Generating Sophisticated Spatial Surrogates Using the MIMS Spatial Allocator

Alison M. Eyth
Carolina Environmental Program, University of North Carolina at Chapel Hill
137 E. Franklin St., Chapel Hill, NC 27599-6116
eyth@unc.edu

William Benjey*
Atmospheric Sciences Modeling Division, Atmospheric Resources Laboratory
National Oceanic and Atmospheric Administration
Research Triangle Park, NC 27711
benjey.william@epa.gov

* In partnership with the U.S. Environmental Protection Agency,
National Exposure Research Laboratory

ABSTRACT

The Multimedia Integrated Modeling System (MIMS) Spatial Allocator is open-source software for generating spatial surrogates for emissions modeling, changing the map projection of Shapefiles, and performing other types of spatial allocation that does not require the use of a commercial Geographic Information System (GIS). The December 2003 version of the Spatial Allocator was able to generate basic point-, line-, and polygon-based surrogates for modeling grids from data contained in Shapefiles. In January 2005, a new version of the Spatial Allocator became available with features that allow it to reproduce the surrogates that are currently being used by EPA. The new features included in this version are the following: subsets of Shapefiles can be defined and used to generate surrogates or create smaller Shapefiles, surrogates can be generated based on a function of multiple attributes in one or more Shapefiles, surrogates can be generated using weighted sums of other surrogates, and new surrogates can be made by filling in the gaps of less comprehensive surrogates. An additional release of the Spatial Allocator will be made in June 2005. This version will be able to create the inputs for biogenic emissions processing that are required by the Sparse Matrix Operator Kernel Emissions (SMOKE) modeling system; perform general spatial allocation functions, such as mapping gridded data to and from county-level data, mapping from grid to grid, and aggregating data from census tract to county levels; and print the attributes of shapes that are overlapped by a grid, bounding box, or set of polygons.

INTRODUCTION

The MIMS Spatial Allocator provides an alternative to using commercial GISs or statistical software for generating spatial surrogates that are input to emissions models. A drawback of using commercial software to generate surrogates is that the software can cost thousands of dollars. In addition, these software systems are very complex and can be difficult to learn and use, and require custom coding to generate spatial surrogates. The MIMS Spatial Allocator is a comparatively small piece of software. It is written in C, and can run on Windows or Unix systems. It is provided free and uses ESRI® Shapefiles—a standard in the GIS industry—as input databases. Surrogates can be generated for point-, line-, or polygon-based data, sets such as ports, airports, housing, population, agriculture, water area, and railroads.

Shapefiles (see http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf) are the primary type of data file processed by the Spatial Allocator. A Shapefile is actually a set of three files: .shp, .shx, and .dbf. The .shp part of the file set contains the coordinates of the shapes. The attributes of the Shapefile are stored in the .dbf part of the file set. Note that .dbf files can be opened by many spreadsheet programs. There can be various types of attributes in a .dbf file. Common types of attributes are strings,

integers, and real numbers. Examples of attributes are the name of the shape, the FIPS state and county code, population, and area. The .shx part of the file set contains a mapping between the .shp and .dbf files.

To create a surrogate, the Spatial Allocator uses a data file, a GRIDDESC file, and a weight file. The data file is a Shapefile that typically consists of county polygons. The GRIDDESC file contains descriptions of one or more air quality model grids, from which one is selected for a particular run of the Spatial Allocator. The weight file is a Shapefile that contains "shapes" that are points, lines, or polygons. These shapes are used to allocate an attribute in the weight file into the grid cells used by the air quality model. The resulting file contains a "surrogate" that is used by an emissions model (e.g., Sparse Matrix Operator Kernel Emission (SMOKE)) to allocate the emissions in a county into appropriate grid cells according to a surrogate type appropriate to that emissions category. For example, dry cleaning emissions are typically allocated using a population surrogate, whereas recreational boat emissions would be allocated based on a water body surrogate. The values for a surrogate typically should sum to 1 for each county. After processing by an emissions model, the gridded, speciated, time-varying emissions are provided as an input to an air quality model.

