of nearest neighbours.

This program handles inhomogeneous and anisotropic variability by allowing the specification of local coordinate orientation, local scales for each grid point and a specified uncertainty for each input data point. Nearest neighbours are selected at each grid point after local coordinate rotation and scaling. The distance metric and the concept of "nearest" depend on the local scales. It is the responsibility of the user to specify the local scales in a physically meaningful way.

2.1 kd tree

The kd-tree is a binary tree data structure with terminal buckets. The maximum number of data that can be stored in a bucket is defined as "bucket size". For a given data set, larger bucket results in smaller number of terminal buckets and hence less of computer time to build the tree.

When the data are read in, an index array is created to record the order of the original data.

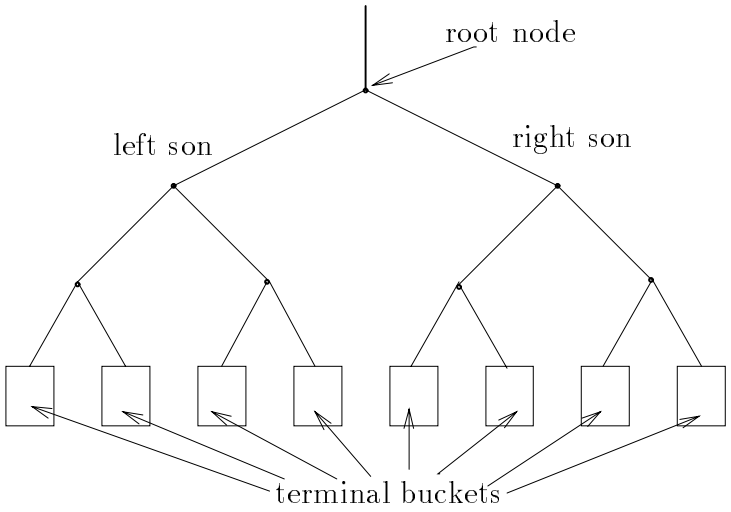

Figure 1: kd tree

During the sorting, the original order changes. The data array is static. Global scale is used to sort the data and decide the cut dimension and its cut value. The maximum number of data points is not limited by the program. The memory needed for storing the data is dynamically allocated.

The kd tree building process can be described by the following figure. The data are virtually put into the terminal buckets at the ends of the tree through link of the index array.

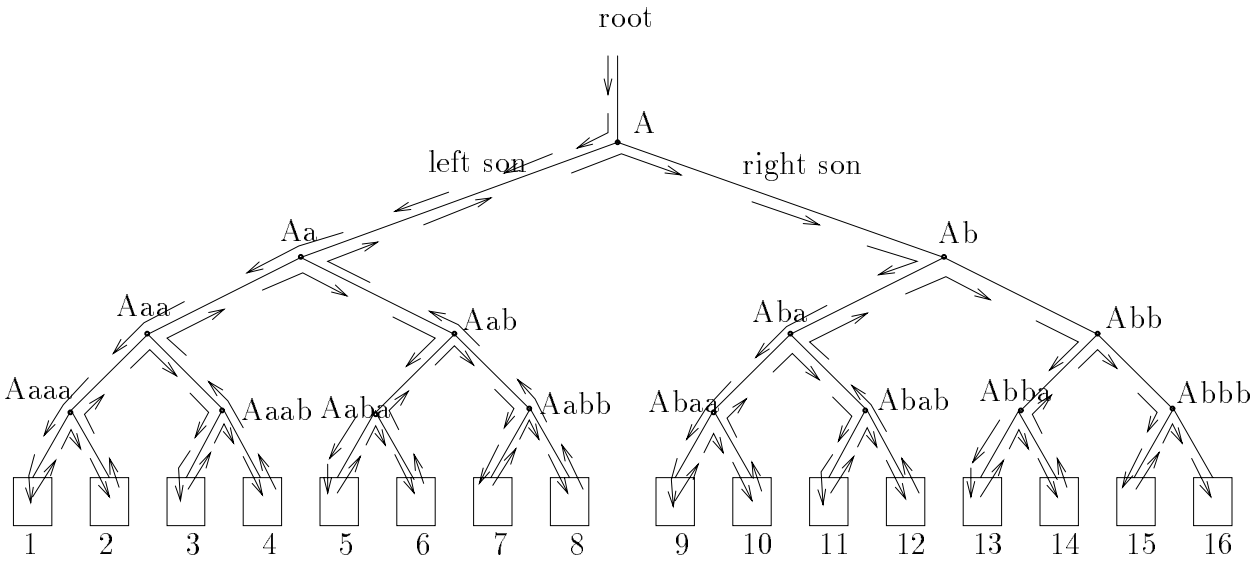

Figure 2: kd tree building process

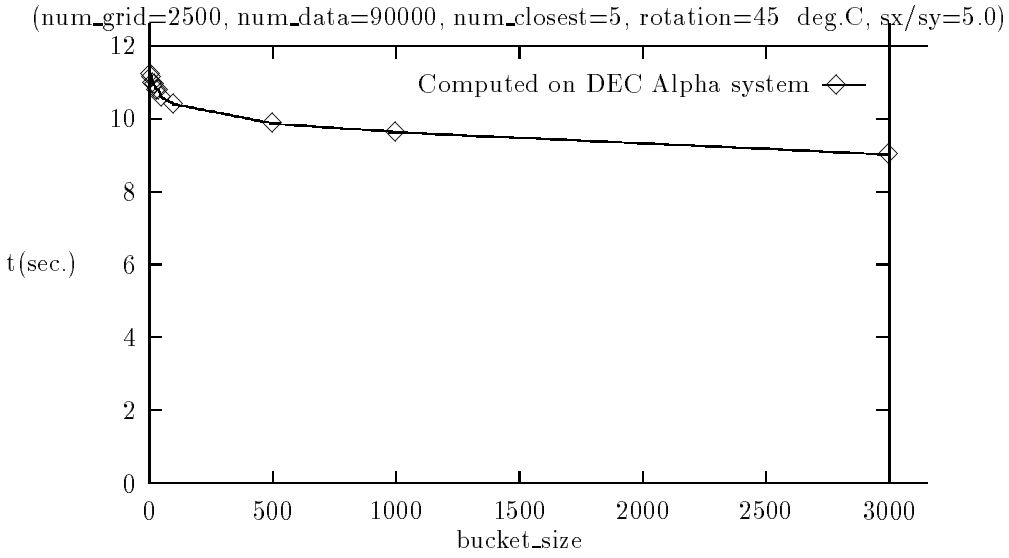

Figure 3: The Influence of Bucket Size on Tree Building Time

kdtree building time depends on the number of data point and the bucket size of the terminal buckets. The more the data point the longer the tree building time. For a given number of data point, the bigger the bucket size the less the tree building time. Fig.3 shows the influence of the bucket size on the tree building time and Fig.4 shows the influence of the number of data point on the tree building time.

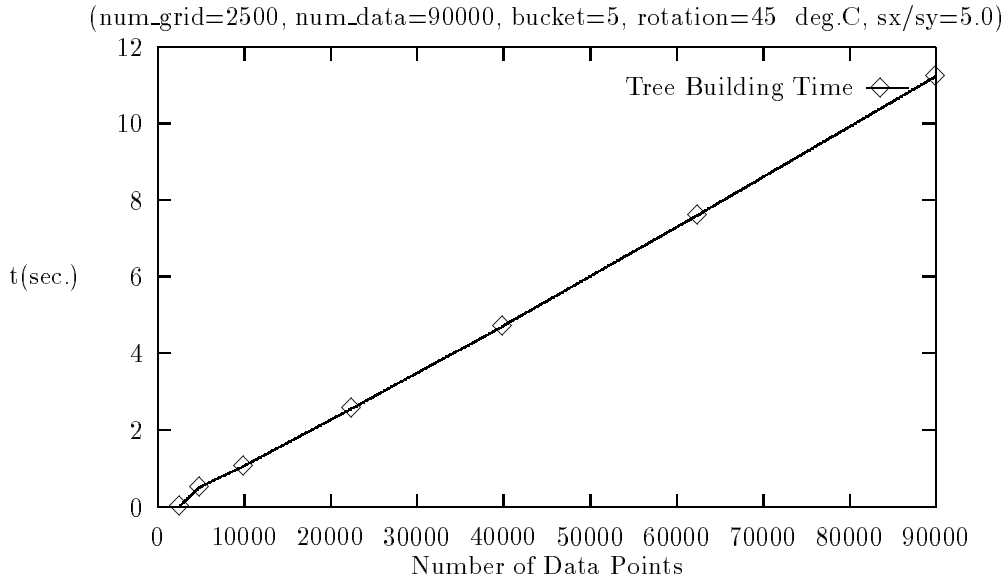

Figure 4: The Influence of Number of Data Point on Tree Building Time

Comparing Fig.3 and Fig.4 we can see that the influence of the number of data point is much more than the influence of the the bucket size. Therefore, for a large data set, the tree building may take a long time. This will affect the total computer time of the optimal estimation. However, The Program running time also dependis on the neighbour search and the number of nearest neighbors used in the optimization.

2.2 Nearest Neighbour Search

Once the tree is built, the closest data points that will be used for the estimation at this grid point are picked out from the terminal buckets by searching the tree with the top-down search scheme. For a given grid point, the search starts at the root of the tree. Each tree node contains an integer value holding the cut dimension and a float value holding the cut value. By comparing the coordinate of the grid point at the cut dimension with the cut value of this tree node, the search will know which branch to go for next level of tree node, it will compare the cut value of this tree node and go down further. This process is repeated until the terminal bucket that contains the potential closest data points is reached. Then some of the data points in this bucket will be selected as the nearest neighbours. If the coordinate of this grid point is closer to the cut value so that their distance is smaller than the maximum distance between the grid point and the data points in the bucket, or if the number of the data points in this bucket is less than the number of the closest data points specified, the neighbouring bucket will be searched. The search will stop if all the nearest neighbours are found. Otherwise it will go up to the parent tree node to search the other branches until all nearest neighbours are found.

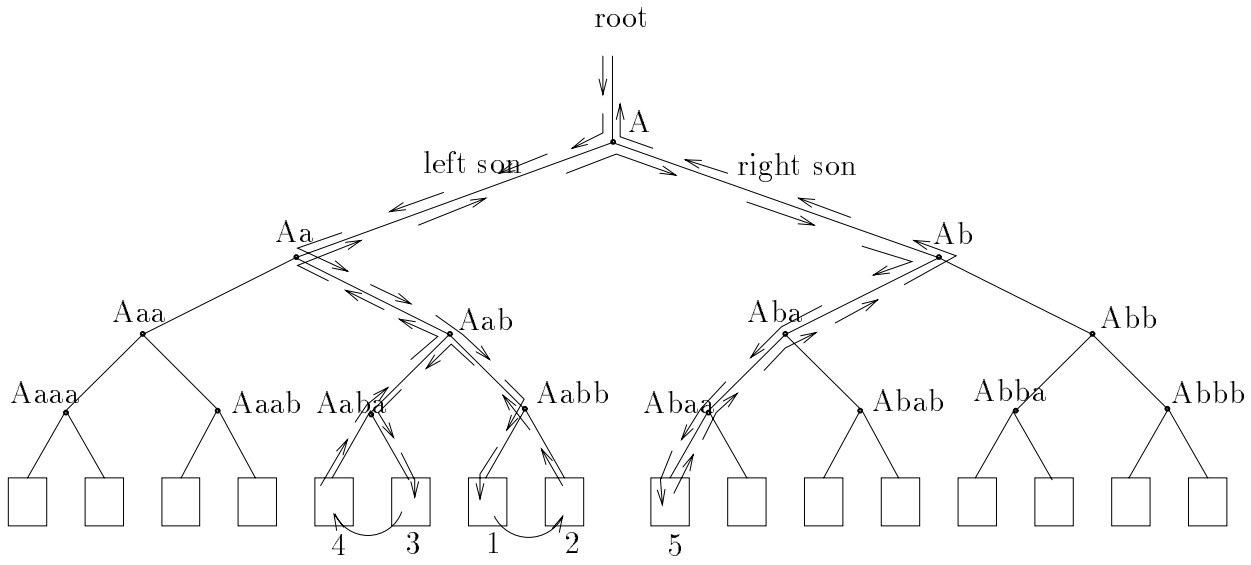

Figure 5: Kd Tree Nearest Neighbor Search Process

Figure [5] describes the nearest neighbour search process in a binary tree structure. This procedure assumes that the scales used in the search are the same as the scales used in building the tree.

In the latest oax6 version, the nearest neighbour search is based on the pseudo-distance calculated from local scales. If the coordinate is rotated and scaled, the nearest neighbour search algorithm implemented in the latest oax6 program is different from the previous versions. A new function "testcut" is added in to compute the the minimum possible local distance from a grid point and any point with coordinate cutdim having a value cutval. This not only avoid unecessary bucket searches

but also ensures all nearest neighbours can be found in the local scaled coordinates. Figure[6] shows the distribution of local scaled nearest neighbours.

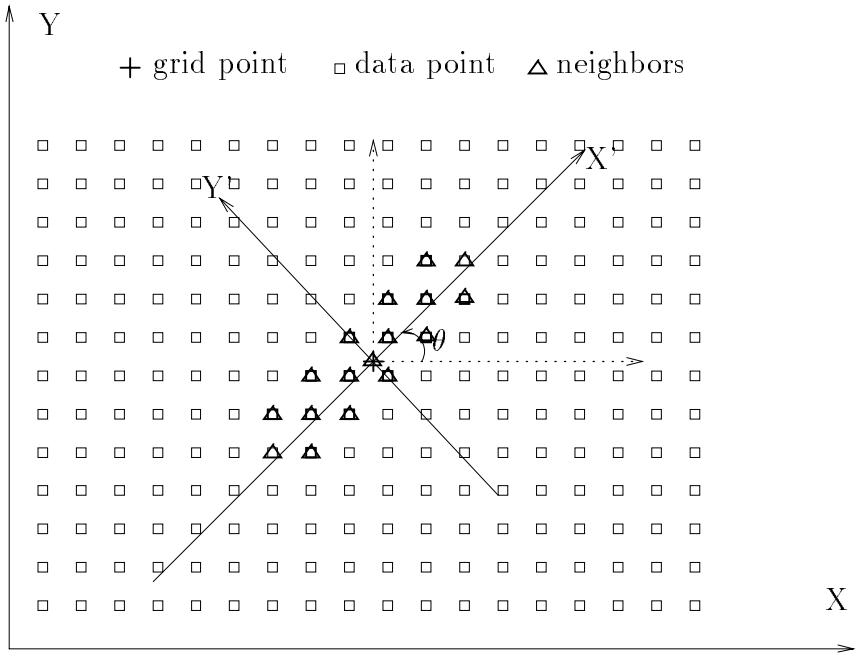

Figure 6: Nearest Neighbor Search and Local Scale

The kd-tree is created using global scales and unrotated coordinates. The tree is searched using local scales. The search algorithm for retrieving points from the kd-tree generally compares the difference between the cutdim-coordinate of the target point and the cutvalue with the maximum of the distances to all the candidate nearest neighbours already selected to determine (nndist[1]) if there could be any points closer than this maximum with cutdim-coordinate on the other side of the cut value. When the test involves either cutdim=0 or cutdim=1, the test using only the cutdim-coordinate could fail but there could still be points closer than the maximum because of the different scales in different directions. testcut computes the minimum possible (local scale) distance for points having the cutdim-coordinate equal to the cutval to test against nndist[1].
... ...
... ...
(the formulation of testcut algorithm and figure)
... ...
... ...

As the testcut algorithm only searches the buckets containing the points whose distance to the grid point is larger than the minimum possible local scaled distance not the maximum local scaled distance, it is much more efficient in the cases where the large local scales are involved. Figure[7]

and [8] shows the comparisions of the two search algorithms computed on a Pentium 90 system.

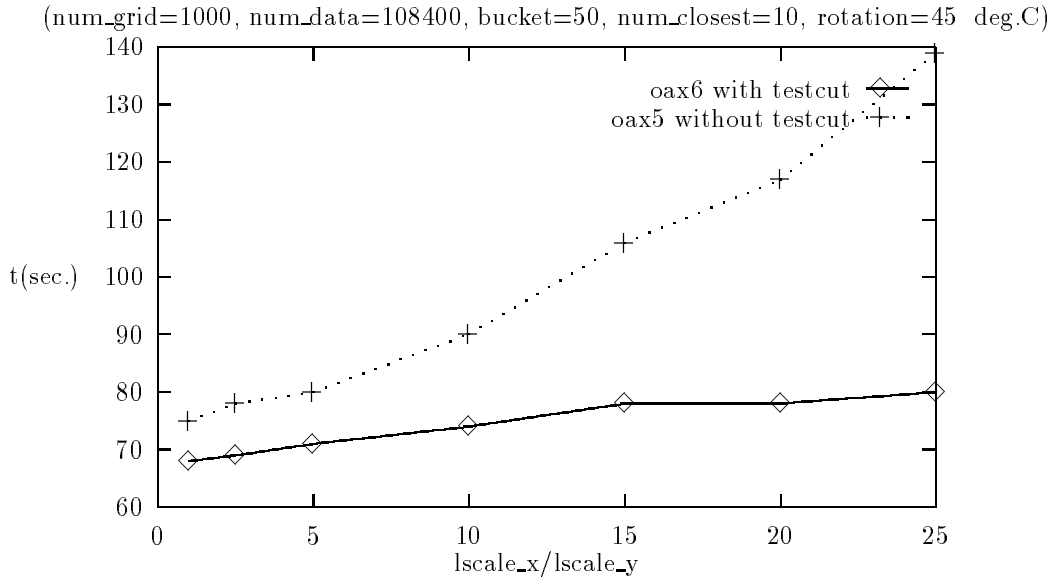

Figure 7: Local Scale Effect on Searching Time

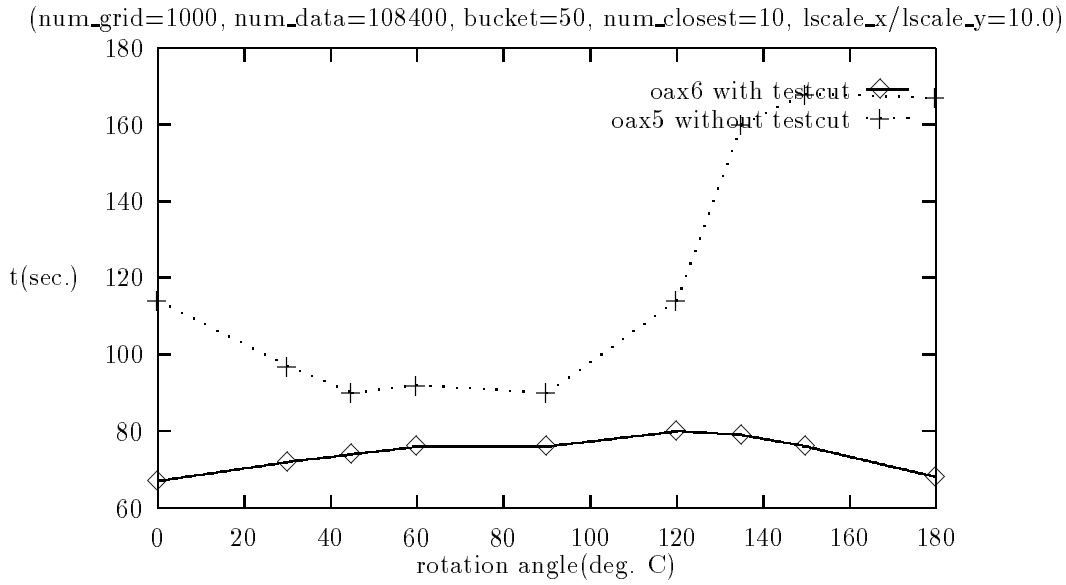

Figure 8: Local Rotation Effect on Searching Time

Figure [7] shows the local scale effect on the search speed. The search time increases very fast for previous search algdrithm as the local scale ratio increases. With the new search algorithm, however, The search time does not change much (slightly increases) when the local scale ratio increases.

Figure [8] shows the rotation effect on the search speed. From Figure [8], we can see that when the local scale ratio is fixed, the search speed of the previous search algorithm strongly depends on the rotation angle. While the new algorithm spends much less searching time and the search speed does not vary much as the rotation angle varies.

Above two examples demonstrate the efficiency of the new search algorithm. From the examples we know that this search algorithm is much more efficient than the previous search algorithm especially when the coordinate is rotated and the large local scale ratio is involved.

Figure[9] also presentes a searching time curve. The difference between Figure[9] and Figure [8] is that they use different data. A real arbitraryly distributed data was used in Figure[8]. Here a uniformly distributed data is used. The perpose of this figure is to show the effect of data distribution on the search time.

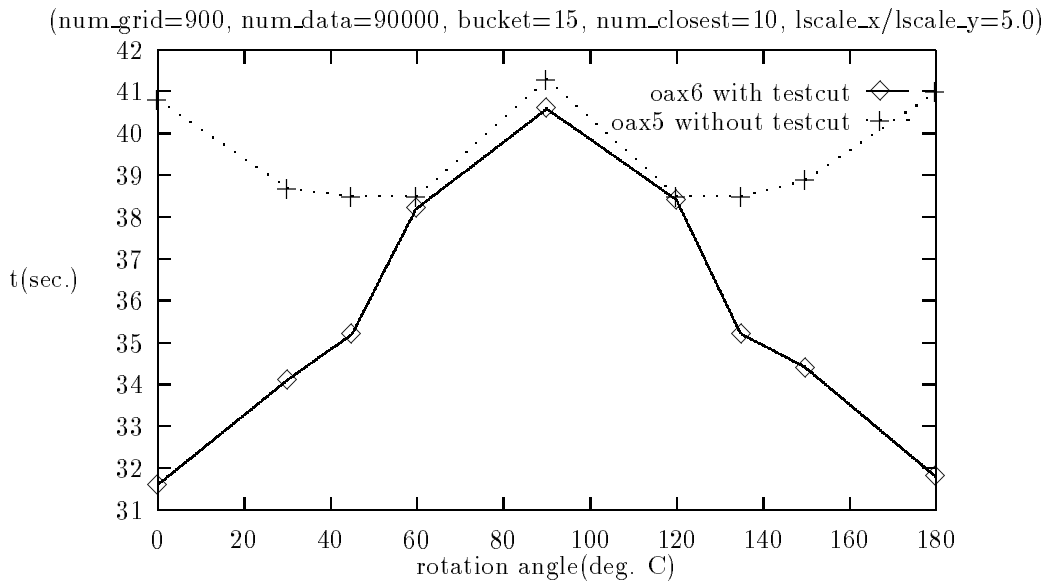

Figure 9: Data Distribution Effect on Searching Time

The results in Figure [9] were also computed on a Pentium 90 system. From Figure [9] we can see that the search time for the new algorithm icreases as the rotation angle increases from 0 degree to 90 degree and then decreases symmetrically as the rotation angle increases from 90 degree to 180 degree. The search time is significantly less than that of the previous search algorithm when the rotation angles are between 0-60 degrees and 120-180 degrees.

It is obvious that the search speed depends on the number of grid point. No mater which search algorithm is used, the more the grid point, the longer the search time. In addition to the number of grid points that affects the search speed, the bucket size and the number of nearest neighbours also influence the search speed. Normally, larger bucket size takes more time to search because all the data points in the bucket have to be checked. Larger number of nearest neighbours also leads to longer search time. But it is not true that smaller bucket size always result in less search time. There is an optimal bucket size value which depends on the number of the nearest neighbours. It is recommended that the bucket size be equal to the number of the nearest neighbours.

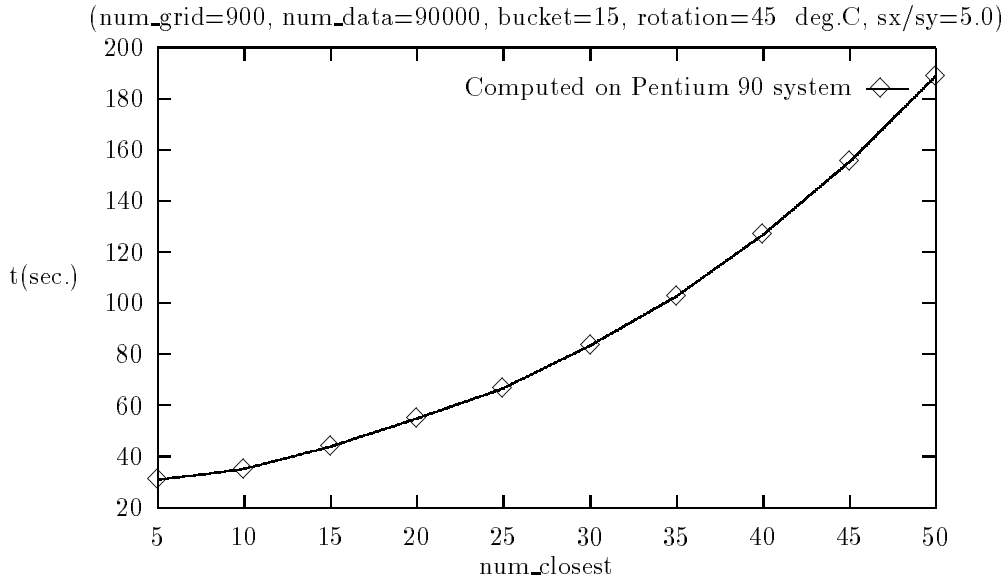

Figure 10: Num_closest and optimization speed

Figure [10] clearly shows the optimization time increases as the number of nearest neighbours increases. Here we use the word optimization time, which is because that the number of nearest neighbours not only affects the search but also affects the optimal estimation time and more importantly, the optimal estimation results. It is the number of the nearest neighbours that is critical for obtaining reliable estimation results. Theoretically, the more the better. However, There is not much improvement to the estimation result when the number of nearest neighbours exceed a certain value, on the contrary, the increase of this number would significantly increase the CPU time because of the inversion of a large size ovariance matrix will have to be involved. Therefore, the number of the nearest should be suitably selected. It is suggested to be any number ranged from 10 to 50.

The bucket size has nothing to do with the estimation results, but it affects the search speed. Figure [11] gives the result from a Pentium 90 computer computation. It shows the search and the tree building time varies with the bucket size. In this example, the number of closest is 15. We can see that when the bucket size equals to the num of closest, the search time takes its minimum value. That's why we suggest that an equal number bucket size and number of closest be used.

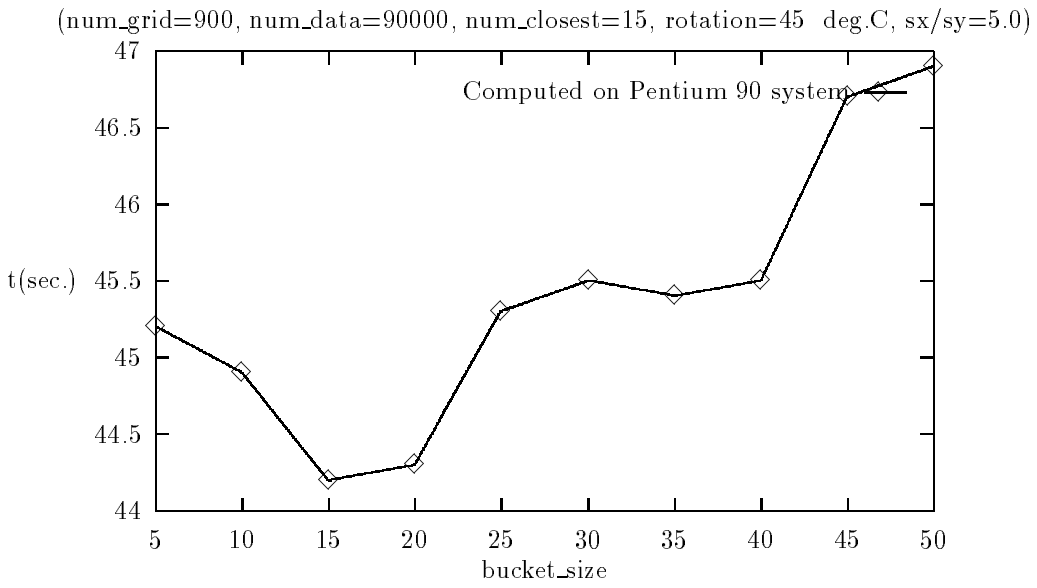

Figure 11: Bucket_size and optimization speed

As the number of nearest neighbours and the bucket size are not independent of each other. Figure [12] combines them together and shows a influence 3-D surface.

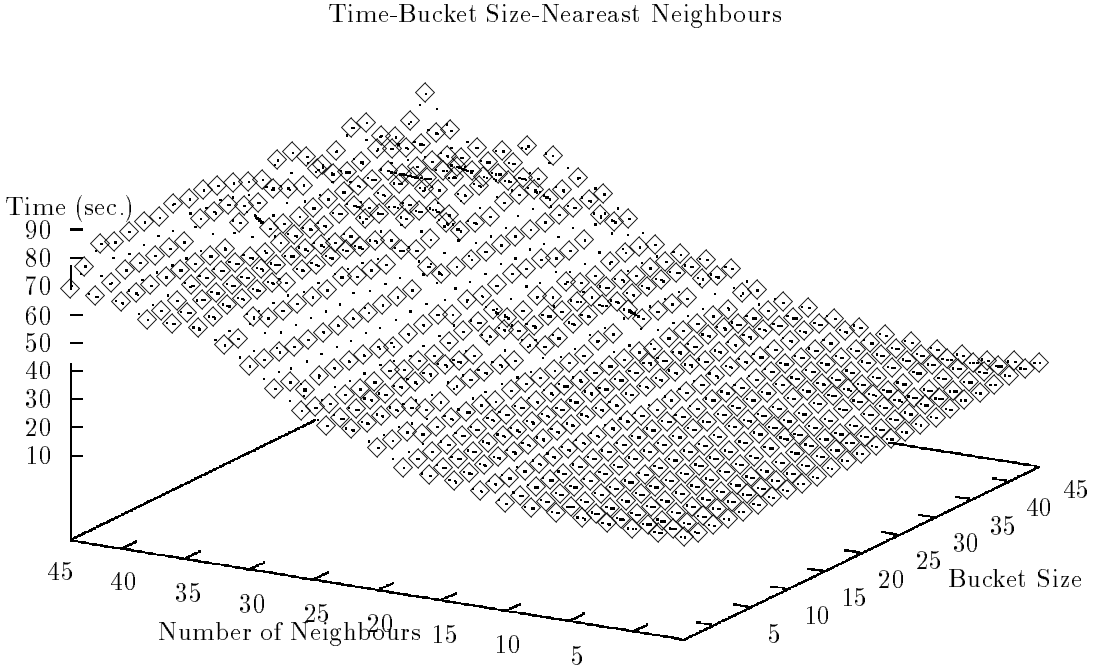

Figure 12: Num_Nearest , Bucket_size and optimization speed

2.3 Covariance Functions

oax uses certain assumptions about the statistical nature of the data. It uses an objective analysis method to interpolate/extrapolate data at grid points based on the values at a finite number of nearest neighbours.

The specific covariance model implemented in the code is of the form:

$$covariance(r) = e^{-r} \cdot (1 + r + \frac{r^2}{3}) \tag{1}$$

where r is a pseudo-distance calculated as

$$\sqrt{\sum_{i=1}^{n} (\frac{x_i - y_i}{a_i})^2} \tag{2}$$

where: a_i is the local scale factor of the ith independent coordinate x_i and y_i are the ith components of x and y respectively.

Other covariance models can be introduced only by modifying the source code.

This pseudo-distance controls the selection of nearest neighbours and the generation of weights (through the covariance model). The relative influence of lags in each of the independent variables is determined by the user-supplied local scales.

This built-in covariance function can be replaced by user specified covariance function in C library and F77 library functions.

2.4 Optimal Linear Estimation

Optimal linear estimation is the core of the oax program.
We suppose that the measured value Φ_r is the true point value plus some random noise

$$\Phi_r = \theta(x_r) + \epsilon_r \qquad r = 1, 2, ..., num\_closest \tag{3}$$

In order to estimate the value at point x (grid point) we assume that the estimated value is a linear combination of the measured values (data points):

$$\tilde{\theta}_x = \sum_{s=1}^{num\_closest} \alpha_{xs}\Phi_s \tag{4}$$

The coefficients α_{xs} are so determined that the expected value of the sum of squared errors is minimized. Two different estimates are possible, depending on the treatment of the mean value:

2.4.1 Anomaly

assumptions:

1. zero mean
$$\tilde{\theta}_x = 0 \tag{5}$$

2. known covariance function
$$\overline{\theta_x \theta_{x+\xi}} = F(\xi) \tag{6}$$
F is coded as a correlation matrix (covariance normalized by the covariance at zero separation)

3. errors are uncorrelated with one another and with the field
$$\overline{\epsilon_r \epsilon_s} = 0 \qquad (r \neq s) \tag{7}$$
$$\overline{\epsilon_r \theta_s} = 0 \tag{8}$$

4. known error variance E
$$\overline{\epsilon_r \epsilon_s} = E \qquad (r = s) \tag{9}$$
E is specified by the user as the dimensionless ratio:
$$NOISE = \frac{noise \;\; variance}{signal \;\; variance} \tag{10}$$

The optimal linear estimator for this is :

$$\hat{\theta}_x = \sum_{r=1}^{num\_closest} C_{xr}(\sum_{s=1}^{N} A_{rs}^{-1}\Phi_s) \tag{11}$$

where

$$A_r s = \overline{\Phi_r \Phi_s} = F(x_r - x_s) + E\delta_{rs} \tag{12}$$

is the covariance matrix and

$$C_{xr} = \overline{\Phi_r \Phi_x} = F(x - x_r) \tag{13}$$

is the covariance vector.

The estimated error variance is

$$\overline{(\theta_x - \theta_x)}^2 = C_{xx} - \sum_{r,s=1}^{N} C_{xr} C_{xs} A_{rs}^{-1} \tag{14}$$

The first term is the natural variation in the absence of any data. The second term measures the information provided by the data. Therefore, only the location of the data points and the knowledge of the covariance function and noise level E determine the error. The error output in the oax program is

$$\sqrt{\overline{(\theta_x - \theta_x)}^2} \tag{15}$$

The noise level E may be different for different locations, but all dependent variables at a given grid point are assigned the same relative noise level E (specified in the last column of data file). Therefore, in a multi-dependent variable estimation case, several separated runs might be needed if user want to apply different noise level to the dependent variables.

2.4.2 Estimated Mean

In the general case that the mean is unknown, the first assumption of the ANOMALY case does not apply. A revised estimate over the set of nearest neighbours can be derived as:

$$\hat{\theta}_x = \tilde{\theta} + \sum_r C_{xr} \{ \sum_s A_{rs}^{-1} (\Phi_s - \tilde{\theta}) \} \tag{16}$$

where θ is the estimated mean value

$$\tilde{\theta}_x = \frac{\sum_r C_{xr} \sum_s A_{rs}^{-1} \Phi_s}{\sum_{r,s} A_{rs}^{-1}} \tag{17}$$

The error variance can be expressed as:

$$\overline{(\theta_x - \hat{\theta}_x)}^2 = C_{xx} - \sum_{r,s} C_{xr} A_{rs}^{-1} C_{sx} + \frac{(1 - \sum_{r,s} C_{xs} A_{sr}^{-1})^2}{\sum_{r,s} A_{rs}^{-1}} \tag{18}$$

The first two terms have been explained in the ANOMALY case. The last term is the increase associated with the uncertainties of the estimated mean.

The error output in OAX is the square root of the error variance. It represents the relative estimation error for all dependent variables because the noise level is assumed to be same for all dependent variables. In order to obtain dimensional errors bounds (standard deviations) for each dependent variable, users may multiply the output error parameter by a user-supplied estimate of the standard deviation of the appropriate field.

3 Computer Program

3.1 Program Structure

The OA Programs can be divided into four groups:

1. Stand Alone C Version
2. C Library Version
3. F77 Library Version
4. AVS Module Version
5. KD Tree Programs

The OA computer programs are written in C (ANSI C Standard) language. The stand alone version (oax6) has been tested in a number of platforms and proved to be very stable. This version is designed to serve as an independent tool for users who want to a lot of mapping work. It is easy to use. User do not need to write program but just prepare the data file, grid file and deck file. The program will load the data and grids and perform the optimal estimation according to the instructions given in deck file and generate mapping result in .out file.

OA C library is designed for users who need to do optimal estimation within their application programs. In this case, one can call the OA C Library functions within his/her C program and directly apply the optimal estimation result in the application. The C Library functions have been tested in a number of platforms. It is machine independent.

OA F77 library is designed for users who need to do optimal estimation within their Fortran application programs. In this case, one can call the OA Fortran Library functions within his/her Fortran program and directly apply optimal estimation result in the application. The Fortran Library functions are based on the C Library functions and they are just an interpreter between C and Fortran. As each machine has different implementations of Fortran Call C. The Fortran Library is machine dependent. The F77 Library of OA program is only for SUN Spac 2000 system.

OA AVS Module is designed for the users who want to view the optimal estimation result graphically on computer screen. Two AVS modules (oax3d and oax2d) have been written based on the OA stand alone version program. AVS modules work on SUN and DEC systems but they have not been intensively tested for real data. The AVS modules allow user to control the bucket size, number of nearest neighbors and the global scales interactively.

The kd tree programs *tdstore* and *tdload* are the by-products of the oa programs. *tdstore* is for building kd tree and store the tree into a disk file either in ASCII or BINARY mode. *tdload* is for loading a previously built tree from the disk file in a corresponding mode.

3.2 Flow Chart of oax6 Program

All groups of OA programs are based on the OA stand alone program oax6. Therefore, we only give the details of the oax6 program structure. Figure 13 shows the program flow chart of oax6.c.

3.3 Program Distributions

All programs in OA packages are availble for users to down load from OA web site through Netscape or other browsers. The OA home page is currently located at the Technical University of Nova Scotia and the Bedford Institute of Oceanography. The packege is organized as follows:

3.3.1 OA Stand Alone C Version (oax6)

- Binary
  - BSD (oax6_BSD.tar ==> 120K)
  - DEC (oax6_DEC.tar ==> 180K)
  - HP (oax6_HP.tar ==> 150K)
  - IBM (oax6_IBM.tar ==> 143K)
  - LINX (oax6_linx.tar ==> 153K)
  - SGI (oax6_SGI.tar ==> 154K)
  - SUN (oax6_SUN.tar ==> 200K)
- Source
- Document

3.3.2 OA C Library (oax_clib)

- Binary
  - BSD (oaxc_lib_BSD.tar ==> 370K)
  - DEC (oaxc_lib_DEC.tar ==> 485K)
  - HP (oaxc_lib_HP.tar ==> 563K)
  - IBM
  - LINX (oax6_linx.tar ==> 153K)
  - SGI (oaxc_lib_SGI.tar ==> 422K)
  - SUN (oaxc_lib_SUN.tar ==> 471K)
- Source (oaxc_lib_src.tar ==> 451K)
- Document (oaxc_lib_doc.tar ==> 251K)

3.3.3 OA F77 Library (oax_flib)

- Binary
  - SUN (oaxf_lib_SUN.tar ==> 471K)
- Source (oaxf_lib_src.tar ==> 451K)
- Document (oaxf_lib_doc.tar ==> 251K)

3.3.4 OA AVS Module

- Binary
  - SUN (oax_2d.tar ==> 471K)
  - DEC (oax_3d.tar ==> 471K)
- Source (oax_2dsrc.tar oax_3dsrc.tar)
- Document (oax_module_doc.tar ==> 251K)

3.3.5 KD Tree Programs

- Binary
  - BSD (tdstore_BSD.tar, tdload_BSD.tar)
  - DEC (tdstore_DEC.tar, tdload_DEC.tar)
  - HP (tdstore_HP.tar, tdload_HP.tar)
  - IBM (tdstore_IBM.tar, tdload_IBM.tar)
  - LINX (tdstore_LIX.tar, tdload_LIX.tar)
  - SGI (tdstore_SGI.tar, tdload_SGI.tar)
  - SUN (tdstore_SUN.tar, tdload_SUN.tar)
- Source (tdstore_src.tar, tdload_src.tar)
- Document (kdtree_doc.tar ==> 251K)

The home machine for storing the OA packages are emeroald at BIO. The OA package is located at:
/usr4/objanal/OA

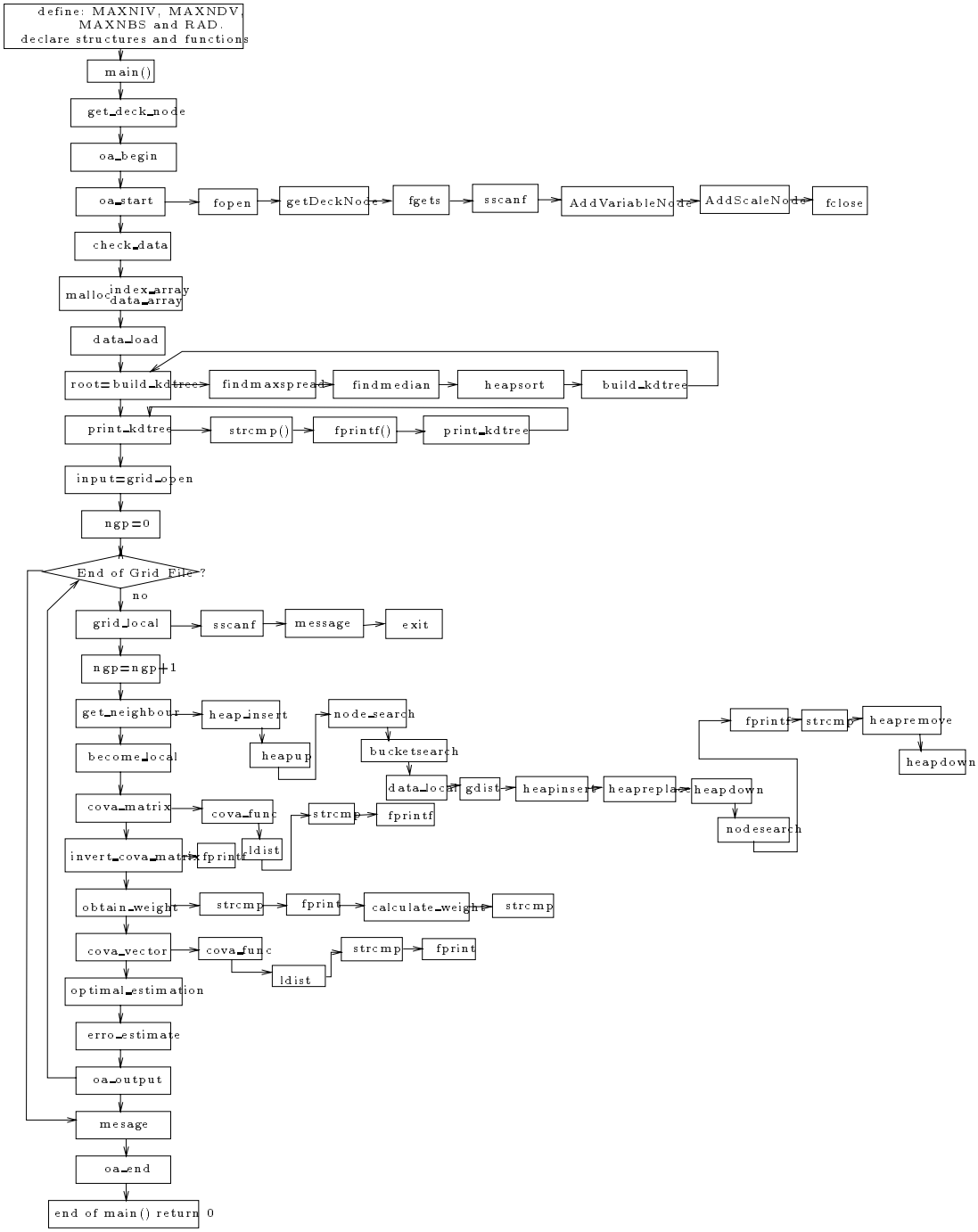

Figure 13: oax6 Program Flow Chart

4 User's Guide

4.1 OA Stand Alone Version oax6

4.1.1 How to Run the Program

To run this program, simply do the following operations:

1. prepare the input files:

   - deck file (example: test.deck)
   - data file (example: test.dat)
   - grid file (example: test.grid)

   The formats of the above files will be described later.

2. at command line, type:
   oax6 "deck file name" -[options] <Enter>
   where deck file name is the family name without .deck extension. The options include d1, d2, d3 and -time.

   Example: oax6 test -d2 <Enter>

   where the command Line Options

   d1: print the grid point and nearest neighbours

   d2: In addition to d1 printout, this debug option also print covariance matrix A, inverse covariance matrix ivA, covariance vector and weights.

   d3: only print kdtree nodes

   time: display and print the system time spent in the task.

   NOTE : The debug information will be printed in log file.

   !TIP! — For online help, type: oax6 <Enter>

3. OUTPUT Results

   - log file : filename.log (Automatic generation) Example: myfile.log
   - result : filename.out (Automatic generation) Example: myfile.out

4.1.2 File Format

The formats of deck file, data file and grid file are as follows:

Deckfile: test.deck

DEPENDENT temperature
INDEPENDENT km_east km_north pressure

GLOBAL_SCALES 1.0 1.0 1.0
DATAFILE test.dat
GRIDFILE test.grid
BUCKET 5
NUM_CLOSEST 3
METHOD EST_MEAN

NOTE: The sample files given above are only a special case. Users can specify their own choices by modifying the deckfile.

DEPENDENT : key word for dependent variables

temperature is the dependent variable name

INDEPENDENT : key word for independent variables

km_east, km_north and pressure are the independent variable names. Here only three independent variables are given. The user can give the independent variable names up to MAXNIV defined in include file.

GLOBAL SCALES : key word for scale factors

The scale factors are float numbers (greater than zero and less then infinity) corresponding to each independent variable. Here the scale factors are 1.0 for km_east, km_north, and pressure. User can specify the scale values for all independent variables.

DATAFILE: key word for data file name and path

test.dat is the data file that stores the data coordinates, the observations of dependent variables and their noise levels. The format of data file is :

0.25 0.25 0.0 100.0 0.1
0.50 0.50 0.0 -100.0 0.1
0.75 0.75 0.0 100.0 0.1

This is a three dimension case with one dependent variable. The first three columns hold the km_east, km_north and pressure coordinates, the fourth column holds the dependent variable values and the fifth column holds the noise level for each data point. The number of columns depends on the number of dependent and independent variables. For a k dimensional data coordinates with n dependent variables, there should be k columns of data plus n columns of observation in this file. But if the column number is more than the dimension number, the program only load the first k+n+1 columns. The number of the data point is 3 in this example.

GRIDFILE: key word for grid file name and path

test.grid is the grid file that stores the grid point coordinates. The format of grid file is :

0.25 0.25 0.0 0.0 1.0 1.0 1.0
0.50 0.50 0.0 30.0 1.0 1.0 1.0
0.75 0.75 0.0 90.0 1.0 1.0 1.0

This is a three dimension case. It depends on the number of independent variables. For a k dimensional data, there should be at least 2k+1 columns of data in this file. But if the column number is more than the dimension number, the program only load the first 2k+1 columns. The number of the grid point is 5 in this example. User can specify his own grid data. The number of grid points is independent of the number of data points.

We notice that the grid file here is different from the grid file of OA2.0. This is because of the new feature of this version. The local coordinate can be rotated by an angle at each grid point(the fourth column holds the angle of each grid point) and scalled by a factor(the last three columns hold the scaling factor at each grid point for three independent coordinates).