The Spatial Allocator creates spatial surrogates for regularly spaced air quality model grids on a variety of map projections, such as Universal Transverse Mercator (UTM), Lambert Conformal, Mercator, polar stereographic, stereographic, and latitude-longitude. The Proj.4 library (http://proj.maptools.org) is used to specify the map projections and ellipsoids used by input Shapefiles. It supports a variety of standard and user-specified ellipsoids to approximate the shape of the earth. In addition to creating emission surrogates, other types of spatial computations can be performed by the tool, such as aggregation. The Spatial Allocator bridges the gap between emission modelers and GIS analysts by providing a zero-cost, easy-to-use, powerful, and portable system for generating spatial surrogates, changing the map projection of Shapefiles, filtering Shapefiles, merging surrogates, and performing other spatial analyses. For additional background on and specifics of using the MIMS Spatial Allocator, see http://www.cep.unc.edu/empd/projects/mims/spatial.

An example of a point-data-based surrogate file is shown in Figure 1. The first line gives information on the grid for which the surrogate was generated. The values on the remaining lines are the surrogate ID, the five-digit FIPS state-county code, the column and row of the grid cell, and the value of the surrogate. The values that following the "!" are comments that show the numerator used to compute the surrogate, the denominator used to compute the surrogate, and the running total of all surrogate values for that county.

Figure 1. Example of a surrogate file based on point data.

```
#GRIDM_08_99NASH 1000000.00 -536000.00 8000.00 8000.00 46 42 1 LAMBERT     meters 30.00
60.00 -100.00 -100.00 40.00
   4  47087    32   32       1      !        6        6     1
   4  47079     3   27       1      !       12       12     1
   4  47085     7   24     0.8      !       24       30   0.8
   4  47085     9   25     0.2      !        6       30     1
   4  47065    38   16  0.263158    !       30      114   0.26316
   4  47065    39   15  0.526316    !       60      114   0.78947
   4  47065    39   16  0.105263    !       12      114   0.89474
   4  47065    40   17 0.0526316    !        6      114   0.94737
   4  47065    40   18 0.0526316    !        6      114     1
   4  47071     6   10       1      !        6        6     1
```

BODY

Background

The December 2003 version of the Spatial Allocator was able to generate basic point-, line-, and polygon-based surrogates for modeling grids from data contained in Shapefiles. It could also create Shapefiles with a different map project from existing Shapefiles, and perform some more general spatial allocation features such as aggregating county-level data to state-level data. However, surrogates have become more complex in recent years. For the tool to become more widely used, some new functionality was needed.

EPA has funded a project to upgrade the Spatial Allocator. ICF Consulting is the prime contractor for this project, with the duration of October 2004 through June 2005. UNC CEP, a subcontractor, is responsible for the software design and modifications. The purpose of the project is to improve the tool's ability to reproduce the more than 60 surrogates that are currently being used for air quality modeling applications at EPA (see http://www.epa.gov/ttn/chief/emch/spatial/newsurrogate.html for more information). Other goals of the project are to make the tool more flexible so that it can be used for more general spatial allocation functions and to create inputs needed by the biogenic emissions model SMOKE-BEIS3. The work for the project consists of 12 tasks that are organized into two phases. In Phase 1, which ended in January 2005**,** new features for creating more sophisticated surrogates were implemented. In Phase 2, new features for performing more general spatial allocation functions are being implemented. In addition, the software will process EPA's most recent Biogenic Emissions Land use Data (BELD3) to create inputs for SMOKE-BEIS3. At the completion of each phase the software is to be compiled into a release and provided to users.

To support the more sophisticated surrogates in use today, several new features were added to the Spatial Allocator to create the "Phase 1" version. First, where previously the Spatial Allocator created surrogates from all of the shapes in the input Shapefile, now the user can specify a subset of the shapes to include in the surrogate based on their attribute values. The same subset specification can also be used to create a Shapefile that is a subset of the original Shapefile. For example, this new feature allows a surrogate or Shapefile for urban interstates to be generated from a Shapefile that contains many road types.