BUCKET : keyword for bucket size

bucket size is an integer number which specifies the maximum number of data points in the terminal bucket. Here the number is 2. User can specify this any number between 1 and NDATA.

NUM_CLOSET: keyword for number of nearest neighbours

NUM_CLOSET is an integer number which specifies the number of nearest neighbours. Here in this example, the number is 5. User can specify any number between 1 and 200.

METHOD : keyword for statistic model

EST_MEAN will use the zero mean in the optimization

Once the deck file and two data files are available, one can run the oax6 program and obtain the output results.

Example: oax6 test <Enter>

Here is the example of file test.out:

0.250000 0.250000 0.000000 34.441101 0.234984
0.500000 0.500000 0.000000 31.117872 0.185583
0.750000 0.750000 0.000000 34.441101 0.234984

The first three columns are the independent variables represent km_east, km_north and pressure. The fourth column is the optimal estimation result of the dependent variable temperature and the last column is the error of the estimation.

The log file will look like the follows:

File: test.log

This file contains the detailed debug information

```
************************************************
Welcome to the Optimal Analysis Program
(Version 6.0 , March, 1996)
************************************************
Creation Time: Tue Mar 7 15:01:53 1996

Reading Parameter File test.deck
————————————————
DEPENDENT temperature
INDEPENDENT km_east km_north pressure
GLOBAL_SCALE 1.000000 1.000000 1.000000
DATAFILE test.dat
GRIDFILE test.grid
deck->Bucket 5
NUM_CLOSEST 3
METHOD EST_MEAN

Number of data points read : 3

Number grid point read : 3
```

If the command line options are involved in, the log file will be different. For example:

oax6 test -d2 <Enter>

The log file will look like:

File: test.log

This file contains the detailed debug information

```
************************************************
Welcome to the Optimal Analysis Program
(Version 6.0 , March, 1996)
************************************************
Creation Time: Tue Mar 7 15:09:06 1996

Reading Parameter File test.deck
```

```
-------------------
DEPENDENT temperature
INDEPENDENT km_east km_north pressure
GLOBAL_SCALE 1.000000 1.000000 1.000000
DATAFILE test.dat
GRIDFILE test.grid
deck->Bucket 5
NUM_CLOSEST 3
METHOD EST_MEAN

Number of data points read : 3

Nearest Neighbour Search and Optimal Estimation:
================================================
-----------------------
At Grid Point ( 0.250000 0.250000 0.000000 )
-----------------------
Nearest neighbours are:
record 1: ( 0.250000 0.250000 0.000000 ) distance=0.000000
record 2: ( 0.500000 0.500000 0.000000 ) distance=0.353553
record 3: ( 0.750000 0.750000 0.000000 ) distance=0.707107

-- Covariance Matrix A
1.100000 0.979707 0.923899
0.979707 1.100000 0.979707
0.923899 0.979707 1.100000

--- Invert Covariance Matrix IvA
4.632940 -3.195022 -1.045620
-3.195022 6.600341 -3.195023
-1.045620 -3.195023 4.632941

---- Obtaining Weights
655.588806 -1311.177856 655.589111

---- Estimated Mean
57.724873

---- Covariance Vector C
1.000000 0.979707 0.923899
"test.log" 118 lines, 3127 characters

---- Estimate and Error
34.441101 0.234984
```

At Grid Point (0.500000 0.500000 0.000000)

Nearest neighbours are:
record 2: (0.500000 0.500000 0.000000) distance=0.000000
record 3: (0.750000 0.750000 0.000000) distance=0.353553
record 1: (0.250000 0.250000 0.000000) distance=0.353553

—— Covariance Matrix A
1.100000 0.979707 0.979707
0.979707 1.100000 0.923899
0.979707 0.923899 1.100000

—- Invert Covariance Matrix IvA
6.600339 -3.195022 -3.195022
-3.195022 4.632940 -1.045620
-3.195022 -1.045620 4.632940

—— Obtaining Weights
-1311.177612 655.588806 655.588806

—— Estimated Mean
57.724949

—— Covariance Vector C
1.000000 0.979707 0.979707

—— Estimate and Error
31.117872 0.185583

At Grid Point (0.750000 0.750000 0.000000)

Nearest neighbours are:
record 3: (0.750000 0.750000 0.000000) distance=0.000000
record 2: (0.500000 0.500000 0.000000) distance=0.353553
record 1: (0.250000 0.250000 0.000000) distance=0.707107

—— Covariance Matrix A
1.100000 0.979707 0.923899
0.979707 1.100000 0.979707
0.923899 0.979707 1.100000

—- Invert Covariance Matrix IvA
4.632940 -3.195022 -1.045620

-3.195022 6.600341 -3.195023
-1.045620 -3.195023 4.632941

—— Obtaining Weights
655.588806 -1311.177856 655.589111

—— Estimated Mean
57.724873

—— Covariance Vector C
1.000000 0.979707 0.923899

—— Estimate and Error
34.441101 0.234984

Number grid point read : 3

4.2 OA C Library

4.2.1 Description of OA C Library

Based on the oax6 and oax Clib 1.0, oax library 2.0 is available to the users. This library is designed for user's convenience to utilize the power of oax directly in their application programs. This version has the following new features:

- covariance function can be specified by the user
- new local search algorithm with testcut function

The following function calls are available from a C program:

- oax_init ———— initialize oax and build kd tree
- oax_nearest ——— find the nearest neighbours only
- oax_compute ——— search neighbour and perform optimal estimation
- oax_print_data —— print the data array
- oax_print_grid —— print the grid array
- oax_print_neighbour - print the nearest neighbours
- oax_print_result —- print the optimal estimation result
- oax_exit ———— free up the allocated memory and exit oax

To perform the optimal estimation, oax_init, oax_compute and oax_exit must be called. The other functions are optional. oax_init should be called first and the oax_exit function should always be called last.

The usage of the oax library functions are described in this document. Two example main programs oax_main1.c and oax_main2.c are also included for user's reference. To compile oax_main1.c and oax_main2.c, use the script files make_main1 and make_main2 or use Makefile to make them. See Makefile for details.

4.2.2 Compile OA C Library

Before compiling user's C program, OA C library must be made. To make OA C library, one can use the Makefile or the script file make_clib. For example, either of the following commands will make clib:

- make clib ¡Enter¿
- make_clib ¡Enter¿

4.2.3 User Supplied Covariance Model

To supply a covariance model, one need to:

1. Supply a C function named "usr_cov.c" that contains user supplied covariance function and return the covariance between two points pt1 and pt2.

2. edit the file usrdef.h (set the CTRL value to number 99).

3. include the file "usrdef.h" in your main function

4.2.4 Compiling User's Program

User's C program must be compiled and linked with the OA C library and user supplied covariance function. The following examples demonstrate how to make user's program.

- make_main1 (cc oax_main1.c cova.c lib_oax.a -o oax_main1 -lm)
- make_main2 (cc oax_main2.c cova.c lib_oax.a -o oax_main2 -lm)

4.2.5 Examples

Example 1: oax_main1.c

The first example demonstrates how to use oax c library by passing user's data array and grid array obtained from the calculation within user's program.

A. PROGRAM SOURCE CODE:
oax_main1.c

B. COMPILING THE MAIN PROGRAM:
make_main1 <Enter>

C. RUNNING THE PROGRAM:
oax_main1 <Enter>

D. PROGRAM OUTPUT:
oax_print_data():
NDATA=25 num_indep=2 num_dep=2 Noise=0.100000
0.000000 0.000000 13.000000 12.000000 0.100000
0.000000 1.000000 14.000000 11.000000 0.100000
0.000000 2.000000 15.000000 10.000000 0.100000
0.000000 3.000000 16.000000 9.000000 0.100000
0.000000 4.000000 17.000000 8.000000 0.100000
1.000000 0.000000 18.000000 7.000000 0.100000
1.000000 1.000000 19.000000 6.000000 0.100000
1.000000 2.000000 20.000000 5.000000 0.100000
1.000000 3.000000 21.000000 4.000000 0.100000

```
1.000000 4.000000 22.000000 3.000000 0.100000
2.000000 0.000000 23.000000 2.000000 0.100000
2.000000 1.000000 24.000000 1.000000 0.100000
2.000000 2.000000 25.000000 0.000000 0.100000
2.000000 3.000000 26.000000 -1.000000 0.100000
2.000000 4.000000 27.000000 -2.000000 0.100000
3.000000 0.000000 28.000000 -3.000000 0.100000
3.000000 1.000000 29.000000 -4.000000 0.100000
3.000000 2.000000 30.000000 -5.000000 0.100000
3.000000 3.000000 31.000000 -6.000000 0.100000
3.000000 4.000000 32.000000 -7.000000 0.100000
4.000000 0.000000 33.000000 -8.000000 0.100000
4.000000 1.000000 34.000000 -9.000000 0.100000
4.000000 2.000000 35.000000 -10.000000 0.100000
4.000000 3.000000 36.000000 -11.000000 0.100000
4.000000 4.000000 37.000000 -12.000000 0.100000

oax_print_grid():
NGRID=10 num_indep=2
1.500000 0.500000 35.000000 3.000000 4.000000
0.500000 1.500000 35.000000 4.000000 5.000000
-0.500000 2.500000 35.000000 5.000000 6.000000
-1.500000 3.500000 35.000000 6.000000 7.000000
-2.500000 4.500000 35.000000 7.000000 8.000000
-3.500000 5.500000 35.000000 8.000000 9.000000
-4.500000 6.500000 35.000000 9.000000 10.000000
-5.500000 7.500000 35.000000 10.000000 11.000000
-6.500000 8.500000 35.000000 11.000000 12.000000
-7.500000 9.500000 35.000000 12.000000 13.000000

oax_nearest():
Nearest Neighbour Search —>

oax_print_neighbour():
NGRID=10 num_indep=2 num_nearest=5

grid point 1
-----------

neighbour 1: 2.000000 0.000000
neighbour 2: 1.000000 1.000000
neighbour 3: 1.000000 0.000000
neighbour 4: 2.000000 1.000000
neighbour 5: 1.000000 2.000000

grid point 2
```

---

neighbour 1: 0.000000 2.000000
neighbour 2: 1.000000 1.000000
neighbour 3: 1.000000 2.000000
neighbour 4: 0.000000 1.000000
neighbour 5: 1.000000 0.000000

.....

.....

.....

grid point 9

---

neighbour 1: 0.000000 4.000000
neighbour 2: 0.000000 3.000000
neighbour 3: 1.000000 4.000000
neighbour 4: 0.000000 2.000000
neighbour 5: 1.000000 3.000000

grid point 10

---

neighbour 1: 0.000000 4.000000
neighbour 2: 0.000000 3.000000
neighbour 3: 1.000000 4.000000
neighbour 4: 0.000000 2.000000
neighbour 5: 1.000000 3.000000

oax_compute():
Covariance Model:—>Default Covariance Model

oax_print_result():
NGRID=10 num_dep=2
20.850103 4.149900 0.147265
17.159758 7.840224 0.145327
16.130186 8.869818 0.159285
17.344471 7.655538 0.215957
17.479303 7.520708 0.267987
17.594402 7.405611 0.316531
17.687315 7.312688 0.357996
17.761311 7.238686 0.392803
17.820366 7.179617 0.422077
17.867933 7.132064 0.446876

Example 2: oax_main2.c

The second example demonstrates how to use oax c library by passing user's data array and grid array obtained from reading the files.

Also, in this example, an user specified covariance model is used to replace the default covariance model built in the program.

A. PROGRAM SOURCE CODE:
oax_main2.c

B. EDIT FILE "usrdef.h"
in "usrdef.h", set Control variable to 99 as follows:

#Define CTRL 99

File "usrdef.h" must be included in oax_main2.c

C. PROVIDE A C FUNCTION "usr_cov.c" FOR CALCULATING COVARIANCE
The function "usr_cov.c" should be in the following format:

```
/*=====================================
Function cov_fun()
Purpose: Calculate covariance function between two points
pt1 and pt2 using the usr supplied model.
Input : r – local scaled distance between pt1 and pt2.
Return: x – covariance function
=====================================*/
#include "oax_lib.h"
float cov_fun(r)
float r;

float x;
x=exp(-r)*(float)(1.+r+r*r/3.);
return x;
```

Note: the name of the user supplied covariance function must be called "cov_fun(r)" with a parameter r.

D. COMPILING THE PROGRAM
make_main2 <Enter>

E. RUNNING THE PROGRAM

oax_main2 <Enter>

F. PROGRAM OUTPUT

```
oax_print_data():
NDATA=9 num_indep=2 num_dep=2 Noise=0.100000
0.000000 0.000000 13.000000 12.000000 0.100000
0.000000 1.000000 14.000000 11.000000 0.100000
0.000000 2.000000 15.000000 10.000000 0.100000
1.000000 0.000000 16.000000 9.000000 0.100000
1.000000 1.000000 17.000000 8.000000 0.100000
1.000000 2.000000 18.000000 7.000000 0.100000
2.000000 0.000000 19.000000 6.000000 0.100000
2.000000 1.000000 20.000000 5.000000 0.100000
2.000000 2.000000 21.000000 4.000000 0.100000

oax_print_grid():
NGRID=2 num_indep=2
1.500000 0.500000 35.000000 3.000000 4.000000
0.500000 1.500000 35.000000 4.000000 5.000000

oax_nearest():
Nearest Neighbour Search —>

oax_print_neighbour():
NGRID=2 num_indep=2 num_nearest=3

grid point 1
---------------
neighbour 1: 1.000000 1.000000
neighbour 2: 2.000000 0.000000
neighbour 3: 1.000000 0.000000

grid point 2
---------------
neighbour 1: 1.000000 1.000000
neighbour 2: 0.000000 2.000000
neighbour 3: 1.000000 2.000000

oax_compute():
Covariance Model:—>User Supplied Covariance Model

oax_print_result():
NGRID=2 num_dep=2
17.410322 7.589689 0.187811
```

16.621202 8.378800 0.185626

4.3 OA F77 Library

4.3.1 Descriptions of OA F77 Library

Based on the oax F77 library 1.0 and oax Clib 2.0, oa F77 library 2.0 is available to the users. This library is designed for user's convenience to utilize the power of oa directly in their Fortran application programs. This version has the same features as the oa C library:

- covariance function can be specified by the user
- new local search algorithm with testcut function

The following function calls are available from a C program:

- oax_finit ———— initialize oax and build kd tree
- oax_fnearest ——— find the nearest neighbours only
- oax_fcompute ——— search neighbour and perform optimal estimation
- oax_fprint_data —— print the data array
- oax_fprint_grid —— print the grid array
- oax_fprint_neighbour - print the nearest neighbours
- oax_fprint_result —- print the optimal estimation result
- oax_fexit ———— free up the allocated memory and exit oax

To perform the optimal estimation, oax_finit, oax_fcompute and oax_fexit must be called. The other functions are optional. oax_finit should be called first and the oax_fexit function should always be called last.

The usage of the oax library functions are described in this document. Two example main program oax_main1.f and oax_main2.f are also included for user's reference. To compile oax_main1.f and oax_main2.f, use the script files make_main1 and make_main2.

4.3.2 Compile OA F77 Library

Before compiling user's Fortran program, OA F77 library must be created. To compile OA F77 library, one can use the script file make_libf.

- make_libf ¡Enter¿

4.3.3 User Supplied Covariance Model

To supply a covariance model, one need to:

1. Supply a C function named "usr_cov.c" that contains user supplied covariance function and return the covariance between two points pt1 and pt2.
2. edit the file usrdef.h (set the CTRL value to number 99).

4.3.4 Compiling User's F77 Main Program

User's F77 program must be compiled and linked with OA C library , OA F77 library and user defined covariance function. This can be done by using the make script files. The script files will generate the object file of user defined covariance function first and then compile the Fortran main programs and link the clib, flib and usr_cov.o to generate the exceutable files.

- make_main1
  - cc -c cova.c -lm
  - f77 oax_main1.f libf_oax.a lib_oax.a cova.o -o oax_fmain1 -lm
- make_main2
  - cc -c cova.c -lm
  - f77 oax_main2.f libf_oax.a lib_oax.a cova.o -o oax_fmain2 -lm

4.3.5 Examples

Example 1: oax_main1.f

The first example demonstrates how to use oax c library by passing user's data array and grid array obtained from the calculation within user's program.

A. PROGRAM SOURCE CODE:
oax_main1.f

B. COMPILING THE MAIN PROGRAM:
make_main1 <Enter>

C. RUNNING THE PROGRAM:
oax_main1 <Enter>

D. PROGRAM OUTPUT:
oax_print_grid():
NGRID=121 num_indep=2

oax_print_data():
NDATA=201 num_indep=2 num_dep=1

oax_print_result():
NGRID=121 num_dep=1
0.985205 0.063577
3.999977 0.256678
7.999991 0.484187
11.999938 0.680563
16.000000 0.843731

20.000031 0.975958
24.000086 1.081110
28.000095 1.163445
...
...
187.998566 0.680560
191.999741 0.484180
195.997665 0.256669
199.014694 0.063454
0.000000 0.000000 0.985205 0.0636
0.000000 0.100000 3.999977 0.2567
0.000000 0.200000 7.999991 0.4842
0.000000 0.300000 11.999938 0.6806
0.000000 0.400000 16.000000 0.8437
...
...
1.000000 0.700000 187.998566 0.6806
1.000000 0.800000 191.999741 0.4842
1.000000 0.900000 195.997665 0.2567
1.000000 1.000000 199.014694 0.0635

Example 2: oax_main2.f

The second example demonstrates how to use oax c library by passing user's data array and grid array obtained from reading the files.

Also, in this example, an user specified covariance model is used to replace the default covariance model built in the program.

A. PROGRAM SOURCE CODE:
oax_main2.f

B. EDIT FILE "usrdef.h"
in "usrdef.h", set Control variable to 99 as follows:

#Define CTRL 99

C. PROVIDE A C FUNCTION "usr_cov.c" FOR CALCULATING COVARIANCE
The function "usr_cov.c" should be in the following format:

/*==========
Function cov_fun()
Purpose: Calculate covariance function between two points
pt1 and pt2 using the usr supplied model.
Input : r – local scaled distance between pt1 and pt2.

```
Return: x – covariance function
=======================================*/
#include "oax_lib.h"
float cov_fun(r)
float r;

float x;
x=exp(-r)*(float)(1.+r+r*r/3.);
return x;
```

D. COMPILING THE MAIN PROGRAM:

make_main2 <Enter>

E. RUNNING THE PROGRAM:

oax_main2 <Enter>

D. PROGRAM OUTPUT:

```
oax_print_grid():
NGRID=1 num_indep=2

oax_print_data():
NDATA=4 num_indep=2 num_dep=1

oax_print_neighbour():
NGRID=1 num_indep=2 num_nearest=4

grid point 1
-------------

neighbour 1: 0.000000 0.000000
neighbour 2: 1.000000 0.000000
neighbour 3: 1.000000 1.000000
neighbour 4: 0.000000 1.000000

oax_print_result():
NGRID=1 num_dep=1
14.999907 0.053696

0.500000 0.500000 14.9999 5.36959E-02
```

4.4 OA kd-tree Store Program

tdstore.c is an independent program for building a kd-tree from given data points and store the tree in a file. To be consistent with the oax program, tdstore.c takes deck file. The deck file has the same foemat as that used in oax program except that the DEPENDENT key word is not needed here. An example of deck file is given below:

```
INDEPENDENT x y
SCALES 1.0 1.0
DATAFILE test.dat
GRIDFILE test.grid
BUCKET 5
NUM_CLOSEST 5
```

INDEPENDENT : key word for independent variables

x and y are the independent variable names. Here only two independent variables are given. The user can give the independent variable names up to MAXNIV defined in include file.

SCALES : key word for scale factors

The factors are float numbers corresponding to each independent variable. Here the scale factors are 1.0 for both x and y. User can specify the scale values for all independent variables.

DATAFILE: key word for data file name and path

test.dat is the data file that stores the data coordinates. The format of data file is :

```
0.0 0.0
0.0 1.0
0.0 2.0
0.0 3.0
1.0 0.0
1.0 1.0
1.0 2.0
1.0 3.0
2.0 0.0
2.0 1.0
2.0 2.0
2.0 3.0
3.0 0.0
3.0 1.0
3.0 2.0
3.0 3.0
```

This is a two dimension case. It depends on the number of independent variables. For a k dimensional data, there should be at least k columns of data in this file. But if the column number is more than the dimension number, the program only load the first k columns. The number of the data point is 16 in this example.

GRIDFILE: key word for grid file name and path

Grid file is not necessary for the kd tree stroe program. It is useful only when the user wants to search the nearest neighbours to check the loaded kd tree in tdload program.

BUCKET : keyword for bucket size

bucket size is an integer number which specifies the maximum number of data points in the terminal bucket. Here the number is 5. User can specify this any number between 1 and NDATA.

NUM_CLOSEST: keyword for number of nearest neighbours

NUM_CLOSEST is an integer number which specifies the number of nearest neighbours. Here in this example, the number is 5. User can specify any number between 1 and NDATA. Once the deck file and two data files are available, it is ready to run the program tdtree or butree to build kd-tree and find the nearest neighbours. To do this, one can simply type the following command:

tdstore test

Here test is the family name of the deck file test.deck. The program will start running and generate an output file test.tree that contains the stored kd tree.

Here is the example of file test.tree:

1 1 0 0 8 1.500000 8 8
0 0 0 0 4 0.500000 4 4
0 0 0 8 4 2.500000 12 4
1
3
2
0
6
4
5
7
8
9
10
11
13

12
14
15

A log file test.log will also be generated. As we can see that if the data file is very large, then the kd tree storage file will be very large. In order to save space and time, tdstore can also store the kd tree in binary mode. If a "-b" option is used

tdstore test -b

then the output file test.out will be a binary file. This will save a lot of space when a large kd tree is built.

If the user did not run the program properly, the warning message or error message will appear on screen to remind the user.

4.5 OA kd-tree Load Program

kd tree load program tdload.c is an independent program that loads the kd tree built by tdstore program. It uses the same deck file as the tdstore program. There are also two modes: ascii mode and binary mode.

If the kd tree was built in ascii mode, the tdload should also use ascii mode

tdload test

If the kd tree was built in binary mode, the tdload should also use binary mode

tdload test -b

Note: tdload program also performes the nearest neighbour search. This is checking the loaded kd tree and demonstraing the user how to use the loaded kd tree.

4.6 OA AVS Modules

4.6.1 Module Descriptions

Two OA AVS modules oax_2d and oax_3d have been developed. By using AVS modules, user can interactively change the global scales, bucket size and the number of nearest neighbours. The AVS module will generate AVS field file and the optimal estimation results can be displayed directly after the computation is done.

oax_2d will generate a 2D fileld file and oax_3d will generate a 3D field file for AVS geometry viewer to display the results.

AVS modules are developed based on the oax4 and oax5 programs. The modules work for SUN and DEC systems. There were no extensive tests to both modules.

4.6.2 Compile the Modules

To compile the AVS modules, use the Makefile.oax_2d and Makefile.oax_3d by typing the following commands:

make -f Makefile.oax_2d ¡Enter¿ (compile oax_2d module)

make -f Makefile.oax_3d ¡Enter¿ (compile oax_3d module)

4.6.3 Module Networks

After compiling the modules, use the AVS module tool in the Network Editor to read the compiled modules oax_2d and oax_3d. They belong to the Filter type. Then use the Network tool to read the network oax_2d.net or oax_3d.net. These two example networks contain the names and paths of the module input field files. For example, in oax_2d.net, it will need to read two field files: 100m_data.fld and 100m_mesh.fld. The data file 100m06.dat is specified in 100m_data.fld and the grid file 100m_line.mesh is specified in 100m_mesh.fld.

4.6.4 Running the Modules

Once the network is loaded into the workspace in the Network Editor, the AVS module is fired to run. The input parameter menus will be pop up. The user have the choices to change the input field file names, set new values for all available input parameters. Every time when the user reset the parameters, the AVS module will rerun from the begining. The ranges for the input parameters are the same as required in oax program.

If the module runs sucessfully, the optimal estimation results will be displayed through the AVS geometry viewer.

4.6.5 Examples

Here are two examples of the AVS output from the oax_2d and oax_3d modules.

5 Applications

oax program has been used by a number of people at BIO. Here we give an example of the application of oax6 program.

6 Discussions

- OA AVS modules should be modified based on the latest oa C library functions.
- The kd tree programs are still based on the previous version of oa programs.
- The Matrix inversion function in oa C library is different from the one used in oax6 program.
- In the stand alone oax6 program, only the default covariance function can be used.
- On line www OA package and hypertext documentations.

7 Acknowledgement

The authors would like to thank the PERO project funding support. Thanks are also extend to ... ...

BIBLIOGRAPHY

1. Abdel, Alim O., Zaki N., and Maling, G.C. Jr (ed.) ,"Correlation between subjective and objective analysis of noise. ",INTER-NOISE-89. 1989, vol. 2, pp. 913-916.

2. Abdullah,-I. and Yean,-Y.S. ,"An objective analysis of the POLYMODE Local Dynamics Experiment. Part 1: General formalism and statistical model selection. ",J.-PHYS.-OCEANOGR. 1986. vol. 16, no. 3, pp. 483-504.

3. Abd-Elmotaal-Hussein ,"Optimal estimation of spatially variable recharge and transmissivity fields under steady-state groundwater flow, Part 1, Theory. ",Bulletin-Geodesique. 66. (4). p. 325-335.

4. Agterberg-Frederik-P ,"On the condition number of covariance matrices in kriging, estimation, and simulation of random fields. ",Anonymous. Joint meeting, Geological Association of Canada, Mineralogical Association of Canada–Reunion annuelle conjointe, Association geologique du Canada, Association mineralogique du Canada.

5. Ababou-Rachid, Bagtzoglou-Amvrossios-C, Wood-Eric-F ,"Forced libration of Mercury and geodetic determination of a possible fluid core. ",Mathematical-Geology. 26. (1). p. 99-133.

6. Aitchison-J ,"The statistical analysis of compositional data. ",Cox, D. R., Hinkley, D. V., Rubin, D., Silverman, B. W. Monographs on statistics and applied probability. 416 p.

7. Alaka Mikhail A. and Elvander Robert C., "Optimum Interpolation From Observations of Mixed Quality", Monthly Weather Review, Vol.100, No.8, pp. 612-624, 1972.

8. Alfaro-M, Miguez-F ,"Optimal interpolation using transitive methods. ",NATO-Adv.-Study-Inst.-Ser.,-Ser.-C,-Math.-Phys.-Sci. 24. Advanced geostatistics in the mining industry. p. 91-99.

9. Andryushchenko-A-A and Belyayev-V-I, "Study of the statistical structure of the temperature and salinity fields of the Black Sea with a view to their objective analysis", Atmos. Oceanic Phys. (Engl. Ed.). Vol. 8, No. 9, p. 582-584, illus, 1972.

10. Arabadzhi-M-S, Vasil'-yev-Yu-M, Mil'-nichuk-V-S ,"Construction of structure maps by correlation analysis in the Caspian basin. ",Explor. Geophys. Vol. 50, p. 107-113, illus. (incl. sketch map), 1969.

11. Arhan,-M. and De-Verdiere,-A.C. ,"Objective analysis and assimilation techniques used for the production of FGGE IIIb analyses. ",J.-PHYS.-OCEANOGR. 1985. vol. 15, no. 2, pp. 153-170.

12. Armstrong-Margaret, Starks-Thomas-H, Sparks-Allen-R ,"Estimation of the generalized covariance function, II, A response surface approach by Thomas H. Starks and Allen R. Sparks, discussion and reply. ",Mathematical-Geology. 19. (8). p. 785-792.

13. Bakr-A-A, Gelhar-L-W, Gutjahr-A-J, MacMillan-J-R ,"Empirical determination of the gravity anomaly covariance function in mountainous areas. ",Int.-Geol.-Congr.-Abstr.-Congr.-Geol.-Int.,-Resumes. (26 Vol. 2). p. 842.

14. Banister-C ,"Developing a census data-mapping package for Sirius. ",Gardiner, V., Unwin, D. J. Thematic mapping using microcomputers. Univ. Leicester, Dep. Geogr., Leicester, United-Kingdom. Computers-and-Geosciences. 11. (3). p. 301-303.

15. Barendregt-L-G ,"The estimation of the generalized covariance when it is a linear combination of two known generalized covariances. ",Water-Resources-Research. 23. (4). p. 583-590.

16. Baussus von Luetzow H. ,"Improved spatial gravity anomaly covariances. ",Eos-(Am.-Geophys.-Union,-Trans.). 58. (6). p. 374.

17. Bayazit-M, Obeysekera-J-T-B, Yevjevich-V ,"Covariance between subsample mean and variance as related to storage variables. ",Journal-of-Hydrology. 78. (1-2). p. 137-150.

18. Bennett, J. and Goulter, I. ,"The North Atlantic Current and its associated eddy field Southeast of Flemish Cap. ",GEOJOURNAL. 1989. vol. 18, no. 2, pp. 213-220

19. Bennett, J, Goulter, I ,"The use of multiobjective analysis for comparing and evaluating environmental and economic goals in wetlands managements. ",GEOJOURNAL. vol. 18, no. 2, pp. 213-220, 1989.

20. Bentley-L-R ,"Spatial analysis and correlation of geological map patterns. ",Anonymous. Geological Society of America, 1993 annual meeting. Abstracts-with-Programs-Geological-Society-of-America. 25. (6). p. 108

21. Bergamasco A., Teatini P. and Carbognin L., "A critical comparison of kriging and objective analysis", Il-Nuovo-Cimento-della-Societa-Italiana-di-Fisica,-Sezione-C. 16. (3). p. 289-302, 1993.

22. Bertiger-Willy-I, Wu-Jiun-tsong, Wu-Sien-C ,"Formal calculation of resolution and covariance matrices associated with mantle tomography. ",Journal-of-Geophysical-Research,-B,-Solid-Earth-and-Planets. 97. (2). p. 1965-1971.

23. Bisagni JJ, "Ocean surface topography measured by the GEOSAT radar altimeter during the Frontal Air-Sea Interaction Experiment", J. GEOPHYS. RES. (C OCEANS), vol. 96, no. C12, pp. 22,087-99, 1991.

24. Boedecker-G ,"Statistical properties of accumulated values and their application in mining estimations. ",Halmos, F., Somogyi, J. Optimization of design and computation of control networks. p. 635-640.

25. Boulanger-F ,"Statistical inference of trend and covariance of a random field with nonstationary mean and stationary covariance properties. ",Armstrong, M. Geostatistics, Proceedings of the Third international geostatistics congress, Volume 1. Ec. Natl. Super. Mines Paris, Cent. Geostatistique, Fontainebleau, France. p. 259-271.

26. Brenner, S. ,"Producing accurate maps of the Gulf Stream thermal front using Objective Analysis. ",J.-GEOPHYS.-RES.-C-OCEANS. 1989. vol. 94, no. C9, pp. 12593-692

27. Borenstein-Herbert, "Statistical comparisons among subjective analyses and an objective analysis", 1979.

28. Bretherton Francis P., Davis Russ E. and Fandry C.B., "A Technique for Objective Analysis and Design of Oceanographic Experiments Applied to MODE-73", Derp Sea Research, Vol.23, pp. 559-582, 1976.

29. Budillon,-G. ,"Representation of hydrological data: Comparison between results obtained by manual techniques and by objective analysis ",ANN.-FAC.-SCI.-NAUT.-IST.-UNIV.-NAV.-NAPOLI. 1992. vol. 59, pp. 127-132.

30. Burgess-T-M, Webster-R, McBratney-A-B ,"Optimal interpolation and isarithmic mapping of soil properties, IV, Sampling strategy. ",Journal-of-Soil-Science. 32. (4). p. 643-659.

31. Burgess-T-M, Webster-R ,"Optimal interpolation and isarithmic mapping of soil properties, II, Block kriging. ",J.-Soil-Sci. 31. (2). p. 333-341.

32. Burgess-T-M, Webster-R ,"Optimal interpolation and isarithmic mapping of soil properties, I, The semi-variogram and punctual kriging. ",J.-Soil-Sci. 31. (2). p. 315-331.

33. Buxton-Bruce-E ,"Global estimation variance formulas and calculation discussion. ",Mathematical-Geology. 18. (7). p. 693-696.

34. Call-R-D, Savely-J-P, Nicholas-D-E ,"Estimation of joint set characteristics from surface mapping data. ",Symp.-Rock-Mech.,-Proc. (17). p. 2B2.1-2B2.9.

35. Carter E. F. and Robinson A. R., "Analysis Models for Estimation of Oceanic Fields", Journal of Atmospheric and Oceanic Technology, March 1981.

36. Cats, G.J., Haan, B.J. and Hafkenscheid, L.M. ,"Verification of an Objective Analysis Scheme. Meteorological Analyses after the Chernobyl Disaster Verified by Trajectories of Radioactive Air Masses. ",Govt-Reports-Announcements-&-Index-(GRA&I),-Issue-18,-1989.

37. Chayes, Felix ,"Effect of a single nonzero open covariance on the simple closure test. ",Plenum Press. p. 11-22, New York-London, 1970.

38. Chelton DB, Schlax MG, Witter DL and Richman JG, "GEOSAT altimeter observations of the surface circulation of the Southern Ocean", J. GEOPHYS. RES. (C OCEANS), vol. 95, no. C10, pp. 17877-903, 1990.

39. Chemerenko,-E.P. ,"Experiments with statistical objective analysis techniques for representing a coastal surface temperature field. ",METEOROL.-GIDROL. 1985. no. 1, pp. 86-90

40. Chin-M ,"On the positive definiteness of a homogeneous and isotropic covariance function. ",Anonymous. American Geophysical Union, 1981 fall meeting. Eos,-Transactions,-American-Geophysical-Union. 62. (45). p. 842

41. Christakos-G ,"Modern statistical analysis and optimal estimation of geotechnical data. ",Engineering-Geology. 22. (2). p. 175-200, 1985.

42. Christy-Alfred-A, Kvalheim-Olav-M, Oygard-Kjell, Dahl-Birger, Karstang-Terje-V ,"Approach to the volcanic tremor by covariance analysis, application to the 1989 eruption of Mt. Etna (Sicily). ",Organic-Geochemistry. 17. (1). p. 63-74.

43. Clancy R. M., "Effect of Observational Error Correlations on Objective Analysis of Ocean Thermal Structure", Deep Sea Res., Vol. 30, No. 9A, 1983, pp. 985-1002.

44. Cover T. M., "Rates of Convergence of Nearest Neighbor Decision Procedures", in Proc. 1st Annu. Hawaii Conf. Systems Theory, Jan. 1967.

45. Cover T. M. and Hart P. E., "Nearest neighbor Pattern Classification", IEEE Trans. Inform. Theory, Vol. IT-13, Jan. 1967, pp.21-27.

46. Clancy RM, "The effect of observational error correlations on objective analysis of ocean thermal structure", DEEP-SEA RES., vol. 30, no. 9A, pp. 985-1002, 1983.

47. Crise,-A., Manca,-B. ,"Digital thematic maps from CTD measurements. A case study in the Adriatic Sea. ",BOLL.-OCEANOL.-TEOR.-APPL. 1992. vol. 10, no. 1, pp. 15-40.

48. Daley,-R., Hollingsworth,-A., Ploshay,-J., Miyakoda,-K., Baker,-W., Kalnay,-E., Dey,-C., Krishnamurti,-T. and Barker,-E. ,"Objective mapping by least squares fitting. ",BULL.-AM.-METEOROL.-SOC. 1985. vol. 66, no. 6, pp. 532-538.

49. Dandonneau,-Y. and Gohin,-F. ,"Objective analysis of simulated equatorial Atlantic Ocean data on seasonal time scales. ",DEEP-SEA-RES. 1984. vol. 31, no. 12A, pp. 1377-1393.

50. David-M, Crozel-D, Robb-J-M ,"Automated mapping of the ocean floor using the theory of intrinsic random functions of order K. ",Marine-Geophysical-Researches. 8. (1). p. 49-74.

51. Davis-B-M, Hagan-R, Borgman-L-E ,"Development of an observation scheme from the variance-covariance matrix of the unknowns. ",Computers-and-Geosciences. 7. (2). p. 199-206.

52. Davis-Michael-W ,"Generating large stochastic simulations, the matrix polynomial approximation method. ",Mathematical-Geology. 19. (2). p. 99-107.

53. Davis-Michael-W ,"Production of conditional simulations via the LU triangular decomposition of the covariance matrix. ",Mathematical-Geology. 19. (2). p. 91-98.

54. Davis R. E., "Objective mapping by least squares fitting", J. GEOPHYS. RES. (C OCEANS)., vol. 90, no. C3, pp. 4773-4777, 1985

55. De-Cauwer, G. ,"Objective analysis of tidal fields in the Atlantic and Indian Oceans. ",ANN.-SOC.-R.-ZOOL.-BELG.-ANN.-K.-BELG.-VER.-DIERKD. 1985. vol. 115, no. 2, pp. 183-196

56. Denman KL and Freeland HJ, "Correlation scales, objective mapping and a statistical test of geostrophy over the continental shelf", J. MAR. RES., vol. 43, no. 3, pp. 517-539, 1985.

57. Derwent, R.G. ,"A comparison of model photochemical ozone formation potential with observed regional scale ozone formation during a photochemical episode over the United Kingdom in April 1987. ",ATMOS.-ENVIRON. vol. 23, no. 6, pp. 1361-1371, 1989.

58. Dick-K-A, Sivjee-G-G ,"O2 Herzberg I bands in the night airglow, covariation with O I (5577). ",J. Geophys. Res. Vol. 76, No. 28, p. 6987-6989, illus, 1971.

59. Dietrich-C-R, Newsam-G-N ,"An application of the deterministic variogram to design-based variance estimation. ",Mathematical-Geology. 27. (2). p. 207-228.

60. Dijkstra, S. ,"Simple uses of covariograms in geology ",Geol.-Mijnbouw. 55. (1-2). p. 105-109.

61. Dimroth-Erich ,"Meimechites and carbonatites of the Castignon Lake complex, New Quebec. ",Neues Jahrb. Mineral., Abh. Vol. 112, No. 3, p. 239-278 (incl. Ger., Fr. sum.), illus. (incl. map), 1970.

62. Dowd-P ,"Generalized covariances and structural analysis, a comparison. ",Water-Resources-Research. 25. (7). p. 1737-1747.

63. Easley-Dale-H ,"Efficient generation of conditional simulations by Chebyshev matrix polynomial approximations to the symmetric square root of the covariance matrix. ",168 pp.

64. Eddy-A, "Objective analysis of convective scale rainfall using gages and radar", J.-Hydrol. 44. (1-2). p. 125-134, 1979.

65. Ehrenberg-J-E, Hernandez-E-N ,"Covariance and spectral analysis of hydraulic conductivity data. ",Bulletin-of-the-Seismological-Society-of-America. 71. (4). p. 1361-1367.

66. Ehrenberg-J-E, Hernandez-E-N ,"Comments on "Covariance-invariant digital filtering, a better digital processing technique for ground motion studies," by J. E. Ehrenberg and E. N. Hernandez. ",Bulletin-of-the-Seismological-Society-of-America. 72. (4). p. 1447.

67. El-Sabh MI and Murty TS, "Seasonal and long-term sea level variations in the Atlantic coast of Canada", MAR. GEOD., vol. 10, no. 3-4, pp. 295-308, 1986.

68. Estes-R-H, Lancaster-E-R ,"A simulation for gravity fine structure recovery from low-low Gravsat SST data. ",Eos-(Am.-Geophys.-Union,-Trans.). 57. (8). AGU spring annual meeting, 1976, postdeadline abstracts. p. 592.

69. Feng-Z-L, Lewis-Roland-W ,"Optimal estimation of in-situ ground stresses from displacement measurements. ",International-Journal-for-Numerical-and-Analytical-Methods-in-Geomechanics. 11. (4). p. 391-408.

70. Foote-Mike ,"Nearest-neighbor analysis of trilobite morphospace. ",Systematic-Zoology. 39. (4). p. 371-382.

71. Forsberg-R ,"Local covariance functions and density distributions. ",Reports-of-the-Department-of-Geodetic-Science-and-Surveying. 356. 52 p.

72. Forsberg-Rene ,"A new covariance model for inertial gravimetry and gradiometry. ",Journal-of-Geophysical-Research,-B,-Solid-Earth-and-Planets. 92. (2). p. 1305-1310.

73. Fisher E. P. and Patrick E. P., "A Preprocessing Algorithm for Nearest Neighbor Decision Rules", in Proc. Nat. Electronics Conf., Vol. 26, pp.481-485, Dec. 1970.

74. Flower-B-P, Kennett-J-P ,"Accurate calculation of block-sample covariances using Gauss quadrature. ",Paleoceanography. 8. (6). p. 811-843.

75. Friedman J. H., Baskett F. and Shustek L. J., "An Algotirhm for Finding Nearest Neighbors", IEEE Transactions on Computers, Oct. 1975, pp. 1000-1006.

76. Freeland H.J. and Gould W.J., "Oblective Analysis of Meso-Scale Ocean Circulation Features", Deep Sea Research, Vol.23, pp. 915-923, 1976.

77. Fruehauf, G., Halpern, P. and Lester, P. ,"Objective analysis of a two dimensional scalar field by successive corrections using a personal computer. ",ENVIRON.-SOFTWARE. vol. 3, no. 2, pp. 72-80, 1988.

78. Fukunaga K. and Hostetler L. D., "Optimization of k-nearest-neighbor Density Estimates", IEEE Trans. Inform. Theory, Vol. IT-19, May 1973, pp. 320-326.

79. Gelb A., "Applied Optimal Estimation", MIT Press, Cambridge, MA, 374pp.,1986.

80. Gambolati,-G., Galeati,-G. ,"Optimal interpolation of bathymetry in the Tyrrhenian Sea. ",HYDRAULIC ENGINEERING. PROCEEDINGS OF THE 1989 NATIONAL CONFERENCE ON HYDRAULIC ENGINEERING. Ports,M.A.ed. 1989. pp. 813-819

81. Gelhar-L-W, Axness-C-L ,"Reply to J. L. Beck's "Comments on 'Covariance-invariant digital filtering, a better digital processing technique for ground motion studies"'. ",Water-Resources-Research. 19. (1). p. 161-180.

82. Gill, T.A.,"Objective analysis of seafood quality ",Food-Reviews-International, 6 (4) 681-714, 1990.

83. Goff-John-A, Holliger-Klaus, Lavander-Alan ,"Spatial interpolation of soil moisture data with universal cokriging. ",Geophysical-Research-Letters. 21. (6). p. 493-496.

84. Gomez-Hernandez-J-Jaime, Freyberg-David-L ,"Propagation and filtering of concentration estimation error covariance for a one-dimensional mass transport problem, application to network design. ",Anonymous. AGU 1987 fall meeting. Eos,-Transactions,-American-Geophysical-Union. 68. (44). p. 1265

85. Gomez-Hernandez-J-J ,"Estimation of spatial cross-covariances by maximum likelihood cross-covalidation, application to hydraulic heads and transmissivities in a sandy aquifer. ",Anonymous. 4th international geostatistics congress. Terra-Abstracts. 4 (Suppl. 3). p. 4

86. Goyal-S-K, "An objective analysis of pump test data from a non-penetrating well", Proceedings, general reports and lectures late papers/International conference on modern approach to groundwater resources management, International Association for Hydraulic Research, Italy, Consiglio Nazionale delle Ricerche, Italy; International Institute for Applied Systems Analysis, Italy, Istituto di Idraulica e Costruzioni Idrauliche, Milan, Italy, p. 115-122, 1983.

87. Graham-Wendy-D, Neff-Christina-R ,"Optimal estimation of spatially variable recharge and transmissivity fields under steady-state groundwater flow, Part 2, Case study. ",Journal-of-Hydrology. 157. (1-4). p. 267-285.

88. Graham-Wendy-D, Tankersley-Claude-D ,"Optimal estimation of spatially variable recharge and transmissivity fields under steady-state groundwater flow, Part 1, Theory. ",Journal-of-Hydrology. 157. (1-4). p. 247-266.

89. Graham-Wendy-D, Tankersley-Claude-D ,"Petrogenetic significance of the covariance relationship between compatible and incompatible elements. ",Journal-of-Hydrology. 157. (1-4). p. 247-266.

90. Greenkorn-Robert-Albert and Johnson-Carlton-Robert, "Variation of a natural sandstone reservoir element–An objective analysis of core measurements", Jour. Petroleum Technology. v. 12, no. 9, p. 58, Sept, 1960.

91. Groten, Erwin ,"On the covariance function of the terrain correction. ",Ceskoslovensk. akad. ved studia geophys. et geod., v. 14, no. 2, p. 264-269, 1970.

92. Hajela-D-P ,"Optimal estimation of high degree gravity field from a global set of 1 degrees X 1 degrees anomalies to degree and order 250. ",Reports-of-the-Department-of-Geodetic-Science-and-Surveying. 358. 60 p.

93. Hajela-D-P ,"Tests for optimal estimation of high degree and order gravity fields. ",Eden, H. Frank. AGU 1984 spring meeting. Natl. Sci. Found., Washington, DC, United-States. Eos,-Transactions,-American-Geophysical-Union. 65. (16). p. 181

94. Hantush-Mohamed-M, Marino-Miguel-A ,"Two-dimensional stochastic analysis and optimal estimation in aquifers, random recharge. ",Water-Resources-Research. 30. (2). p. 559-569.

95. Harper-Denver, Olyphant-Greg-A ,"Empirical covariance functions between seismic, density and gravity data, an important constraint in 3D gravimetric-seismic stochastic inversion. ",Journal-of-Hydrology. 146. (1-4). p. 49-71.

96. Harrison-M-J, Venn-C, Skoog-S-Y, Gledhill-H-M, Noto-R-C, Simpson-E-L ,"Nearest neighbor analysis of Lower Cambrian Skolithos and Moncraterion ichnofossils, implications for preservation and sequence boundary identification. ",Anonymous. Geological Society of America, Southeastern Section, 43rd annual meeting. Abstracts-with-Programs-Geological-Society-of-America. 26. (4). p. 18

97. Hawkins,-J.D. ,"TOTS: Three dimensional ocean thermal structure analysis. ",SEA-TECHNOL. 1992. vol. 33, no. 1, pp. 62-63

98. Hessler G, "Experiments with statistical objective analysis techniques for representing a coastal surface temperature field", BOUNDARY-LAYER METEOROL., vol. 28, no. 3-4, pp. 375-389, 1984.

99. Hessler, G. ,"Objective analysis of hydrographic data sets from mesoscale surveys. ",BOUNDARY-LAYER-METEOROL. 1984. vol. 28, no. 3-4, pp. 375-389

100. Hessler G, "Investigations on Boundary Temperature and Wind Fields in the Convergence Coast-Sea on the Example of Kiel Bight", INST. MEERESKUNDE, CHRISTIAN-ALBRECHTS UNIV., KIEL (FRG), 1981, 93 pp, BER.

101. Hiller W and Kaese RH, "Objective analysis of hydrographic data sets from mesoscale surveys", 1983; 78 pp, BER. INST. MEERESKD. CHRISTIAN-ALBRECHTS-UNIV. KIEL., no. 116.

102. Hiller,-W., Kaese,-R.H. ,"Objective analysis of hydrographic data sets from mesoscale surveys. ",BER.-INST.-MEERESKD.-CHRISTIAN-ALBRECHTS-UNIV.-KIEL. 1983. no. 116, 78 pp

103. Hitchcock-GL, Mariano-AJ, Rossby-T ,"Mesoscale pigment fields in the Gulf Stream: Observations in a meander crest and trough ",J.-GEOPHYS.-RES.-(C-OCEANS) vol. 98, no. C5, pp. 8425-8445, 1993.