Another new feature is the generation of surrogates based on a function of multiple attributes in one or more Shapefiles. This is to support the creation of surrogates such as the "light and high tech industrial" surrogate, which is based on the sum of the light and high tech industrial square footage in each census tract; and the "half housing half population" surrogate, which is based on equal weightings of the housing and population surrogates. An additional new feature generates surrogates using "gap filling," This technique is needed when the weight file used to generate the surrogate does not contain entries for all counties that have emissions specified in the emission inventory.

The Phase 2 work on the software is being performed from March to June 2005. Before the Spatial Allocator is updated to process BELD3 data to create the inputs for biogenic processing that are required by SMOKE, the tool will be split into two programs and a library: one program will focus on creating surrogates, the other program will perform more general spatial allocation tasks, and the library will contain functions that are required by both programs. The general Spatial Allocator will support functions such as mapping gridded data to and from county-level data, mapping from grid to grid, and aggregating data from census tract to county levels. It will also be able to print the attributes of the points, lines, or polygons that are overlapped by a grid, bounding box, or set of polygons.

Current Release

Creating Filtered Shapefiles and Surrogates

An issue with the December 2003 version of the Spatial Allocator was that it created surrogates for all shapes in a Shapefile. This was a limitation for producing surrogates because Shapefiles can contain more shapes than should be included for the processing of a surrogate. This limitation was addressed in the January 2005 (Phase 1) Spatial Allocator by letting the user provide a "filter file" to specify criteria that cause only a subset of the shapes to be used when generating the surrogate. Thus, it is now possible to generate an interstate highways surrogate from a Shapefile with many road types, or a water surrogate from a Shapefile containing both land and water bodies.

A filter file is a text file that contains criteria for including or excluding shapes based on the values of their attributes. A filter file is used to generate either a filtered Shapefile or a surrogate that uses only a subset of values from a non-filtered Shapefile. The filter file consists of blocks of keyword-value pairs (i.e., lines of the form keyword=value). There are two required keywords in each filter file: ATTRIBUTE_NAME and ATTRIBUTE_TYPE. The value given for the ATTRIBUTE_NAME specifies the attribute (i.e., column) in the .dbf file of a Shapefile set whose values will be evaluated against the filter. The ATTRIBUTE_TYPE should be set to either DISCRETE or CONTINUOUS. When an attribute is treated as DISCRETE, the user specifies a list of values or a "regular expression" based on which shapes are included or excluded. When an attribute is treated as CONTINUOUS, the criteria for the attribute are specified in the form of mathematical relations or ranges of values (e.g., >= 1000, 100-200).

Once the attribute name and type are selected, the rest of the filter criteria can be specified by including or excluding values. The keyword INCLUDE_VALUES specifies a list of values for which the corresponding shapes in the Shapefile should be retained in the output Shapefile; the keyword EXCLUDE_VALUES specifies a list of values for which the corresponding shapes in the Shapefile should be *not* be included in the output Shapefile. The user must specify either INCLUDE_VALUES or EXCLUDE_VALUES for each ATTRIBUTE_NAME that is specified. If desired, both INCLUDE_VALUES and EXCLUDE_VALUES may be set, and their effect is cumulative. This means that any shapes that pass the INCLUDE_VALUES criteria must also pass the EXCLUDE_VALUES criteria for the shape to be placed in the output Shapefile. Only a single INCLUDE_VALUES and EXCLUDE_VALUES line should be specified for an ATTRIBUTE_NAME. Lines in filter files that start with "#" are treated as comments.

The example filter file shown in Figure 2 filters a U.S. roads Shapefile to create a Shapefile or surrogate that includes only the shapes with CFCC values of A10-A18 (highways).