104. Hoeksema-R-J, Kitanidis-P-K ,"Analysis of the spatial structure of properties of selected aquifers. ",Water-Resources-Research. 21. (4). p. 563-572.

105. Hofmeister-Anne-M ,"Calculation of bulk modulus and its pressure derivatives from vibrational frequencies and mode Grueneisen parameters, solids with cubic symmetry or one nearest-neighbor distance. ",Journal-of-Geophysical-Research,-B,-Solid-Earth-and-Planets. 96. (10). p. 16,181-16,203.

106. Hogan PJ, Hurlburt HE, Jacobs G, Wallcraft AJ, Teague WJ and Mitchell JL, "Simulation of GEOSAT, TOPEX/Poseidon, and ERS-1 altimeter data from a 1/8 degree Pacific Ocean model: Effects of space-time resolution on mesoscale sea surface height variability", MAR. TECHNOL. SOC. J., vol. 26, no. 2, pp. 98-107. 1992.

107. Horai-Ki-iti ,"Cross-Covariance Analysis of Heat Flow and Gravitational Field of the Earth. ",Eos (Am. Geophys. Union, Trans.). Vol. 52, No. 11, p. 924, 1971.

108. Horai-Ki-iti ,"Cross-covariance analysis of heat flow and seismic delay times for the Earth. ",Earth Planet. Sci. Lett. Vol. 7, No. 2, p. 213-220, illus. (incl. sketch maps), 1969.

109. Horai-Ki-iti ,"Cross-covariance analysis of heat flow and seismic delay times for the Earth. ",Earth and Planetary Sci. Letters. v. 7, no. 2, p. 213-220, illus., table, 1969.

110. Hua, B.L., McWilliams, J.C. and Owens, W.B. ,"Objective analysis of thermocline depth distributions obtained in the Tropical Atlantic Ocean during FGGE, 1979. ",J.-PHYS.-OCEANOGR. 1986. vol. 16, no. 3, pp. 506-522

111. Hua BL, McWilliams JC and Owens WB, "An objective analysis of the POLYMODE Local Dynamics Experiment. Part 2: Streamfunction and potential vorticity fields during the intensive period", J. PHYS. OCEANOGR, vol. 16, no. 3, pp. 506-522, 1986.

112. Imawaki,-S., Ichikawa,-K., Nishigaki,-H. ,"Mapping the mean sea surface elevation field from satellite altimetry data using optimal interpolation. ",MAR.-GEOD. 1992. vol. 15, no. 1, pp. 31-46

113. Imawaki,-S., Nishigaki,-H., Ichikawa,-K. ,"Satellite altimetry data processing. Correction of radial orbit error by optimal interpolation technique. ",SORA-TO-UMI. 1989. no. 11, pp. 25-38

114. Isemer,-H.-J., Hasse,-J. ,"GEOSAT altimeter observations of the surface circulation of the Southern Ocean. ",BERLIN-FRG -SPRINGER-VERLAG 1987. 252 pp.

115. Jalickee, J.B., and Hamilton, D.R. (Cent.-Exp.-Design-and-Data-Anal.,-Environ.-Data-Serv.,-NOAA,-Washington,-DC-20235,-USA) ,"Objective analysis and classification of oceanographic data. ",Tellus, 1977 29(6), 545-560.

116. Jekeli,-C. ,"Optimal estimation of gravity from gravity gradients at aircraft altitude. ",TECH.-REP.-U.S.-AIR-FORCE-GEOPHYS.-LAB. 1985. 12 pp

117. Jekeli-C ,"On optimal estimation of gravity from gravity gradients at aircraft altitude. ",Reviews-of-Geophysics. 23. (3). p. 301-311.

118. Jervis-M, Stoffa-P-L, Sen-M-K ,"Knowledge of correlation length scales can lead to improved hydrogeologic parameter estimates. ",Anonymous. AGU 1993 spring meeting. Eos,-Transactions,-American-Geophysical-Union. 74. (16 Suppl.). p. 201

119. Ji,Ming, Leetmaa, A. and Derber, J. ,"An ocean analysis system for seasonal to interannual climate studies ",MON.-WEATHER-REV. 1995 vol. 123, no. 2, pp. 440-481

120. Johnson-E-R, Bras-R-L ,"Real-time estimation of velocity and covariance structure of rainfall events using telemetered raingage data, a comparison of methods. ",J.-Hydrol. 44. (1-2). p. 97-123

121. Johnson, J.E., Wittmann, P.A. and Mooers, C.N. ,"The use of multiobjective analysis for comparing and evaluating environmental and economic goals in wetlands managements. ",REP.-U.S.-NAV.-POSTGRAD.-SCH. 1988. 95 pp.

122. Jonah-Maxwell-V ,"Structured covariance matrices. ",unknown p.

123. Jordan-S-K, Heller-W-G ,"Upward continuation of gravity disturbance covariance functions. ",Eos-(Am.-Geophys.-Union,-Trans.). 56. (12). Fall annual meeting. p. 973.

124. Jordan-S-K, Heller-W-G ,"Upward continuation of gravity disturbance covariance functions. ",J.-Geophys.-Res. 83. (B7). p. 3382-3388.

125. Julian PR and Thiebaux HJ, "Regime dependent analysis experiments", MESOSCALE ANALYSIS AND FORECASTING: PROCEEDINGS OF AN INTERNATIONAL SYMPOSIUM, VANCOUVER, CANADA, 17-19 AUGUST, 1987, pp. 445-451, SPEC. PUBL. EUR. SP. AGENCY.

126. Jurdy-Donna-M, Stefanick-Michael ,"Errors in plate rotations as described by covariance matrices and their combination in reconstructions. ",Journal-of-Geophysical-Research,-B,-Solid-Earth-and-Planets. 92. (7). p. 6310-6318.

127. Kalinina-T-B ,"Quantitative estimation of the maximum efficiency of solutions of inverse problems in magnetic prospecting. ",Phys. Solid Earth (Engl. Ed.). No. 8, p. 487-494, illus, 1970.

128. Kaminski, J.W., Mcconnell, J.C. and Sandilands J.W. ,"Calculations of Arctic Ozone Chemistry Using Objectively Analyzed Data in a 3-D CTM. ",Govt-Reports-Announcements-& Index (GRA&I), Issue-02, 1995.

129. Kane-V-E, Begovich-C-L, Butz-T-R, Myers-D-E ,"Interpretation of regional geochemistry using optimal interpolation parameters. ",Computers-and-Geosciences. 8. (2). p. 117-135.

130. Kautzleben-H, Harnisch-M, Schwahn-W ,"The statistical description and interpretation of geophysical potential fields using covariance functions. ",Fuchs, K., Mueller, G. Proceedings of the Eleventh international symposium on mathematical geophysics. J.-Geophys. 43. (1-2). p. 163-177.

131. Kistler, R.E., McPherson, R.D. and Yates, H.W. (ed.) ,"Use of satellite data in operational numerical weather prediction. ",RECENT-ADVANCES-IN-CIVIL-SPACE-REMOTE-SENSING. 1984, vol. 481, pp. 92-99.

132. Kitanidis-P-K ,"Parameter uncertainty in estimation of spatial functions, Bayesian analysis. ",Water-Resources-Research. 22. (4). p. 499-507.

133. Kitanidis-P-K ,"Minimum-variance unbiased quadratic estimation of covariances of regionalized variables. ",Journal-of-the-International-Association-for-Mathematical-Geology. 17. (2). p. 195-208.

134. Kitanidis-Peter-K ,"Parametric estimation of covariances of regionalized variables. ",Water-Resources-Bulletin-(Urbana). 23. (4). p. 557-567.

135. Kitty-Kevin-T, "Objective analysis", Peters, Douglas C. GeoTech '88, geocomputing tools, PCs, workstations, and more. Comput. Orient. Geol. Soc., Denver, CO, United-States. COGS-Computer-Contributions. 4. (3). p. 132-133, 1988.

136. Kolesova-V-I, "Experimental use of objective analysis for constructing magnetic maps, Geomagnetism-and-Aeronomy. 19. (6). p. 764-766, 1979.

137. Kontoyiannis,-H., Watts,-D.R.* ,"Observations on the variability of the Gulf Stream path between 74 degree W and 70 degree W ",J.-PHYS.-OCEANOGR. 1994 vol. 24, no. 9, pp. 1999-2013.

138. Krajewski-Witold-F, Duffy-Christopher-J ,"Estimation of correlation structure for a homogeneous isotropic random field, a simulation study. ",Computers-and-Geosciences. 14. (1). p. 113-122.

139. Krauss,-W., Fahrbach,-E., Aitsam,-A., Elken,-J. and Koske,-P. ,"Derived Lagrangian statistics on the Vancouver Island continental shelf and implications for salmon migration. ",DEEP-SEA-RES. 1987. vol. 34, no. 7A, pp. 1163-1185.

140. Krumbein-W-C, Jones-Thomas-A ,"The influence of areal trends on correlations between sedimentary properties. ",J. Sediment. Petrology. Vol. 40, No. 2, p. 656-665, illus, 1970.

141. Krumbein, William C. ,"Areal variation and statistical correlation. ",in Mathematical models of sedimentary processes, p. 167-173. Plenum Press. New York-London, 1972.

142. Kushnir-A-F, Pinskii-V-I (Pinskiy, V. I.) ,"Optimal estimation of response parameters of a medium from observations of sounding signals at spaced points. ",Computational-Seismology:-Earthquake-Prediction-and-the-Structure-of-the-Earth. 18. p. 188-202.

143. Kushnir-A-F, Pinskiy-V-I ,"Optimal'nyye otsenki parametrov otklika sredy po nablyudeniyam signalov zondirovaniya v raznesennykh tochkakh. ",Keylis-Borok, V. I., Levshin, A. L. Teoriya i analiz seysmologicheskoy informatsii. Theory and analysis of seismological information. Vychislitel'naya-Seysmologiya. 18. p. 201-217.

144. Lachapelle-G, Schwarz-K-P ,"An investigation on the relationship among the models of covariance function used in geology. ",The-Canadian-Surveyor. 34. (3). p. 251-264.

145. Lamden-R-J ,"A Fourier covariance analysis of the long period magnetic field variations in the British Isles during 1958. ",R. Astron. Soc., Geophys. JVol. 20, No. 2, p. 177-189, illus, 1970.

146. Lardner,-R.W., Das,-S.K. ,"Optimal estimation of eddy viscosity for a quasi-three-dimensional numerical tidal and storm surge model ",INT.-J.-NUMER.-METHODS-FLUIDS 1994 vol. 18, no. 3, pp. 295-312

147. Legler-David-M and Navon-I-M, "VARIATM; a Fortran program for objective analysis of pseudostress wind fields using large-scale conjugate-gradient minimization", Computers-and-Geosciences. 17. (1). p. 1-21, 1991.

148. Le Traon P.Y., "A Method for Optimal Analysis of Fields with Spatially Variable Mean", Journal of Geophysical Research, Vol.95, No.C8, pp. 13543-13547, August 1990.

149. Le Traon P.Y., "Time Scales of Mesoscale Variability and Their Relationship with Space Scales in the North Atlantic", Journal of Marine Research, Vol. 49, 1991, pp. 467-492.

150. Le Traon P.Y. and Hernandez F., "Mapping the Oceanic Mesoscale Circulation: Validation of Satellite Altimetry Using Surface Drifters", Journal of Atmospheric and Oceanic Technology, March 1992.

151. Lewis,-M.R. and Smith,-J.C. ,"The effect of observational error correlations on objective analysis of ocean thermal structure. ",MAR.-ECOL.-PROG.-SER.. 1983. vol. 13, no. 1, pp. 99-102

152. Li, K.L. ,"On some aspects of objective analysis of humidity over Indian region by the optimum interpolation method. ",J.-TROP.-METEOROL.-REDAI-QIXIANG. 1987. vol. 3, no. 1, pp. 63-71.

153. Li K-L, "The objective analysis of the surface wind field over the South China Sea", J. TROP. METEOROL./REDAI QIXIANG, vol. 3, no. 1, pp. 63-71, 1987. NOAA TECH. MEMO. ERL-PMEL.

154. Li-Zhe, Jin-Mingzhi, He-Wei, Liu-Milan ,"Next nearest neighbor effect in Mossbauer spectra of chromite. ",Kexue-Tongbao-(Foreign-Language-Edition). 33. (18). p. 1548-1551.

155. Liebelt P. B., "An Introduction to Optimal Estimation", Addison Wesley, Reading, MA, 273pp., 1967.

156. Lim-Suan-Medel-P ,"Covariance and harmonic analysis of the interaction of lunar gravity and its surface characteristics. ",160 p.

157. Liu-Xinlan, Sun-Hongbing ,"Estimation of resolution and covariance for large matrix inversions. ",Anonymous. Geological Society of America, Southeastern Section, 44th annual meeting. Abstracts-with-Programs-Geological-Society-of-America. 27. (2). p. 70

158. Livieratos-E, Vlachos-D ,"Best invariant covariance component estimators and their application to the generalized multivariate adjustment of heterogeneous deformation observations. ",Tectonophysics. 77. (3-4). p. 323-332

159. Lo-Kwok-wai-Kenneth, "Comparisons between subjective analysis and the NCAR multivariate statistical objective analysis scheme", 1978.

160. Lopez-Hernandez-I-Danilo, Burnham-C-P ,"The covariance of phosphate sorption with other soil properties in some British and tropical soils. ",J. Soil Sci. Vol. 25, No. 2, p. 196-206, 1974.

161. Lorenc A.C., "A Global Tree-Dimensional Multivariate Statistical Interpolation Scheme", Monthly Weather Review, Vol.109, pp. 701-721, April 1981.

162. MacBeth-Colin, Yardley-Gareth-S ,"Optimal estimation of crack-strike. ",Geophysical-Prospecting. 40. (8). p. 849-872.

163. Macbeth-Colin, Yardley-Gareth-S ,"Optimal estimation of crack strike. ",Anonymous. European Association of Exploration Geophysicists, 53rd meeting and technical exhibition, technical programme and abstracts of papers. Technical-Programme-and-Abstracts-of-Papers-Europ

164. Lee-Sang-Il, Kitanidis-Peter-K ,"Optimal estimation and scheduling in aquifer remediation with incomplete information. ",Water-Resources-Research. 27. (9). p. 2203-2217.

165. Mac-Caskie-D-R ,"Identification of petrogenetic processes using covariance plots of trace-element data. ",Chemical-Geology. 42. (1-4). p. 325-341.

166. Mantoglou-A, Wilson-J-L ,"Stochastic analysis of spatial variability in two-dimensional steady groundwater flow assuming stationary and nonstationary heads. ",Water-Resources-Research. 18. (5). p. 1379-1394.

167. Mao,-J.P. and Si,-G.-W. ,"Evaluation of surface wind and flux analysis techniques using conventional data in marine cyclones. ",DONGHAI-MAR.-SCI.-DONGHAI-HAIYANG. 1986. vol. 4, no. 1, pp. 1-11.

168. Marcus-Allan-H ,"Distribution and covariance function of elevations on a cratered planetary surface, part 1, Theory. ",Moon. Vol. 1, No. 3, p. 297-337, illus, 1970.

169. Mariano A. J., "Contour Analysis: A New Approach for Melding Geophysical Fields", Journal of Atmospheric and Oceanic Technology, April 1990.

170. Mariano,-A.J., Brown,-O.B. ,"Efficient objective analysis of dynamically heterogeneous and nonstationary fields via the parameter matrix. ",DEEP-SEA-RES. 1992. vol. 39, no. 7-8A, pp. 1255-1271.

171. Marshall-R-J, Mardia-K-V ,"Minimum norm quadratic estimation of components of spatial covariance. ",Journal-of-the-International-Association-for-Mathematical-Geology. 17. (5). p. 517-525.

172. McBratney-A-B, Webster-R ,"Optimal interpolation and isarithmic mapping of soil properties, V, Co-regionalization and multiple sampling strategy. ",Journal-of-Soil-Science. 34. (1). p. 137-162.

173. McIntosh,-P.C. ,"Oceanographic data interpolation: Objective analysis and splines ",J.-GEOPHYS.-RES.-C-OCEANS 1990 vol. 95, no. 8, pp. 13529-13541.

174. McPhaden,-M.J., Reverdin,-G., Merle,-J., Du-Penhoat,-Y., Kartavtseff,-A. ,"Objective analysis of simulated equatorial Atlantic Ocean data on seasonal time scales. ",DEEP-SEA-RES. 1984. vol. 31, no. 5A, pp. 551-569

175. McQuivey-Raul-S, Keefer-Thomas-N ,"Measurement of velocity-concentration covariance. ",Am. Soc. Civ. Eng., Proc., J. Hydraul. Div. Vol. 98, No. HY9, p. 1625-1646, illus, 1972.

176. McWilliams,-J.C., Owens,-W.B., Hua,-B.L. ,"An objective analysis of the POLYMODE Local Dynamics Experiment. Part 1: General formalism and statistical model selection. ",J.-PHYS.-OCEANOGR. 1986. vol. 16, no. 3, pp. 483-504

177. McWilliams,J.C., Shen,C.Y.(Natl.-Cent.-Atmos.-Res.,-Boulder,-CO-80307,-USA) ,"An objective analysis of the boundary-layer thermodynamic structure during GATE. Part II: analysis. ",J.-Phys.-Oceanogr., 1980 10(5), 741-752.

178. McWilliams J. C. and Owens W. B., "Estimation of Spatial Covariances from the Mid-Ocean Dynamics Experiment", NCAR Technical Note NCAR/TN-115+STR, Nat'l Center for Atmospheric Research, Boulder, Colo, 25pp., 1976.

179. McWilliams J. C., Owens W. B. and Hua L. B., "An Objective Analysis of the Polymode Local Dynamics Experiment. Part I: General Formalism and Statistical Model Selection", Journal of Phys. Ocean., Vol. 16, 1986, pp. 483-504.

180. McWilliams,-J.C., Owens,-W.B., Hua,-B.L. ,"An objective analysis of the POLYMODE Local Dynamics Experiment. Part 2: Streamfunction and potential vorticity fields during the intensive period.

181. Means-Joseph-D ,"Use of the Three-Dimensional Covariance Matrix in Analyzing the Polarization Properties of Plane Waves. ",J. Geophys. Res. Vol. 77, No. 28, p. 5551-5559, illus, 1972.

182. Meissl-Peter ,"A study of covariance functions related to the Earth's potential. ",Eos (Am. Geophys. Union, Trans.). Vol. 52, No. 4, p. 184, 1971.

183. Miesch-A-T, Chao-E-C-T, Cuttitta-Frank ,"Multivariate analysis of geochemical data on tektites. ",Jour. Geology. v. 74, no. 5, pt. 2, p. 673-691, illus., tables, 1966.

184. Meyers,-G., Phillips,-H., Smith,-N., Sprintall,-J. ,"Space and time scales for optimal interpolation of temperature – Tropical Pacific Ocean. ",PROG.-OCEANOGR. 1991. vol. 28, no. 3, pp. 189-218

185. Mikhaylova-N-P, Glevasskaya-A-M, Tsykora-V-N ,"Nekotoryye cherty orogennogo vulkanizma Karpat po dannym paleomagnitnogo kartirovaniya. ",Kulikov, V. S., Svetov, A. P. Paleovulkanizm i yego produkty, sistematika, geologiya, petrologiya, metallogeniya. Paleovolcanism and its products, systematics, geology, petrology, metallogeny. p

186. Maurin-A-F, Riguidel-M-J ,"Optimal mixing and representation of multidimensional data mapping. ",Int.-Geol.-Congr.-Abstr.-Congr.-Geol.-Int.,-Resumes. (26 Vol. 2). p. 826

187. Milliff RF and Robinson AR, "Structure and dynamics of the Rhodes Gyre System and dynamical interpolation for estimates of the mesoscale variability", J. PHYS. OCEANOGR., vol. 22, no. 4, pp. 317-337, 1992.

188. Mooers,-C.N.K. ,"(Objective analysis of actual wind and pressure fields at the sea surface, North Atlantic.). ",APPLICATIONS-OF-REAL-TIME-OCEANOGRAPHIC-CIRCULATION-MODELING: SYMPOSIUM PROCEEDINGS. Parker, B.B. ed. 1986. pp. 189-210.

189. Naumov,-A.D. ,"Distribution, abundance and diversity of benthic macroinvertebrates on the Canadian continental shelf and slope of southern Davis Strait and Ungava Bay. ",SOV.-METEOROL.-HYDROL. 1986. no. 12, pp. 78-85.

190. Nelson-C-A-investigator, Etheridge-J-investigator ,"Some effects of nearest neighbor, bilinear interpolation, and cubic convolution resampling on Landsat data. ",Geological-Survey-Professional-Paper-(Washington,-D.C.). (1175). p. 306

191. Nikkarinen-M-E, Makinen-J-E, Salminen-R-K ,"Statistical interpretation of the regional geochemical mapping data based on the heavy fraction of till in southern Finland. ",Autio, S. Geological Survey of Finland, current research 1989-1990. Special-Paper-Geological-Survey-of-Finland. 12. p. 181-186.

192. Obenson, Gabriel F.T. ,"The covariance matrix for deflections of the vertical and undulations based on actual gravity data - u.s. air force contract f19628-69-c-0127, sci. rept. 9. ",Ohio state univ. dept. geod. sci. rept. 136 (afcrl-70-0408), 19 p, 1970.

193. Ogura,Y., Chen,Y.-L., Russell,J. and Soong,S.-T. (Lab.-Atmos.-Res.,-Univ.-Illinois,-Urbana,-IL-61801,-USA) ,"An attempt at objective analysis of meteo-oceanic parameters. ",Monthly-Weather-Rev., 1979 107(7), 426-441.

194. Osborne-M-D, Fleet-M-E, Bancroft-G-M ,"A Mossbauer study of next-nearest neighbor effects in the solid solution series, chromite-hercynite. ",The Geological Society of America,

95th annual meeting. Abstracts-with-Programs-Geological-Society-of-America. 14. (7). p. 581

195. Pachut-Joseph-F ,"Maximum likelihood estimation of biases, rates, scaling factors and additive covariance matrix for Earth orientation data series. ",Journal-of-Paleontology. 66. (5). p. 750-757.

196. Patrick E. A. and Fisher F. P., "Generalization k-nearest Neighbor Decision Rule", J. Inform. Contr., Vol. 16, Apr. 1970, pp. 128-152.

197. Paul,U. ,"Objective analysis of oceanographic data with orthogonal polynomial surfaces. ",DTSCH. HYDROGR. Z. 1994 vol. 46, no. 1, pp. 29-60.

198. Perrie Will and Toulany B., "Correlations of Sea Level Pressure Fields for Objective Analysis", Mothly Weather Review, Vol. 117, No. 9, pp. 1965-1975, Sept. 1989.

199. Pokrovskii OM and Denisov SG, "Informativeness of the oceanic station network for objective analysis of field of geopotential in the Northern Hemisphere", SOV. METEOROL. HYDROL., no. 10, pp. 29-35, 1985.

200. Papo-H-B, Perelmuter-A ,"Pre-zero-epoch covariance matrix in sequential analysis of deformations. ",Bulletin-Geodesique. 58. (1). p. 75-83.

201. Paul,-U. ,"Objective analysis of oceanographic data with orthogonal polynomial surfaces. ",DTSCH.-HYDROGR.-Z. 1994 vol. 46, no. 1, pp. 29-60

202. Prado,G.-(The-Charles-Stark-Draper-Lab.,-Inc.,-Cambridge,-MA-02139,-USA) ,"Optimal estimation of ship's attitudes and attitude rates. ",IEEE-J.-Oceanic-Eng., 1979 4(2), 52-59

203. Preisendorfer RW and Mobley CD, "Data intercomparison theory. 5. Case study: Effects of objective analysis on a Tropical Pacific sea surface temperature set", NOAA, SEATTLE, WA (USA), 1982; 89 pp, NOAA TECH. MEMO. ERL-PMEL.

204. Prell-W-L ,"Covariance patterns of foraminiferal delta (18)O, an evaluation of Pliocene ice volume changes near 3.2 million years ago. ",Science. 226. (4675). p. 692-694.

205. Rao DB and Schwab DJ, "A Method of Objective Analysis for Currents in a Lake With Application to Lake Ontario", J. PHYS. OCEANOGR., vol. 11, no. 5, pp. 739-750, 1981.

206. Rao, D.B., Steenrod, S.D. and Sanchez, B.V. ,"Comparison of results of objective analysis according to FGGE data. ",NASA-TECH.-MEMO. 1987. 24 pp.

207. Reyment-Richard-A ,"Observations on homogeneity of covariance matrices in paleontologic biometry. ",Biometrics. v. 18, no. 1, p. 1-11, 1962.

208. Reyment-R-A, Ramden-Hans-Ake, Wahlstedt-W-J ,"Fortran IV program for the generalized statistical distance and analysis of covariance matrices for the CDC 3600 computer. ",Kans. Geol. Surv., Comput. Contrib. No. 39, p. 1-41, illus, 1969.

209. Reyment-Richard-Arthur ,"Covariance structure and morphometric analysis, a contribution to paleogenetics. ",Int. Ass. Math. Geol., J. Vol. 1, No. 2, p. 185-197, illus, 1969.

210. Reverdin,-G., Molinari,-R. and Du-Penhoat,-Y. ,"Experience with real-time mesoscale ocean prediction (analysis and forecast) in the EEZ off California. ",DEEP-SEA-RES. 1986. vol. 33, no. 1A, pp. 43-53.

211. Reverdin G, Molinari R and Du Penhoat Y, "Objective analysis of thermocline depth distributions obtained in the Tropical Atlantic Ocean during FGGE, 1979", DEEP-SEA RES., vol. 33, no. 1A, pp. 43-53, 1986.

212. Reyment-R-A ,"Homogeneity of covariance matrices in relation to generalized distances and discriminant functions. ",in Computer applications in the earth sciences–Colloquium on classification procedures. Kansas Geol. Survey Computer Contr. 7. p. 5-9, illus, 1966.

213. Reznik GM and Vinogradova KG, "Calculation of a quasigeostrophic stream function from measurements at self-contained buoy stations", OCEANOL. ACAD. SCI. USSR., vol. 23, no. 4, pp. 502-506, 1983.

214. Rienecker,-M.M., Miller,-R.N. ,"Ocean data assimilation using optimal interpolation with a quasi-geostrophic model. ",J.-GEOPHYS.-RES.-C-OCEANS. 1991. vol. 96, no. C8, pp. 15,093-103

215. Robinson, A.R. and Haidvogel, D.B. ,"A Method of Objective Analysis for Currents in a Lake With Application to Lake Ontario. ",J.-PHYS.-OCEANOGR. 1980. vol. 10, no. 12, pp. 1909-1928

216. Robinson, A.R., Leslie, W.G. ,"Estimation and prediction of oceanic eddy fields. ",ESSAYS-ON-OCEANOGRAPHY:-A-TRIBUTE-TO-JOHN-SWALLOW. Crease,-J.,Gould,-W.J., Saunders,-P.M.-eds. 1985. vol. 14, no. 1-4 pp. 485-510

217. Robinson-J-E ,"Spatial filters for geological data. ",Oil and gas jour., v. 67, no. 37, p. 132-134, 138, 140, 1969.

218. Roemmich,-D. ,"Optimal estimation of hydrographic station data and derived fields. ",J.-PHYS.-OCEANOGR. 1983. vol. 13, no. 8, pp. 1544-1549

219. Ryabinin, A.I., Kravets,V.N. (Sevast.-Otd.-Gos.-Okeanogr.-Int.-Sevastopol'-,-USSR) ,"[Evolution study by objective analysis of an eddy]. ",Okeanologiya-Oceanology-Mosc., ISSN:-0030-1574. 1980 (no. 3), 468-476

220. Sambridge-Malcolm, Braun-Jean, McQueen-Herbert ,"Parameterization and interpolation of irregular data using natural neighbours. ",Anonymous. AGU 1994 fall meeting. Eos,-Transactions,-American-Geophysical-Union. 75. (44 Suppl.). p. 688-689

221. Samper-F-Javier, Neuman-Shlomo-P ,"Estimation of spatial covariance structures by adjoint state maximum likelihood cross validation, 3, Application to hydrochemical and isotopic data. ",Water-Resources-Research. 25. (4). p. 707-712.

222. Samper-F-Javier, Neuman-Shlomo-P ,"Estimation of spatial covariance structures by adjoint state maximum likelihood cross validation, 2, Synthetic experiments. ",Water-Resources-Research. 25. (3). p. 373-384.

223. Samper-F-Javier, Neuman-Shlomo-P ,"Estimation of spatial covariance structures by adjoint state maximum likelihood cross validation, 1, Theory. ",Water-Resources-Research. 25. (3). p. 363-371.

224. Samper-F-Javier, Neuman-Shlomo-P ,"Estimation of spatial covariance structures with application to hydrological, hydrochemical and isotopic data from aquifers, state-of-the-art and adjoint state maximum likelihood cross-validation ", NATO advanced workshop on Advances in analytical and numerical groundwater flow and quality modelling, Polytech. Univ., Catalonia, Sp.

225. Sanchez BV, Rao DB and Steenrod SD, "Objective analysis of tidal fields in the Atlantic and Indian Oceans", National Aeronautics and Space Admin., Greenbelt, MD (USA) OURCE, 1986, 25 pp.

226. Sanchez, B.V., Rao, D.B. and Steenrod, S.D. ,"Informativeness of the oceanic station network for objective analysis of field of geopotential in the Northern Hemisphere. ",1986. 25 pp.

227. Salvadori-G, Ratti-SP, Belli-G, Lovejoy-S and Schertzer-D ,"Multifractal objective analysis of Seveso ground pollution ",TOXICOL.-ENVIRON.-CHEM. vol. 43, no. 1-2, pp. 63-76, 1994.

228. Sandilands, J.W., Kaminski J.W., Mcconnell J.C., Beagley S.R., and Mcfarlane, N. ,"Modelling Stratospheric Polar Ozone Using Objective Analysis. ",Govt-Reports-Announcements-&-Index-(GRA&I),-Issue-02,-1995

229. Santoleri R; Marullo S; Boehm E, "An objective analysis scheme for AVHRR imagery", INT. J. REMOTE SENS, vol. 12, no. 4, pp. 681-693, 1991.

230. Savrov-L-A, Kuchik-E-K ,"Anisotropic covariance functions for local gravity prediction. ",Anonymous. Twentieth lunar and planetary science conference, abstracts. Abstracts-of-Papers-Submitted-to-the-Lunar-and-Planetary-Science-Conference. 20 (Part 3). p. 948-949.

231. Schaffrin-B ,"Mineralogical applications of the lattice constant variance-covariance matrices. ",Int.-Union-Geod.-Geophys.,-Gen.-Assem.,-Abstr. (17). p. 308.

232. Schaffrin-B ,"On the covariance estimation between repeated levellings as applied to the "Testnet Pfungstadt". ",Anonymous. American Geophysical Union, 1983 spring meeting. Natl. und., United-States. Eos,-Transactions, -American-Geophysical-Union. 64. (18). p. 209

233. Schaffrin-B ,"Covariance-invariant digital filtering, a better digital processing technique for ground motion studies. ",Vyskocil, P., Green, R., Maelzer, H. Recent crustal movements, 1979. Tectonophysics. 71. (1-4). p. 354-355

234. Schwahn-W ,"Estimation of local gravity anomaly covariance functions. ",Gerlands-Beitraege-zur-Geophysik. 84. (2). p. 143-154.

235. Shaw-D-M ,"A review of k-rb fractionation trends by covariance analysis. ",Geochim. et cosmochim. acta, v. 32, no. 6, p. 573-601, 1968.

236. Shaw-D-M ,"Covariance analysis of K/Rb fractionation trends. ",Geol. Soc. America Spec. Paper 115. p. 201, 1968.

237. Shaw-D-M ,"A review of K-Rb fractionation trends by covariance analysis. ",Geochim. et Cosmochim. Acta. v. 32, no. 6, p. 573-601, illus., tables, 1968.

238. Shaw-Denis-M ,"A review of K-Rb fractionation trends by covariance analysis. ",Geochim. Cosmochim. Acta. Vol. 32, No. 6, p. 573-601, illus, 1968.

239. Shaw-D-M ,"Covariance analysis of K-Rb fractionation trends. ",Canadian Mineralogist. v. 9, pt. 2, p. 306-307, 1967.

240. Shtemenko-Yu-N ,"Optimal estimation of the magnitude of a source from the data of a combined system of stations. ",Physics-of-the-Solid-Earth. 16. (3). p. 214-218.

241. Silver-P-G, Jordan-T-H ,"Optimal estimation of scalar seismic moment. ",Anonymous. American Geophysical Union, 1981 fall meeting. Eos,-Transactions,-American-Geophysical-Union. 61. (46). p. 1029

242. Silver-P-G, Jordan-T-H ,"Optimal estimation of scalar seismic moment. ",The-Geophysical-Journal-of-the-Royal-Astronomical-Society. 70. (3). p. 755-787.

243. Silver-Paul-Gordon ,"Optimal estimation of scalar seismic moment. ",251 p.

244. Singh-T-R-P ,"On the estimation of the generalized covariance function. ",Journal-of-the-International-Association-for-Mathematical-Geology. 14. (2). p. 107-123.

245. Singh, T.R.P. ,"Point-block covariance, a general solution for the two-dimensional stationary random functions. ",Int.-Assoc.-Math.-Geol.,-J. 8. (6). p. 627-634.

246. Singh-T-R-P ,"Best invariant covariance component estimation and its application to the generalized multivariate adjustment of heterogeneous deformation observations. ",Indian-J.-Earth-Sci. 7. (2). p. 110-118.

247. Sinha SK, Talwalkar DR and Rajamani S, "On some aspects of objective analysis of humidity over Indian region by the optimum interpolation method", ADV. ATMOS. SCI, vol. 4, no. 3, pp. 332-342, 1987.

248. Smedstad O. M., "Data Assimilation and Parameter Estimation in Oceanographic Models", Mesoscale Air-Sea Interaction Group Technical Report, Geophysical Fluid Dynamics Institute, Florida State University, July 1989, NASA Contract NAGW-985.

249. Smith N, Blomley JE, Meyers G. "A univariate statistical interpolation scheme for subsurface thermal analyses in the tropical oceans", PROG. OCEANOGR, vol. 28, no. 3, pp. 219-256, 1991.

250. Smith,-W.O., Heburn,-G.W., Barber,-R.T., O'-Brien,-J.J. ,"Data intercomparison theory. 5. Case study: Effects of objective analysis on a Tropical Pacific sea surface temperature set. ",J.-MAR.-RES. 1983. vol. 41, no. 3, pp. 539-556.

251. Sprintall,-J. and Meyers,-G. ,"An optimal XBT sampling network for the eastern Pacific Ocean. ",J.-GEOPHYS.-RES.-C-OCEANS. 1991. vol. 96, no. C6, pp. 10539-552

252. Stacey MW, Pond S and Le Blond PH, "An objective analysis of the low-frequency currents in the Strait of Georgia", ATMOSPHERE-OCEAN, vol. 26, no. 1, pp. 1-15, 1988.

253. Stammer,-D., Hinrichsen,-H.-H. and Kaese,-R.H. ,"An objective analysis scheme for AVHRR imagery. ",J.-GEOPHYS.-RES.-C-OCEANS. 1991. vol. 96, no. C4, pp. 7005-7014.

254. Stanfield, C.B. ,"Objective analysis of tropical cyclone intensity, strength, and size using routine aircraft reconnaissance data. ",NTIS, SPRINGFIELD, VA (USA) 1986, 126 pp. NTIS SPRINGFIELD, VA (USA)

255. Starks-Thomas-H, Sparks-Allen-R ,"Estimation of the generalized covariance function, II, A response surface approach. ",Mathematical-Geology. 19. (8). p. 769-783.

256. Stein-Michael-L ,"A modification of minimum norm quadratic estimation of a generalized covariance function for use with large data sets. ",Mathematical-Geology. 18. (7). p. 625-633.

257. Stein-Michael-L, Handcock-Mark-S ,"Some asymptotic properties of kriging when the covariance function is misspecified. ",Water-Resources-Research. 25. (3). p. 351-362.

258. Stevens-J-B, "Objective analysis of the embedded Markov matrix representation of a stratigraphic section", The Geological Society of America, 96th annual meeting, Abstracts-with-Programs-Geological-Society-of-America. 15. (6). p. 697, 1983.

259. Stramma,-L. and Siedler,-G. ,"Hydrographic data from the OPTOMA (Ocean Prediction Through Observation, Modeling and Analysis) Program: OPTOMA23, 9-19 November 1986. ",J.-GEOPHYS.-RES.-C-OCEANS. 1988. vol. 93, no. C7, pp. 8111-8118

260. Sun LC, Allen AA and Billing CB, "Statistical models for the optimal estimation of oceanic fields", 1989; 127 pp; REP. U.S. COAST GUARD RES. DEV.

261. Tam-SM ,"Optimal estimation in survey sampling under a regression superpopulation model. ",BIOMETRIKA. vol. 71, no. 3, pp. 645-647, 1984.

262. Teague W. J., Carron M. J. and Hogan P. J., "A Comparison Between the Generalized Digital Environmental Model and Levitus Climatologies", Journal of Geophysical Research, Vol. 995, No. C5, May 15, 1990, pp. 7167-7183.

263. Temple-J-T ,"An empirical study of robustness of nearest-neighbor relations in numerical taxonomy. ",Journal-of-the-International-Association-for-Mathematical-Geology. 14. (6). p. 675-678.

264. Thiebaux H.J., Spatial Objective Analysis with Applications in Atmospheric Science, Academic Press, 299pp., Toronto, 1987.

265. Thiebaux J.J., "Experiments with Correlation Representations for Objective Analysis", Mon. Wea. Rev., Vol. 103, 1975, pp. 617-627.

266. Thiebaux J.J., "Anisotropic Correlation Functions for Objective Analysis", Mon. Wea. Rev., Vol. 104, 1976, pp. 994-1002.

267. Thomas-P-L, Dodson-A-H, Ashkenazi-V ,"A gravity anomaly covariance function for the British Isles. ",Anonymous. Abstracts, The Tenth UK geophysical assembly. The-Geophysical-Journal-of-the-Royal-Astronomical-Society. 85. (1). p. 252

268. Toepfer-K-D ,"Representation and interpretation of resistivity mapping data in groundwater prospecting in Zambia. ",J.-Geophys. 42. (2). p. 147-158.

269. Toompuu,-A., Wulff,-F. ,"Spatial large-scale correlations for optimal interpolation of temperature, salinity and nutrient concentrations in the Gulf of Finland ",ENVIRONMETRICS 1995 vol. 6, no. 1, pp. 55-72

270. Tscherning-C-C ,"Investigations into the uses of variance-covariance analysis of error equation coefficients in aerial triangulation. ",Geophysical-Journal-International. 105. (3). p. 771-776.

271. Van-Egmond, N.D. and Onderdelinden, D. ,"Objective Analysis of Air Pollution Monitoring Network Data, Spatial Interpolation and Network Density ",ATMOS.-ENVIRON. vol. 15, no. 6, pp. 1035-1046, 1981.

272. Van-Lent-Thomas, Kitanidis-Peter-K ,"A numerical spectral approach for the derivation of covariance functions of piezometric head. ",315 p.

273. Vere, Jones D. ,"Space time correlations for microearthquakes, a pilot study. ",Tweedie, R. L. Spatial patterns and processes. Adv.-Appl.-Probab. 10. (1 supplement). p. 73-87

274. Uhrhammer-R-A ,"The optimal estimation of earthquake parameters. ",Engdahl, E. R. Earthquake algorithms. U. S. Geol. Surv., Branch Global Seismol., Denver, CO, United-States. Physics-of-the-Earth-and-Planetary-Interiors. 30. (2-3). p. 105-118.

275. Vyskocil, Vincenc ,"On the covariance and structure functions of the anomalous gravity field. ",Ceskoslovensk. akad. ved studia geophys. et geod., v. 14, no. 2, p. 174-177, 1970.

276. Wagner T. J., "Convergence of the Nearest Neighbor Rule", IEEE Trans. Inform. Theory, Vol.IT-17, Sept. 1971, pp.566-571.

277. Wahlstedt-Warren-C, Davis-John-C ,"FORTRAN IV program for computation and display of principal components. ",Kansas Geol. Survey Computer Contr. 2127 p., illus., tables, 1968.

278. Walstad, L.J., Allen, J.S., Kosro, P.M. and Huyer,-A. ,"Statistical models for the optimal estimation of oceanic fields. ",1991. 67 pp

279. Walstad LJ, Allen JS, Kosro PM and Huyer A, "Dynamics of the coastal transition zone through data assimilation studies", GEOPHYS. RES. (C OCEANS), vol. 96, no. C8, pp. 14,959-977, 1991.

280. Wang,-Jizhi ,"An objective analysis of the low-frequency currents in the Strait of Georgia. ",PROCEEDINGS OF THE NATIONAL CONFERENCE ON TYPHOONS, 1985. TAIFENG HUIYI WENJI, 1985. National Coop. Group of Typhoon Res.,Beijing China, 1987. pp. 334-342.

281. Watts DR, Tracey KL and Friedlander AI, "Producing accurate maps of the Gulf Stream thermal front using Objective Analysis", J. GEOPHYS. RES. (C OCEANS), vol. 94, no. C6, pp. 8040-8052, 1989.

282. Webster-R, Oliver-M-A ,"Optimal interpolation and isarithmic mapping of soil properties, VI, Disjunctive kriging and mapping the conditional probability. ",Journal-of-Soil-Science. 40. (3). p. 497-512.

283. Webster-R, Burgess-T-M ,"Optimal interpolation and isarithmic mapping of soil properties, III, Changing drift and universal kriging. ",J.-Soil-Sci. 31. (3). p. 505-524.

284. Wilcox, Luman E. ,"An investigation of areal gravity-elevation relations using covariance and empirical techniques. ",Hawaii, Univ., Inst. Geophys., [Rep.]. No. HIG-71-10, 118 p., illus. (incl. sketch maps), 1971.

285. Williams, R.H., Horton, C.W. ,"Covariance analysis of aeromagnetic data. ",Eos-(Am.-Geophys.-Union,-Trans.). 56. (12). p. 1112.

286. Xu-Huiping, Sun-Yunsheng, Pu-Chunhua ,"Translated title: The technology of geophysical data mapping, and geophysical image storing and remapping imagery. ",Journal of Changchun College-of-Geology, Changchun Dizhi Xueyuan Xuebao. 23. (3). p. 334-339

287. Yang-Hao, Gu-Lianxing ,"Estimation of covariance parameters in kriging via restricted maximum likelihood. ",Guilin Yejin Dizhi Xueyuan Xuebao, Journal of Guilin College of Geology. 10. (2). p. 201-208.

288. Yu-Yun-Sheng, Heidari-Manoutchehr, Wang-Guang-Te ,"Optimal estimation of contaminant transport in ground water. ",Water-Resources-Bulletin-(Urbana). 25. (2). p. 295-300. 1995

289. Zafar, M. ,"Covariance of trace elements in certain geologic units, Khetri copper belt, Rajasthan, India. ",Indian-Sci.-Congr.-Assoc.,-Proc. (61 Part 3). Abstracts. p. 161.

290. Zolotov-I-G, Roze-Y-N ,"Optimal'naya interpolyatsiya dannykh morskikh gradiyentometricheskikh s'yemok. ",Geomagnetizm-i-Aeronomiya. 21. (6). p. 1081-1086.

8 History of OA Development

8.1 oa1.0

This is the first version of oa program released on Dec., 1993.

- **Program name:** oa1.c
- **Release Date:** Dec. 18, 1993
- **Location:** emeroald:/usr2/objanal/OA/OA1.0
- **Fetures:**
  1. It is an independent oa program without kdtree and search functions.
  2. It takes all the data as the nearest neighbours and perform the optimal estimation.

8.2 kd tree

There are three tree programs tdtree.c, tdstore.c, tdload.c. tdtree.c is an independent programs for building kd-tree from given data points and finding out the nearest neighbours of given grid points. tdstore.c has the same functions as tdtree.c but it can also store the tree into a file (binary or ascii). tdload.c is a program to load the tree stored by tdstore.c in a corespondind format (binary or ascii).

- **Program name:** tdtree.c, tdload.c, tdstore.c
- **Release Date:** March 9, 1994
- **Location:** emeroald:/usr2/objanal/OA/KDTREE
- **Fetures:**
  1. avoid repeated work for unnecessary tree building
  2. It has the options for store and load the kd tree in Binary and ASCII formats.

8.3 oa2.0

In this version 2.0, the kdtree and neighbour search functions are implemented. It is an independent self-contained program which loads the data points and observations of NDV dependent variables, builds a kd tree, finds NUM_CLOSET nearest neighbours and perform the optimal estimation from the observations at the nearest neighbour points for each grid point.

- **Program name:** oad.c
- **Release Date:** April 7, 1994
- **Location:** emeroald:/usr2/objanal/OA/OA2.0

- **Fetures:**

    1. symetrical property of the covariance matrix is considered in the numerical implementation(for example: symmetric matrix inversion, only upper triangle of the covariance matrix is generated)
    2. Dynamically allocate the memory for data_array and index_array so that there is no limit to the number of data points (if the computer resource is available)
    3. float data type is used in numerical computation.
    4. fixed a bug in nodesearch() function(add rescale[]) and a bug in build_kdtree(remove the balance part).
    5. Timing function added on
    6. Error message for prompting user
    7. Error_estimate function is made independent

8.4 oa3.0

The oa programs version 3.0 is an upgrade of OA2.0 with two bugs fixed by Ross Hendry. In addition to all the features of OA2.0, OA3.0 allows the user to specify the local scales and noise levels in grid file and data file.

- **Program name:** oax3.c
- **Release Date:** May 9, 1994
- **Location:** emeroald:/usr2/objanal/OA/OA3.0
- **Fetures:**

    1. Global and Local scale are both considered;
    2. User can specify the noise factor for each data point

8.5 oa4.0

In this oa program version 4.0, the program is restructured based on previous version 3.0. The Global variables are eliminated and changed into local variables so that each function is more independent. This version is the prototype of the PIPE oax implementation. The nearest neighbour search scheme is also improved.

- **Program name:** oax4.c
- **Release Date:** June 1, 1994
- **Location:** emeroald:/usr2/objanal/OA/OA4.0
- **Fetures:**

    1. An extra function data_local is added in for transforming the coordinate of the data point into local grid point coordinate system.

2. All global variables changed to local variables and passed into the functions as parameters.
3. The nearest neighbour search scheme is improved to be able to consider the unisotropic data by the implementation of local scale search in the tree.
4. Graphic display of the nearest neighbour distribution for a given grid point is implemented (using gnuplot).

8.6 oa5.0

oa5.0 is a distributed version of the oa programs. It is an upgrade of oa4.0.

- **Program name:** oax5.c
- **Release Date:** Sept. 1, 1994
- **Location:** emeroald:/usr2/objanal/OA/OA5.0
- **Fetures:**
  1. The local coordinate transformation is made only to the necessary points within the bucket being searched which will save quite amount of time.
  2. The nearest neighbour search routine is changed to be abale to handle any local scales which ensures that the right nearest neighbours are found independent of bucket size
  3. much faster than oax4.
- **Major Changes made to the functions main,data_local,gdist, nodesearch,bucketsearch are:**
  1. tmp_array[][] —¿tmp_array[MAXNIV]
  2. data_local moved into bucketsearch() function.
  3. find out the maximum local scale in get_neighbour() and pass a parameter lscale_max into nodesearch()
  4. replace local_scale[cutdim] with lscale_max in function.
  5. all other related function parameters passing

8.7 oa6.0

oa6.0 is the final distributed version of the oa programs. It is an upgrade of oa5.0.