Figure 2. Filter file to select highways using a discrete criteria.

```
# Filter File for Highways
ATTRIBUTE_NAME=CFCC
ATTRIBUTE_TYPE=DISCRETE
INCLUDE_VALUES=A10,A11,A12,A13,A14,A15,A16,A17,A18
```

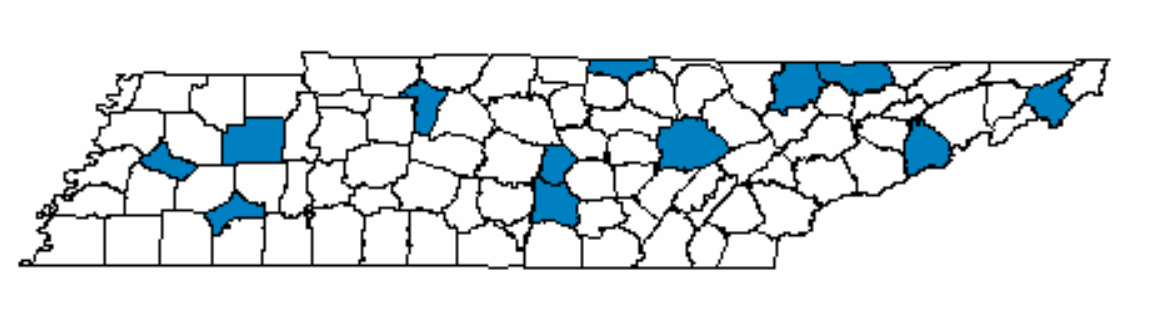
The filter file in Figure 3 illustrates the use of a regular expression, which is supported when the ATTRIBUTE_TYPE is specified as DISCRETE. The file in Figure 3 specifies that only those counties for which the value of the STATE_NAME attribute is "Tennessee" and for which the value of the [county] name attribute starts with the letter "C" will pass the filter. The results of applying this filter file to a US_Counties Shapefile are shown in Figure 4. The ability to use regular expressions as criteria opens up a realm of possibilities for filtering Shapefiles based on discrete attributes. Some other examples of regular expressions are "[A-K]*" for values starting with one of the letters from A through K, or "92??" for four-character values starting with "92".

Figure 3. Filter file for STATE_NAME=Tennessee and COUNTY_NAME="C*".

```
ATTRIBUTE_NAME=STATE_NAME
ATTRIBUTE_TYPE=DISCRETE
INCLUDE_VALUES=TENNESSEE

ATTRIBUTE_NAME=COUNTY_NAME
ATTRIBUTE_TYPE=DISCRETE
INCLUDE_VALUES=C*
```

Figure 4. Result of applying the Figure 3 filter file for state and county name.

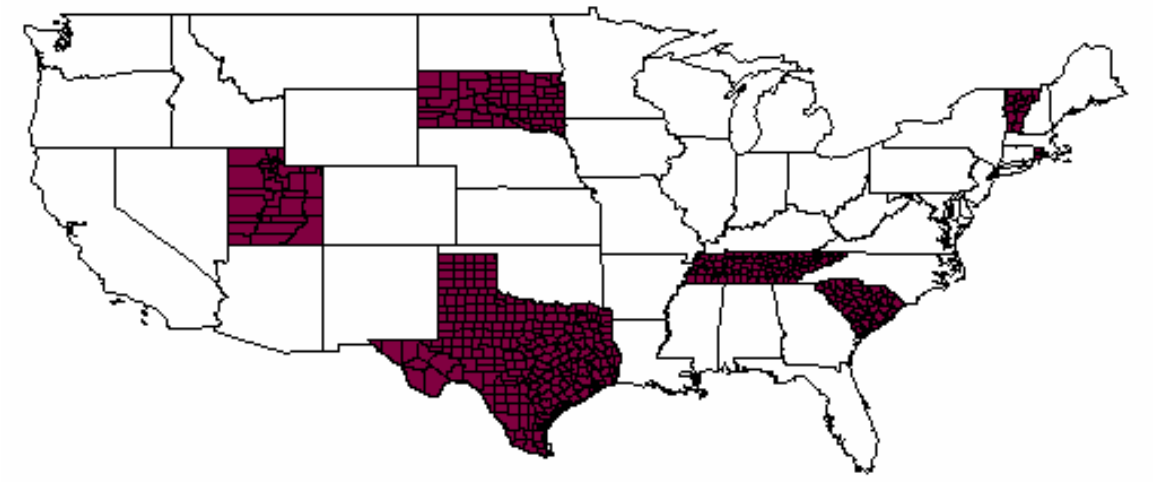


In Figure 5, the filter is based on the STATE_FIPS attribute, which is treated as CONTINUOUS. This filter causes shapes for which the value is between 44 and 50 to be included in the output Shapefile. Figure 6 shows the result of applying this filter to the US_Counties Shapefile.