- **Program name:** oax6.c
- **Release Date:** Jan. 21, 1996
- **Location:** emeroald:/usr2/objanal/OA/OA6.0
- **Fetures:**
  1. Covariance functions can be provided by user

2. Covariance matrix Inversion function is replaced with the one from Numerical Recipes
3. Index of the arraies are ajusted to start from 0 instead of 1
4. Nearest neighbour search algrithm is improved by adding "testcut" function.
5. heapsort() is renamed to hpsort() to avoid conflit with the standard library declearation for BSD system.
6. much faster than oax5.

- **Major Changes made to the functions main,get_neighbour() nodesearch,bucketsearch are:**

1. A new function testcut() is added and it is involved in bucketsearch().
2. Invert_cova() is replaced by a new one from "Numerical Recipea in C"
3. the computation for maximum local scale in get_neighbour() is deleted and the parameter passing of lscale_max into nodesearch() is eliminated.
4. all other related function parameters passing

8.8 oa AVS module

Two AVS modules was written based on oax5.c.

- **Program name:**
- **Release Date:**
- **Location:** emeroald:/usr2/objanal/OA/OA_AVS
- **Fetures:**

1.
2.
3.

8.9 oa_clib

oax C library functions are a number of functions that allow users to call within user's C application programs. oax C functions are based on the oax 5.0 code.

- **Program name:** oax_clib , it include the the following functions:
  - oax_init.c
  - oax_nearest.c
  - oax_compute.c
  - oax_exit.c
  - oax_print_data.c
  - oax_print_grid.c

- oax_print_neighbour.c
- oax_print_result.c

- **Release Date:** Feb 21, 1995
- **Location:** emeroald:/usr2/objanal/OA/OA_CLIB
- **Fetures:**

1. oax_clib has the same functions as oa version 5.0
2. more flexible and allow user to perform the optimal estimation within his application C programs interactively.
3. oax_clib can be compiled and installed in the following systems:
   - DEC alpha workstation running OFS system.
   - SUN Sparc
   - SGI Workstations Running IRIX 5.0
   - PC Pentium/90 Running FreeBSD 2.0+
   - HP 9000 HU-UX
   - IBM 6000
   - PC Pentium/90 Running Linux 1.2.1.(POSIX)
4. The minimum memory requirement depends on the data size and the grid size(if you are going to use oax_nearest() function, otherwise there will be no limitation to the number of grid). The default values are: MAXNBS = 100 MAXNIV = 10 MAXNDP = 10

8.10 oa_flib

oax Fortran library is based on oax C library. It is just an interpreter between C and fortran which provide an interface for user to call the oax C library functions within user's Fortran applications. As the Fortran and C interface programming is machine dependent, it is not possible to write a generalized Fortran library for diffrent platforms. This is the version that has been tested in emerald system at BIO (SUN Sparc 2000 Solaries 2.3 4CPUs 64MB).

- **Program name:** oa_flib, it includes the following functions:
  - oax_finit.c
  - oax_fnearest.c
  - oax_fcompute.c
  - oax_fexit.c
  - oax_fprint_data.c
  - oax_fprint_grid.c
  - oax_fprint_neighbour.c
  - oax_fprint_result.c
- **Release Date:** May 25, 1995

- **Location:** emeroald:/usr2/objanal/OA/OA_FLIB
- **Fetures:**
  1. Allow user to perform optimal estimation in user's Fortran application programs;
  2. Machine dependent.

9 Reference Manual

9.1 Subroutine Functions of oax6 - OA Stand Alone Version

9.1.1 check_data()

NAME : check_data(Datafile_name)
TYPE : integer
PURPOSE: get the number of data point from the data file
INPUT : Datafile_name – character data file name
RETURN : ndp – integer number of data point
Example: $ndp = check\_data(deck->Datafile\_name);$

9.1.2 oa_begin()

NAME : oa_begin(name,argc,argv)
TYPE : struct Flname*
PURPOSE: Start the oa program and pass the command line take down the
program starting time create log file and out file and open them
INPUT : 1. struct Flname *name – a pointer to Flname structure
2. int argc – integer number giving the number command-line arguments available.
This value is always at least 1 because the program name is counted.
3. char *argv[ ]– an array of pointers to strings. The valid subscripts for this
array are 0 througt $argc - 1$. The pointer $argv[0]$ points to the program
name(including path information), $argv[1]$ points to the first argument that
follows the program name, and so on.
RETURN : a pointer to Flname structure
Example: $name = oa\_begin(name, argc, argv);$
Note: See Structure definition for details of Flname structure.

9.1.3 oa_start()

NAME: oa_start(deck,fp1,in_name)
TYPE : struct Deck*
PURPOSE: Get the information from deck file
INPUT: 1. struct Deck *deck – pointer to Deck structure
2. FILE *fp1 – pointer to FILE type
3. char in_name[] – character (up to 80) array for storing deck file name
RETURN : a pointer to Deck structure which holds the dependent, independent,
Data filename, Scale, Grid filename, bucket and num_closest.
Example: $deck = oa\_start(deck, fp1, in\_name);$

9.1.4 data_load()

NAME: data_load(index_array,data_array,Datafile_name,num_indep,num_dep,fp1)
TYPE : void
PURPOSE: load data value from data file
INPUT: 1. char Datafile_name[80] – character array storing data file name
2. int num_indep – number of independent variables
3. int num_dep – number of dependent variables
4. FILE *fp1 – FILE pointer for log file
RETURN : 1. float data_array[i][j] – 2-D float array storing the data values
i=0,num_indep+num_dep; j=0,NDATA-1
2. int index_array[i] – 1-D array holding the data point index.
i=0,NDATA
Example: $data\_load(index\_array, data\_array, NDATA, Datafile\_name, num\_indep, num\_dep, fp1)$

9.1.5 build_kdtree()

NAME: build_kdtree(Start,Count,index_array,data_array,num_indep,
global_scale,Bucket,NDATA)
TYPE : struct kdnode*.
PURPOSE: build a kd tree from the input data points.
INPUT: 1. int Start – start number of ...
2. int Count – number of ... in each cut side.
3. int index_array[i] – 1-D array holding the data point index,
i=0,NDATA-1.
4. float data_array[i][j] – 2-D float array storing the data values,
i=0,num_indep+num_dep; j=0,NDATA-1.
5. int num_indep – number of independent variables.
6. float global_scale[i] – 1-D array holding the global scales,
i=0,num_indep-1.
7. int Bucket – maximum number of data in each bucket.
8. int NDATA – number of data point.
RETURN : struct kdnode *root – pointer to kdnode structure.
Example: $root = build\_kdtree(0, NDATA, index\_array, data\_array, num\_indep, global\_scale, Bucket, NDATA);$

9.1.6 findmaxspread()

NAME: findmaxspread(ii,nd,cutdim,data_array,num_indep,global_scale)
TYPE : void
PURPOSE: finds the dimension with the maximum (global scaled) spread.
INPUT: 1. int ii[n] – 1-D array storing the index of the data point.
2. int nd – number of data in this data set.

3. float global_scale[i] – 1-D array holding the global scales, i=0,num_indep-1.
4. float data_array[i][j] – 2-D float array storing the data values, i=0,num_indep+num_dep; j=0,NDATA-1.
6. int num_indep – number of independent variables.

RETURN: int *cutdim – the number that represents the cut dimension, *cutdim=0,1,...,num_indep-1.

Example: $findmaxspread(index\_array + Start, Count, \&cutdim, data\_array, num\_indep, global\_scale)$;

9.1.7 findmedian()

NAME: findmedian(ii,nd,cutdim,data_array)

TYPE : float

PURPOSE: finds the median value of a set of numbers

INPUT:
1. int ii[n] – 1-D array storing the index of the data point. n=0,nd.
2. float data_array[i][j] – 2-D float array storing the data values, i=0,num_indep+num_dep; j=0,NDATA-1.
3. int nd – number of data in this data set.
4. int cutdim – integer number represents the cut dimension. this number is between 0 to num_indep-1.

RETURN: a float variable which represents the median value of the given data.

Example: $cutval = findmedian(index_array + Start, Count, cutdim, data\_array)$;

9.1.8 hpsort()

NAME: hpsort(n,ra,rb)

TYPE : void

PURPOSE: heapsort method to sort the array and the index in order.

INPUT:
1. int n – number of data point.
2. float ra[n] – 1-D array holding the data value of cut dimension.
3. int rb[n] – 1-D array storing the index of data point.

RETURN: the sorted array ra[i] and index array rb[i], i=1,2,...,n.

Example: $hpsort(nd, x - 1, ii - 1)$

9.1.9 print_kdtree()

NANE: print_kdtree(node,NDATA,fp1,dflag)

TYPE : void

PURPOSE: recursively traverses and dumps the tree.

INPUT:
1. struct kdnode *node – pointer to kdnode structure.
2. int NDATA – number of data point.

3. FILE *fp1 – FILE pointer to log file that stores the tree.
4. char dflag[80] – character array holding the command line debug flag. when dflag="-d3", tree dumping is set on, otherwise off.

RETURN: tree node structure dumped to log file pointed by fp1.

Example: $print_k dtree(node, NDATA, fp1, dflag)$;

9.1.10 grid_open()

NAME: grid_open(NDATA,Gridfile_name,num_closest,fp1,dflag)

TYPE : FILE *

PURPOSE: opens the grid file, checks the number of nearest neighbour and the number of grid point.

INPUT:
1. int NDATA – number of data point.
2. char Gridfile_name[80] – character array holding the grid file name.
3. int num_closest – number of nearest neighbours.
4. FILE *fp1 – FILE pointer to log file that stores the tree.
5. char dflag[80] – character array holding the command line debug flag.

RETURN :
1. pointer to the opened grid file.
2. error messages if the number of nearest neighbours is greater than the number of data point, or it is less and equal to zero.

Example: $input = grid\_open(NDATA, Gridfile\_name, \&num\_closest, fp1, dflag)$;

9.1.11 grid_load()

NAME: grid_load(buf,local_scale,angle,target,num_indep,Gridfile_name)

TYPE : void

PURPOSE: load the coordinates of a grid point and the rotation angle

INPUT :
1. char buf[200] – character array for holding the string of a grid point coordinate.
2. num_indep – number of independent variables.
3. char Gridfile_name[80] – character array holding the grid file name.

RETURN:
1. float local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
2. float target[num_indep] – 1-D array for storing grid point coordinate.
3. float angle – clockwise rotation angle (degree) of the first two coordinate axes.

Example: $grid\_load(buf, local\_scale, \&angle, target, num_indep, Gridfile\_name)$;

9.1.12 get_neighbour()

NAME: get_neighbour(root,index_array,data_array,noise,neighbour_array,angle, num_closest,num_indep,num_dep,local_scale,obv,target, fp1,fp2,dflag)

TYPE: void

PURPOSE: get the nearest neighbours of a given grid point.

INPUT :
1. struct kdnode *root – pointer to kdnode structure.
2. float data_array[i][j] – 2-D float array storing the data values, i=0,num_indep+num_dep; j=0,NDATA-1.
3. float target[num_indep] – 1-D array for storing grid point coordinate.
4. int num_indep – number of independent variables.
5. int num_closest – number of nearest neighbours.
6. int index_array[] – 1-D array storing the index of data point.
7. double HUGEVAL – huge value for initilization of the heap.

RETURN :
1. float neighbour_array[][] – 2-D array storing the nearest neighbours.
2. float obv[][] – 2-D array storing the dependent variables.
3. double nndist[] – 1-D array storing the distances of the grid point to the nearest neighbours.

Example: *get_neighbour*(*root*, *index_array*, *data_array*, *noise*, *neighbour_array*, *angle*, *num_closest*, *num_indep*, *num_dep*, *local_scale*, *local_scale*, *obv*, *target*, *fp*1, *fp*2, *dflag*)

9.1.13 nodesearch()

NAME: nodesearch(node,nndist,nn,heapbottom,index_array,data_array,angle, num_closest,num_indep,num_dep,local_scale,target)

TYPE: void

PURPOSE: top down recursive nearest neighbour search starting from the root of the tree and going down to check each node.

INPUT :
1. struct kdnode *node – pointer to kdnode structure.
2. int *heapbottom – pointer to integer
3. int index_array[] – 1-D array storing the index of data point.
4. float data_array[][] – 2-D float array storing the data values, i=0,num_indep+num_dep; j=0,NDATA-1.
5. float angle – rotation angle of the first two coordinates.
6. int num_closest – number of nearest neighbours.
2. int num_indep – number of independent variables.
2. int num_dep – number of dependent variables.
7. float local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
8. float target[] – 1-D array holding the coordinates of the grid points.

RETURN:
1. double nndist[] – 1-D array holding distances between the grid point
2. int nn[] – 1-D array holding the index of the nearest neighbours.

Example: *nodesearch*(*root*, *nndist*, *nn*, &*heapbottom*, *index_array*, *data_array*, , *angle*, *num_closest*, *num_indep*, *num_dep*, *local_scale*, *target*);

9.1.14 bucketsearch()

NAME: bucketsearch(nndist,nn,son,start,count,heapbottom,index_array,
data_array,angle,num_closest,num_indep,
num_dep,local_scale,target)

TYPE: void

PURPOSE: check whether this is a terminal bucket, if so, search the
terminal bucket for nearest neighbours, otherwise, go back
to the nodesearch() function to search the the other node.

INPUT :
1. struct kdnode *son – pointer to kdnode structure for litle
son or greater son.
2. int start – starting point of the branch for neighbour search.
3. int count – number of data points in the branch.
4. int *heapbottom – integer counter for nearest neighbour heap
5. int index_array[] – 1-D array storing the index of data point.
5. float data_array[] – 2-D float array storing the data values,
i=0,num_indep+num_dep; j=0,NDATA-1.
6. int num_closest – number of nearest neighbours.
7. int num_indep – number of independent variables.
8. int num_dep – number of dependent variables.
9. float local_scale[i] – 1-D array storing the local scales of each
coordinate of the independent variables, i=0,1,...,num_indep-1.
10. float target[] – 1-D array holding the coordinates of the grid points.

RETURN:
1. double nndist[] – 1-D array holding distances between the grid point
2. int nn[] – 1-D array holding the index of the nearest neighbours.

Example: $bucketsearch(nndist, nn, son, start, count, heapbottom, index\_array,$
$data\_array, angle, num\_closest, num\_indep,$
$num\_dep, local\_scale, target)$;

9.1.15 gdist()

NAME: gdist(tmp_array,num_indep,local_scale,target)

TYPE: float

PURPOSE: computes the distance from a grid point in n data space to a
specified data point in local grid coordinate system.

INPUT :
1. float target[] – 1-D array holding the coordinates of the grid points.
2. int num_indep – number of independent variables.
3. float tmp_array[] – 1-D array for storing the data point coordinates.
4. local_scale[i] – 1-D array storing the local scales of each
coordinate of the independent variables, i=0,1,...,num_indep-1.

RETURN: float sqare root of locally scaled distance between two points.

NOTE: the target array holds the global coordinates of grid points.
the tmp_array holds the transformed local coordinates of
data points which are originated at grid point. Therefore,
target[] should always be zero in the distance calculation.

that's why we have (for k=0,1):
ddist = tmp_array[k]/local_scale[k];
not
ddist=(target[k]-tmp_array[k])/local_scale[k];

Example: $testdist = gdist(tmp\_array, num\_indep, local\_scale, target)$;

9.1.16 testcut()

NAME: testcut(cutdim,cutval,target,angle,local_scale)
TYPE: float
PURPOSE : computes the minimum possible local distance from a grid point and any point with coordinate cutdim having a value cutval
INPUT : 1. int cutdim – integer for holding the cut dimension value.
2. float cutval – a float variable which represents the median value of the given data
3. flaot target[ ] – 1-D array holding the coordinates of the grid points.
4. float angle – rotation angle of the first two coordinates.
5. local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
RETURN : the minimum possible local distance
Example: $tcdis = testcut(cutdim, cutval, target, angle, local\_scale)$
NOTE : target[] holds the global coordinates of a grid point.
local_scale[i] holds the local scales associated with target[].

9.1.17 heapinsert()

NAME: heapinsert(nndist,nn,heapbottom,dis,index)
TYPE: void
PURPOSE: insert new value onto nearest neighbour heap
INPUT : 1. *heapbottom – integer counter for nearest neighbour heap.
2. dis – double variable holds the test distance passed in from the calling function.
3. index – integer number holding the index of the data point passed in from the calling function.
RETURN: 1. nndist[] – double 1-D array holds the distances of the nearest neighbours.
2. nn[] – integer 1-D array holds the index of the nearest neighbours.
Example: $heapinsert(nndist, nn, heapbottom, testdist, i)$;
NOTE: heapinsert inserts a huge value to the top of the heap first and then keep inserting the new value to the heap.

9.1.18 heapreplace()

NAME: heapreplace(nndist,nn,heapbottom,dis,index)
TYPE: void
PURPOSE: replace top of heap with new value and then percolate new value down to where it belongs. This function is called only when *heapbottom=num_closest that is when the points searched reached the number which is equal to the number of nearest neighbours given then this function is called to reploce the top of the heap nndist[1]. nndist[1] is not changed before. It is always HUGEVAL/10.0 before this function is called.
INPUT : 1. *heapbottom – integer counter for nearest neighbour heap.
2. dis – double variable holds the test distance passed in from the calling function.
3. index – integer number holding the index of the data point.
RETURN: 1. nndist[] – double 1-D array holds the distances of the nearest neighbours.
2. nn[] – integer 1-D array holds the index of the nearest neighbours.
Example: $heapreplace(nndist, nn, heapbottom, testdist, i)$;

9.1.19 heapremove()

NAME: heapremove(nndist,nn,heapbottom,index,v)
TYPE: void
PURPOSE: remove item from top of the heap and then fix the heap
INPUT : 1. *heapbottom – integer counter for nearest neighbour heap.
2. index – integer number holding the index of the data point.
3. v – double variable temporarily holds the distance passed in from the calling function.
RETURN: 1. nndist[] – double 1-D array holds the distances of the nearest neighbours.
2. nn[] – integer 1-D array holds the index of the nearest neighbours.
Example: $heapremove(nndist, nn, \&heapbottom, \&index, \&value)$;

9.1.20 heapup()

NAME: heapup(nndist,nn,upitem)
TYPE: void
PURPOSE: percolate a value up to its proper level in the heap
INPUT : 1. nndist[ ] – double 1-D array holds the distances of the nearest. neighbours.
2. nn[] – integer 1-D array holds the index of the nearest neighbours.
3. upitem – integer counter of the nearest neighbours.
RETURN: 1. nndist[] – double 1-D array holds the distances of the nearest.

2. nn[] – integer 1-D array holds the index of the nearest neighbours.

Example: $heapup(nndist, nn, *heapbottom)$;

9.1.21 heapdown()

NAME: heapdown(nndist,nn,heapbottom,downitem)

TYPE: void

PURPOSE: percolate a value down to its correct level in the heap

INPUT : 1. nndist[ ] – double 1-D array holds the distances of the nearest.
2. nn[ ] – integer 1-D array holds the index of the nearest neighbours.
3. *heapbottom – integer counter for nearest neighbour heap.
4. downitem – integer counter of the nearest neighbour heap that may be re-evaluated in the process.

OUTPUT: 1. nndist[] – double 1-D array holds the distances of the nearest.
2. nn[] – integer 1-D array holds the index of the nearest neighbours.

Example: $heapdown(nndist, nn, heapbottom, (int)0)$;

9.1.22 data_local()

NAME: data_local(i,num_indep,angle,data_array,index_array,tmp_array,target)

TYPE: void

PURPOSE: At this particular grid point transform the given data point to a local coordinate system corresponding to this grid point local coordinate system (a translation + rotation in XY plane).

INPUT: 1. int i – integer counter of the data points in the searching bucket.
2. float data_array[i][j] – 2-D float array storing the data values, i=0,num_indep+num_dep; j=0,NDATA-1.
3. int num_indep - number of independent variables.
4. float tmp_array [] – 1-D array for storing transformed grid point coordinate.
5. float target[] – 1-D array for storing grid point coordinate.
6. float angle – rotation angle of the first two coordinates.
7. float RAD – global variable defined as the radius.

RETURN : 1. transformed tmp_array[] in local frame of the given grid. Here only data_array[0][id] and data_array[1][id] are transformed. The other elements of the neighbour array do not change because the transformation only takes place in in XY plane. Once the data array becomes local, we can search the nearest neighbours by local scale.

Example: $data\_local(i, num\_indep, angle, data\_array, index\_array, tmp\_array, target)$;

9.1.23 become_local()

NAME: become_local(num_indep,angle,neighbour_array,num_closest,target)

TYPE: void

PURPOSE: at this particular grid point transform all coordinates of the nearest neighbours obtained from kd tree global coordinate system to a local coordinate system corresponding to this grid point local coordinate system (a translation + rotation in XY plane).

INPUT:
1. int num_indep – numbr of independent variables.
2. angle – rotation angle.
3. neighbour_array[][] – 2-D array holding the nearest neighbours.
4. num_closest – number of nearest neighbours.
5. target[] – 1-D array for storing grid point coordinate.

RETURN : float neighbour_array[][] – transformed 2-D array in local frame. Here only neighbour_array[0][id] and neighbour_array[1][id] are transformed. The other elements of the neighbour array do not change because the transformation only takes place in XY plane. Once the neighbour array becomes local, we can calculate the covariance matrix and covariance vector for the local coordinate frame using the same functions as before.

Example: $become\_local(num\_indep, angle, neighbour\_array, num\_closest, target)$;

9.1.24 cova_matrix()

NAME: cova_matrix(A,ivA,noise,local_scale,neighbour_array,num_closest, num_indep,fp1,dflag)

TYPE: void

PURPOSE: generate the covariance matrix of upper triangle (The covarianve matrix is always symmetrical)

INPUT:
1. float noise – float variable holds the noise value.
2. float local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
3. float neighbour_array[][] – transformed 2-D array in local frame.
4. num_closest – number of nearest neighbours.
5. num_indep – number of independent variables.
6. fp1 – FILE pointer.
7. dflag[] – character string holding the command line debug flag.

RETURN : covariance matrix A[][]

Example: $cova\_matrix(A, ivA, noise, local\_scale, neighbour\_array, num\_closest, num\_indep, fp1, dflag)$;

9.1.25 cova_func()

NAME: cova_func(id1,pt1,id2,pt2,A,noise,local_scale,num_indep)

TYPE: void

PURPOSE: calculate the covariance function of a given model

INPUT:
1. id1 – integer identifier for data points pt1.
2. id2 – integer identifier for data points pt2.

id1=1,2,3,...,NDATA; id2=1,2,3,...,NDATA;
3. pt1[] – stores the coordinates of pt1.
4. pt2[] – stores the coordinates of pt2.
5. noise – float variable holds the noise value.
6. local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
7. num_indep – number of independent variables.

RETURN : an element of covariance matrix A[id1][id2]

Example: $cova\_func(id1, pt1, id2, pt2, A, noise, local\_scale, num\_indep)$;

9.1.26 ldist()

NAME: ldist(pt1,pt2,local_scale,num_indep)

TYPE: float

PURPOSE: Calculate the distance between two points pt1 and pt2.

INPUT :
1. int num_indep – number of independent variables.
2. pt1[] – stores the coordinates of pt1.
3. pt2[] – stores the coordinates of pt2.
4. local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.

RETURN: float r – distance between pt1 and pt2.

Example: $r = ldist(gpt, dpt, local\_scale, num\_indep)$;

9.1.27 invert_cova_matrix()

NAME: invert_cova_matrix(A,num_closest,fp1,dflag)

TYPE: void

PURPOSE : invert the symmetrical possitive definite matrix using LU decomposition method (see Numerical Recipes in C, Second Edition, Section 2.3).

INPUT :
1. float *A[] – 2-D array storing the covariance matrix in the begining.
2. int num_closest – number of nearest neighbours.
3. FILE *fp1 – FILE pointer
4. dflag[] – character string holding the command line debug flag.