Figure 5. An example of a continuous filter for the STATE_FIPS attribute.

```
# Continuous Filter Example 1
ATTRIBUTE_NAME=STATE_FIPS
ATTRIBUTE_TYPE=CONTINUOUS
INCLUDE_VALUES=44-50
```

Figure 6. Result of applying the Figure 4 continuous filter for STATE_FIPS=44-50.



In Figure 7, the filter is based on the attribute LENGTH (i.e., road length), which is treated as a CONTINUOUS attribute. This filter causes shapes for which the value of LENGTH is not between 100 and 200 (inclusive), or greater than 400, to be included in the output Shapefile. Note that continuous criteria may include values with decimal points (e.g., 10.5, 2.25).

Figure 7. A continuous filter illustrating EXCLUDE_VALUES and multiple criteria.

```
# Example C
ATTRIBUTE_NAME=LENGTH
ATTRIBUTE_TYPE=CONTINUOUS
EXCLUDE_VALUES=100-200,>400
```

Creating Surrogates Using Functions of Attribute Values

The second major enhancement in the January 2005 version of the Spatial Allocator was to allow surrogates to be based on a function of attributes. To use this feature, the user specifies an environment variable called WEIGHT_FUNCTION that contains a mathematical expression of attributes. The result of this function is the weight to be used for the shape (instead of the usual process, which is to use the value of a single attribute). An expression parser to parse the functions and a computational engine to compute the results of the functions were added to the Spatial Allocator to support this feature.

When all of the attribute values used in computing the surrogate weight have the same units and reside in a single Shapefile, the corresponding surrogates can be computed using the new weight function feature. For example, you could set the WEIGHT_FUNCTION variable in order to compute a surrogate for industrial space, which uses the sum of all types of industrial space for each census tract. Using attribute names, the formula for this function is IND1+IND2+IND3+IND4+IND5+IND6. Note that when the attributes to be used for the surrogate weight are in different Shapefiles, or they have different units (e.g., the weighted roadway miles and population surrogate 0.75*length+0.25*pop), the weight function feature cannot be used and the new srgmerge program (described below) should be used to compute the new surrogate based on other independently generated surrogates.

A weight function can be any arithmetic equation containing the operators +, -, *, /, (,), numeric constants, and names of attributes that exist in the weight Shapefile. Exponential notation and power functions are not currently supported, nor are unary negative numbers used as constants (e.g., X1 + -5 should be expressed as X1 - 5). Examples of acceptable weight functions are

```
WEIGHT_FUNCTION=(IND1+IND5)
WEIGHT_FUNCTION=0.75*urban+0.25*rural
```

Parentheses can be used to change the order of operation in the function. The attributes used in the weight function must be numeric (i.e., floating point or integer) data types. It is the responsibility of the user to use only attribute names that are in the weight Shapefile. The use of string attributes such as county name or road type is not allowed in a weight function. No limit to the length of the weight function is imposed by the system.

Creating Surrogates Using Functions of Other Surrogates

In some cases it is desirable to apply a weight function, but the attributes of interest do not reside in a single Shapefile, or their units are not consistent. Surrogates can be computed in these cases, but the input surrogates must first be computed independently, and then combined using weighted averages. To implement this feature, a separate program called srgmerge has been developed. Srgmerge takes as input a single text file that specifies functions for combining surrogates (typically the functions used will be weighted averages). Srgmerge reads previously computed surrogates from files and outputs new surrogates. It can both merge surrogates and perform gap filling, which is described in the next section. The computational engine used for srgmerge is the same as the one used for processing the weight function.