RETURN: float A[][] – 2-D array storing the inverse of covariance matrix (overwritten to input A[][]).

Example: $invert_cova_matrix(ivA, num\_closest, fp1, dflag)$;

9.1.28 cova_vector()

NAME: cova_vector(C,local_scale,neighbour_array,num_closest,num_indep, target,fp1,dflag)

TYPE: void

PURPOSE: calculate the covariance vector.

INPUT: 1. float local_scale – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
2. neighbour_array[][] – 2-D array for storing the coordinates of the nearest neighbours.
3. num_closest – integer number of nearest neighbours.
4. num_indep – number of independent variables.
5. target[] – 1-D array for storing grid point coordinate.
6. FILE *fp1 – FILE pointer
7. dflag[] – character string holding the command line debug flag.

RETURN : float C[] – 1-D array holding covariance vector.

Example: $cova_vector(C, local\_scale, neighbour\_array, num\_closest, num_indep, target, fp1, dflag);$

9.1.29 cova_vecfunc()

NAME: cova_vecfunc(id,gpt,dpt,C,local_scale,num_indep)

TYPE: void

PURPOSE: calculate the covariance vector function of a given model.

INPUT: 1. int id – index identifiers for data points pt1 and pt2. id=1,2,3,...,num_closest;
2. float gpt[] – store the coordinates of grid point.
3. float dpt[] – store the coordinates of data point.
4. float local_scale[i] – 1-D array storing the local scales of each coordinate of the independent variables, i=0,1,...,num_indep-1.
5. int num_indep – number of independent variables.

RETURN : float C[id] – an element of covariance vector.

Example: $cova\_vecfunc(id, gpt, dpt, C, local\_scale, num\_indep);$

9.1.30 obtain_weight()

NAME: obtain_weight(ivA,w,est_mean,num_closest,Method_name, num_dep,obv,fp1,dflag)

TYPE: void

PURPOSE: obtain the weighting functions.

INPUT: 1. float *ivA[] – 2-D array storing the inverse of covariance matrix.
2. float est_mean[] – 1-D array storing estimated mean.
3. num_closest – integer number of nearest neighbours.
4. char Method_name[] – character array storing the method name key word.
5. num_dep – number of dependent variables.
6. float obv[][MAXNDV] – 2-D array storing dependent variables.
7. FILE *fp1 – FILE pointer
8. dflag[] – character string holding the command line debug flag.

RETURN : float w[kd][id] – 2-D array storing the weighting function.

Example: $obtain_weight(ivA, w, est\_mean, num\_closest, Method\_name, num_dep,$

$obv, fp1, dflag);$

9.1.31 calculate_weight()

NAME:	calculate_weight(ivA,id,w,ivAsum,num_closest,num_dep,obv)
TYPE:	void
PURPOSE:	calculate the weighting function
INPUT:	1. float *ivA[] – 2-D array storing the inverse of covariance matrix.
	2. id – integer index of nearest neighbour point.
	3. num_closest – integer number of nearest neighbours.
	4. num_dep – number of dependent variables.
	5. obv[idd][kd] – 2-D array storing dependent variables.
RETURN :	1. w[kd][id] – 2-D array storing the weighting function.
	2. ivAsum[kd][id] – 2-D array storing the sum of inverse covariance matrix.
Example:	$calculate_weight(ivA, id, w, ivAsum, num\_closest, num\_dep, obv);$

9.1.32 optimal_estimation()

NAME:	optimal_estimation(C,w,est,est_mean,num_closest,num_dep,
	Method_name,fp1,dflag)
TYPE:	void
PURPOSE:	performs the optimal estimation.
INPUT:	1. C[id] – 1-D array storing the covariance vector.
	2. w[kd][id] – 2-D array storing the weighting function.
	3. est_mean[kd] – 1-D array storing estimated mean.
	4. num_closest – integer number of nearest neighbours.
	5. num_dep – number of dependent variables.
	6. char Method_name[] – character array storing the method name key word.
	7. FILE *fp1 – FILE pointer.
	8. dflag[] – character string holding the command line debug flag.
RETURN :	float est[] – 1-D array
Example:	$optimal\_estimation(C, w, est, est\_mean, num\_closest, num\_dep,$

9.1.33 error_estimate()

NAME:	error_estimate(ivA,C,error,num_closest,Method_name,fp1,dflag)
TYPE:	void
PURPOSE:	calculate the error estimate
INPUT:	1. float *ivA[] – 2-D array storing the inverse of covariance matrix.
	2. float C[id] – 1-D array storing the covariance vector.
	3. num_closest – integer number of nearest neighbours.
	4. char Method_name[] – character array storing the method name key word.
	5. FILE *fp1 – FILE pointer.

6. dflag[] – character string holding the command line debug flag.

RETURN : float error[] – 1-D array

Example: $error\_estimate(ivA, C, error, num\_closest, Method\_name, fp1, dflag)$;

9.1.34 oa_output()

NAME: oa_output(est,error,num_dep,fp2)

TYPE: void

PURPOSE: output the estimation result and the error

INPUT: 1. int num_dep – number of dependent variables.
2. FILE *fp2 – FILE pointer.

RETURN : 1. float est[kd] – 1-D array storing the optimal estimation results.
2. float error[kd] – 1-D array storing the optimal errors.

Example: $oa\_output(est, error, num\_dep, fp2)$;

9.1.35 message()

NAME: message(key,dflag,fp1)

TYPE: void

PURPOSE: echo the operation message

INPUT : 1. int key – key switch for different cases.
2. char dflag[] – character string holding the command line debug flag.
3. FILE *fp1 – FILE pointer.

RETURN : messages to standard output and log file pointed by FILE pointer fp1.

Example: $message(0, "none", fp1)$;

9.1.36 oa_end()

NAME: oa_end(ngp,name)

TYPE: void

PURPOSE: end the oa program with closing the opened files, echoing the number of grid read and sending the message back to the user.

INPUT: 1. int ngp – number of grid point.
2. struct Flname *name – pointer to Flname structure.

RETURN : message to screen

Example: $oa\_end(ngp, name)$;

9.2 OA C Library Functions

9.2.1 oax_init()

NAME : oax_init(NDATA,data_array,num_indep,num_dep,Bucket,global_scale)
TYPE : struct Oax_Wsa *
PURPOSE: initialize the oax, allocate memories and build the kd-tree.
INPUT : 1. NDATA : number of data point (integer), NDATA >= 1.
There is no limit to the number of data if the computer resource allow.
2. num_indep: integer number holding the number of independent variables,
1=< num_indep <=10.
3. num_dep : integer number holding the number of dependent variables,
1=< num_dep <=10.
4. data_array[i][j]: 2-D array for holding independent variables,
dependent variables and the noise.
i=0,1,2,...,NDATA-1;
j=0,1,2,...,num_indep+num_dep.
An example of data array look like

0.000000 0.000000 13.000000 12.000000 0.100000
0.000000 1.000000 14.000000 11.000000 0.100000
0.000000 2.000000 15.000000 10.000000 0.100000
1.000000 0.000000 16.000000 9.000000 0.100000
1.000000 1.000000 17.000000 8.000000 0.100000
1.000000 2.000000 18.000000 7.000000 0.100000
2.000000 0.000000 19.000000 6.000000 0.100000
2.000000 1.000000 20.000000 5.000000 0.100000
2.000000 2.000000 21.000000 4.000000 0.100000

In this example:
NDATA=9 (nine lines of data)
num_indep=2 (first two columns)
num_dep = 2 (third and fourth columns)
noise = 0.1 (last columns)
NOTE:
noise is the variance of errors (may be local sampling error or instrumental uncertainty). Here we assume that the measurement of all dependent variables have the same noise level.
gl_scale(i) : 1-D float array for storing the global scales of each independent variables of the data point.
i=0,2,...,num_indep-1
gl_scale is used in building kd tree.
bucket : integer number holding the bucket size of kd-tree, bucket >= 1.
bucket determines the size of the terminal bucket of the kd tree. A suitable number of bucket will significantly increase the speed of the program.
the speed of the program.
RETURN : oax_wsa (pointer to oax_wsa structure)

where oax_wsa is the pointer to oax working storage
area structure Oax_Was.
struct Oax_Wsa
{
struct kdnode *root;
int num_dep,num_indep,num_nearest,
NDATA,Bucket,*index_array;
float **data_array;
}

Example: $oax\_init(NDATA, data\_array, num\_indep, num\_dep, Bucket, global\_scale)$

9.2.2 oax_nearest()

NAME : oax_nearest(oax_wsa,NGRID,grid_array,num_nearest,neighbours)
TYPE : void
PURPOSE: search the nearest neighbours from kd-tree.
INPUT : oax_wsa : pointer to Oax_Wsa structure
NGRID : number of grid point.
grid_array : 2-D array storing the coordinates f grid points, the rotation angle and the local scales for each grid points.
num_nearest : number of nearest neighbours.
RETURN : neighbours[i][j][k], 3-D array holding the coordinates of the num_nearest nearest neighbours for each grid point.
i=0,1,2, ..., NGRID-1;
j=0,1,2, ..., num_indep-1;
k=0,1,2, ..., num_nearest-1.
An example of grid array look like

1.500000 0.500000 35.000000 3.000000 4.000000
0.500000 1.500000 35.000000 4.000000 5.000000
In this example:
NGRID = 2 (two lines of data)
num_indep=2 (first two columns)
rotation angles (third column)
local scales (fourth and fifth columns)

Example: $oax\_nearest(oax\_wsa, NGRID, grid\_array, num\_nearest, neighbours)$;

9.2.3 oax_compute()

NAME : oax_compute(oax_wsa,NGRID,grid_array,num_nearest,Method_name,result)
TYPE : void
PURPOSE: 1. finds the nearest neighbours of the given grid points;
2. perform the optimal estimation.
INPUT : 1. oax_wsa : pointer to Oax_Wsa structure

2. NGRID : number of grid point
3. grid_array : 2-D array storing the coordinates of grid points,
the rotation angle and the local scales for each grid points.
4. num_nearest : number of nearest neighbours.
5. method : caracter variable for holding the estimation method options:
1) "EST_MEAN"
2) "ANOMALY"

RETURN : result[i][j], 2-D float array holds the optimal estimation results and errors.
i=0,1,2, ..., NGRID-1;
j=0,1,2, ..., num_dep.

Example: $oax\_compute(oax\_wsa, NGRID, grid\_array, num\_nearest, Method\_name, result)$;

9.2.4 oax_exit()

NAME : oax_exit(oax_wsa)
TYPE : void
PURPOSE: frees up the memory allocated and exit the program
INPUT : oax_wsa — pointer to Oax_Was structure
RETURN : none
Example: $oax\_exit(oax\_wsa)$;

9.2.5 oax_print_data()

NAME : oax_print_data(oax_wsa,start_index,NDISPLAY)
TYPE : void
PURPOSE: print out the data array.
INPUT : oax_wsa – pointer to Oax_Wsa structure.
start_index – the first element of data_array to start printing.
NDISPLAY : number of points to print.
RETURN : standard output of data array.
Example: $oax\_print\_data(oax\_wsa, start\_index, NDISPLAY)$

9.2.6 oax_print_grid()

NAME : oax_print_grid(oax_wsa,start_index,NDISPLAY,NGRID,grid_array)
TYPE : void
PURPOSE: print out the grid array.
INPUT : oax_wsa – pointer to Oax_Was structure.
start_index – the first element of grid_array to start printing.
NDISPLAY – number of points to print.
NGRID – number of grid points.
grid_array : 2-D array storing the coordinates of grid points, the
rotation angle and the local scales for each grid points.

RETURN : standard output of grid array.
Example: $oax\_print\_grid(oax\_wsa, start\_index, NDISPLAY, NGRID, grid\_array)$

9.2.7 oax_print_neighbour

NAME : oax_print_neighbour(oax_wsa,neighbours)
TYPE : void
PURPOSE: print out the neighbour array.
INPUT : oax_wsa – pointer to Oax_Wsa structure
neighbours: 3-D array holding the nearest neighbours.
RETURN : standard output of nearest neighbour array.
Example: $oax\_print\_neighbour(oax\_wsa, neighbours)$

9.2.8 oax_print_result

NAME : oax_print_result(oax_wsa,result)
TYPE : void
PURPOSE: print out the optimal estimation results.
INPUT : oax_wsa – pointer to Oax_Wsa structure
result – 2-D array storing the optimal estimated results and errors.
RETURN : standard output of grid array.
Example: $oax\_print\_result(oax\_wsa, result)$

9.3 OA F77 Library Functions

OA F77 library functions are actually a number of external C functions that are called within the Fortran program. As the function calls between Fortran and Fortran functions are machine dependent, the OA F77 library functions are only for the SUN Spac systems.

9.3.1 oax_finit()

NAME : oax_finit(NDATA,data_array,num_indep,num_dep,Bucket,global_scale,context)
PURPOSE: initialize the oax, allocate memories and build the kd-tree.
INPUT : 1. NDATA : number of data point (integer), NDATA >= 1.
There is no limit to the number of data if the computer resource allow.
2. num_indep: integer number holding the number of independent variables,
1=< num_indep <=10.
3. num_dep : integer number holding the number of dependent variables,
1=< num_dep <=10.
4. data_array[i][j]: 2-D array for holding independent variables,
dependent variables and the noise.
i=0,1,2,...,NDATA-1;
j=0,1,2,...,num_indep+num_dep.
An example of data array look like

0.000000 0.000000 13.000000 12.000000 0.100000
0.000000 1.000000 14.000000 11.000000 0.100000
0.000000 2.000000 15.000000 10.000000 0.100000
1.000000 0.000000 16.000000 9.000000 0.100000
1.000000 1.000000 17.000000 8.000000 0.100000
1.000000 2.000000 18.000000 7.000000 0.100000
2.000000 0.000000 19.000000 6.000000 0.100000
2.000000 1.000000 20.000000 5.000000 0.100000
2.000000 2.000000 21.000000 4.000000 0.100000

In this example:
NDATA=9 (nine lines of data)
num_indep=2 (first two columns)
num_dep = 2 (third and fourth columns)
noise = 0.1 (last columns)
NOTE:
noise is the variance of errors (may be local sampling error or instrumental uncertainty). Here we assume that the measurement of all dependent variables have the same noise level.
gl_scale(i) : 1-D float array for storing the global scales of each independent variables of the data point.
i=0,2,...,num_indep-1
gl_scale is used in building kd tree.
bucket : integer number holding the bucket size of kd-tree, bucket >= 1.
bucket determines the size of the terminal bucket of the kd tree. A suitable

number of bucket will significantly increase the speed of the program.
the speed of the program.

RETURN : context : interger number for passing the pointer to work storage area
where oax_wsa is a C pointer to oax working storage
area structure Oax_Was.
struct Oax_Wsa
{
struct kdnode *root;
int num_dep,num_indep,num_nearest,
NDATA,Bucket,*index_array;
float **data_array;
}

Example: *call oax_finit*$(NDATA, data\_array, num\_indep, num\_dep, Bucket, global\_scale, context)$

9.3.2 oax_fnearest()

NAME : oax_fnearest(context,NGRID,grid_array,num_nearest,neighbours)

PURPOSE: search the nearest neighbours from kd-tree.

INPUT : context : interger number for passing the pointer to work storage area
structure Oax_Wsa between fortran and C programs.
NGRID : number of grid point.
grid_array : 2-D array storing the coordinates f grid points, the rotation
angle and the local scales for each grid points.
num_nearest : number of nearest neighbours.

RETURN : neighbours(i,j,k), 3-D array holding the coordinates of the
num_nearest nearest neighbours for each grid point.
i=1,2, ..., NGRID;
j=1,2, ..., num_indep;
k=1,2, ..., num_nearest.
An example of grid array look like

1.500000 0.500000 35.000000 3.000000 4.000000
0.500000 1.500000 35.000000 4.000000 5.000000
In this example:
NGRID = 2 (two lines of data)
num_indep=2 (first two columns)
rotation angles (third column)
local scales (fourth and fifth columns)

Example: *call oax_fnearest*$(context, NGRID, grid\_array, num\_nearest, neighbours)$;

9.3.3 oax_fcompute()

NAME : oax_fcompute(context,NGRID,grid_array,num_nearest,Method_name,result)

PURPOSE: 1. finds the nearest neighbours of the given grid points;

2. perform the optimal estimation.

INPUT : 1. context : interger number for passing the pointer to work storage area structure Oax_Wsa between fortran and C programs.
2. NGRID : number of grid point
3. grid_array : 2-D array storing the coordinates of grid points,
the rotation angle and the local scales for each grid points.
4. num_nearest : number of nearest neighbours.
5. method : caracter variable for holding the estimation method options:
1) "EST_MEAN"
2) "ANOMALY"

RETURN : result(i,j), 2-D float array holds the optimal estimation results and errors.
i=1,2, ..., NGRID;
j=1,2, ..., num_dep+1.

Example: *call oax_fcompute(context, NGRID, grid_array, num_nearest, Method_name, result);*

9.3.4 oax_fexit()

NAME : oax_fexit(context)

PURPOSE: frees up the memory allocated and exit the program

INPUT : context : interger number for passing the pointer to work storage area structure Oax_Wsa between fortran and C programs.

RETURN : none

Example: *call oax_fexit(context);*

9.3.5 oax_fprint_data()

NAME : oax_fprint_data(context,start_index,NDISPLAY)

PURPOSE: print out the data array.

INPUT : context : interger number for passing the pointer to work storage area structure Oax_Wsa between fortran and C programs.
start_index – the first element of data_array to start printing.

NDISPLAY : number of points to print.

RETURN : standard output of data array.

Example: *call oax_fprint_data(context, start_index, NDISPLAY)*

9.3.6 oax_fprint_grid()

NAME : oax_fprint_grid(context,start_index,NDISPLAY,NGRID,grid_array)

PURPOSE: print out the grid array.

INPUT : context : interger number for passing the pointer to work storage area structure Oax_Wsa between fortran and C programs.
start_index – the first element of grid_array to start printing.
NDISPLAY – number of points to print.

NGRID – number of grid points.
grid_array : 2-D array storing the coordinates of grid points, the
rotation angle and the local scales for each grid points.

RETURN : standard output of grid array.

Example: *call oax_fprint_grid(context, start_index, NDISPLAY, NGRID, grid_array)*

9.3.7 oax_fprint_neighbour

NAME : oax_fprint_neighbour(context,neighbours)

PURPOSE: print out the neighbour array.

INPUT : context : interger number for passing the pointer to work storage area
structure Oax_Wsa between fortran and C programs.
neighbours: 3-D array holding the nearest neighbours.

RETURN : standard output of nearest neighbour array.

Example: *call oax_fprint_neighbour(context, neighbours)*

9.3.8 oax_fprint_result

NAME : oax_fprint_result(context,result)

PURPOSE: print out the optimal estimation results.

INPUT : context : interger number for passing the pointer to work storage area
structure Oax_Wsa between fortran and C programs.
result – 2-D array storing the optimal estimated results and errors.

RETURN : standard output of grid array.

Example: *call oax_fprint_result(context, result)*

9.4 Subroutine Functions kd-tree Store Program tdstore

9.4.1 kd_begin()

NAME : kd_begin(name,argc,argv)
TYPE : struct Flname*
PURPOSE: Start the oa program and pass the command line take down the program starting time create log file and out file and open them
INPUT : 1. struct Flname *name – a pointer to Flname structure
2. int argc – integer number giving the number command-line arguments available. This value is always at least 1 because the program name is counted.
3. char *argv[]– an array of pointers to strings. The valid subscripts for this array are 0 througt $argc - 1$. The pointer $argv[0]$ points to the program name(including path information), $argv[1]$ points to the first argument that follows the program name, and so on.
RETURN : a pointer to Flname structure
Example: $name = kd\_begin(name, argc, argv);$
Note: See Structure definition for details of Flname structure.

9.4.2 kd_start()

NAME: kd_start(deck,fp1,in_name)
TYPE : struct Deck*
PURPOSE: Get the information from deck file
INPUT: 1. struct Deck *deck – pointer to Deck structure
2. FILE *fp1 – pointer to FILE type
3. char in_name[] – character (up to 80) array for storing deck file name
RETURN : a pointer to Deck structure which holds the dependent, independent, Data filename, Scale, Grid filename, bucket and num_closest.
Example: $deck = kd\_start(deck, fp1, in\_name);$

9.4.3 ascii_store_kdtree()

NAME: ascii_store_kdtree()
TYPE : void
PURPOSE: 1. recursively traverses and dumps the tree(ASCII MODE).
INPUT: tree node
RETURN : kdnode structure
Example: $ascii_store_kdtree(node- > lt_son);$

9.4.4 binary_store_kdtree()

NAME: binary_store_kdtree()
TYPE : void

PURPOSE: 1. recursively traverses and dumps the tree(BINARY MODE).
INPUT: tree node
RETURN : kdnode structure
Example: $binary_store_kdtree(node->lt_son);$

9.5 Subroutine Functions kd-tree load Program tdsload

9.5.1 kd_begin()

NAME : kd_begin(name,argc,argv)
TYPE : struct Flname*
PURPOSE: Start the oa program and pass the command line take down the
program starting time create log file and out file and open them
INPUT : 1. struct Flname *name – a pointer to Flname structure
2. int argc – integer number giving the number command-line arguments available.
This value is always at least 1 because the program name is counted.
3. char *argv[]– an array of pointers to strings. The valid subscripts for this
array are 0 througt $argc - 1$. The pointer $argv[0]$ points to the program
name(including path information), $argv[1]$ points to the first argument that
follows the program name, and so on.
RETURN : a pointer to Flname structure
Example: $name = kd\_begin(name, argc, argv);$
Note: See Structure definition for details of Flname structure.

9.5.2 kd_start()

NAME: kd_start(deck,fp1,in_name)
TYPE : struct Deck*
PURPOSE: Get the information from deck file
INPUT: 1. struct Deck *deck – pointer to Deck structure
2. FILE *fp1 – pointer to FILE type
3. char in_name[] – character (up to 80) array for storing deck file name
RETURN : a pointer to Deck structure which holds the dependent, independent,
Data filename, Scale, Grid filename, bucket and num_closest.
Example: $deck = kd\_start(deck, fp1, in\_name);$

9.5.3 ascii_load_kdtree()

NAME: ascii_load_kdtree()
TYPE : struct kdnode *
PURPOSE: 1. recursively traverses and loads the prebuild tree(ASCII MODE).
2. Performs nearest neighbour search in the loaded tree.
INPUT: kdnode structure
RETURN : kdnode structure

Example: $root = ascii_load_kdtree();$

9.5.4 binary_load_kdtree()

NAME: binary_load_kdtree()
TYPE : struct kdnode *
PURPOSE: 1. recursively traverses and loads the prebuild tree(BINARY MODE).
2. Performs nearest neighbour search in the loaded tree.
INPUT: kdnode structure
RETURN : kdnode structure
Example: $root = binary_load_kdtree();$

9.6 OA AVS Module Functions

9.6.1 oax_2d

NAME : oax_2d()
TYPE : AVS Filter Module
PURPOSE: Performs optimal estimation and generates AVS field.
INPUT : input 0 "mesh_in" field float REQUIRED
input 1 "data_in" field 1D float REQUIRED
param 0 "Num_Nearest" idial 20 1 500
param 1 "Bucket_size" idial 500 1 20
param 2 "tree" radio_buttons "build" "build:load" ":"
RETURN : output 0 "oax_out" field 3D 3-space float

9.6.2 oax_3d

NAME : oax_3d()
TYPE : AVS Filter Module
PURPOSE: Performs optimal estimation and generates AVS field.
INPUT : input 0 "mesh_in" field float REQUIRED
input 1 "data_in" field 1D float REQUIRED
param 0 "Num_Nearest" idial 20 1 500
param 1 "Bucket_size" idial 500 1 20
param 2 "tree" radio_buttons "build" "build:load" ":"
RETURN : output 0 "oax_out" field float