The ASCII input file to srgmerge uses keyword-value pairs in a manner similar to the filter file described earlier. The following three keywords *must* be found in the srgmerge input file: OUTFILE, XREFFILE, and OUTSRG. The function of these keywords is as follows:

- OUTFILE specifies an output file name to which the new surrogates will be written.
- XREFFILE specifies a file name for a surrogate cross-reference file.
- OUTSRG specifies the name of the output surrogates and a function that describes how the input surrogates are to be combined.

Note that multiple OUTSRG= lines may exist in the input file, and the specifications that follow OUTSRG= must be given on a single line. When multiple OUTSRG lines are found in a file, there will be multiple sets of surrogates in the resulting output file. The syntax for the functions specified in the OUTSRG lines is similar to that of the weight functions, with two exceptions: the names of the files that contain the input surrogates are included as part of the syntax (as shown in Figure 8), and division and subtraction are not allowed. So, the inputs to these functions are the file and input surrogate names, as opposed to the weight functions, which use attribute names. The name of the new surrogate to be generated is also included on the OUTSRG line.

Figure 8. An example srgmerge input file to compute a weighted average of surrogates.

```
# Input file for srgmerge
OUTFILE=output/merged_srgs.txt
XREFFILE=data/srg_xref.txt
OUTSRG=Half Pop Half Housing; 0.5*({data/surrgs.txt|Population})
+0.5*({data/surrgs.txt|Housing})
OUTSRG=Population; 1*{data/surrgs.txt|Population}
OUTSRG=Housing; 1*{data/surrgs.txt|Housing}
```

The example in Figure 8 depicts a typical input file for srgmerge. Any lines that begin with the character "#" are considered to be comments and are ignored by the program. All of the output surrogates will be saved in the new file specified in the OUTFILE line. In this case it will be named output/merged_srgs.txt. The XREFFILE line specifies the name of the cross-reference file for surrogate names and IDs (more detail is given on this file below). The first OUTSRG= line in this example specifies that the name of the first set of new surrogates to compute is "Half Pop Half Housing". The new surrogate fractions will be computed using an evenly weighted average of the Population surrogates in the file data/surrgs.txt and the Housing surrogates in the same input file. The remaining OUTSRG lines specify that the Population and Housing surrogates should be copied to the output file without making any changes. Note that the names of the input surrogates are enclosed in curly braces (i.e., {}) using the syntax {*file name*|*surrogate name*}; a pipe ("|") is used to separate the file name from the surrogate name. We would have preferred to use a colon, but there were issues because a colon separates the disk drive from the file name on the Windows platform.

The allowable operators and functions in the srgmerge input file have the same limitations as described above for the weight functions. One exception is that division and subtraction are not allowed in srgmerge because applying these operators to surrogates could cause spurious results (e.g. divide by zero and negative fractions). The most commonly used function will be to specify a weighted average for which the "weights" that precede the surrogate names sum to 1. This will ensure that the output surrogate will sum to 1 for each county. Note that you should use either forward or back slashes for the file names as is appropriate to the computer you are running on (i.e., back slashes for Windows, or forward slashes for Linux or other Unix systems). The srgmerge program does some checking on the surrogate fractions that it outputs. It checks to make sure each surrogate fraction it outputs is between 0 and 1. Also, if it finds that the total of all the fractions for a county is not 1, it adds a comment stating this to the end of the surrogate line.

The XREFFILE keyword specifies the name of the surrogate cross-reference file that contains surrogate names and IDs for both the input and output surrogates. This reference file links surrogate ID codes and surrogate names. All surrogates in the OUTSRG list either must appear as comments (e.g., #SRGDESC=number, name) in the input surrogate files, or must exist within the XREFFILE. So, for the above example to work properly, the corresponding cross-reference file should contain entries for at least "Half Pop Half Housing." The descriptions for Population and Housing may be provided as comments in the input surrogate files; if they are not included there, they should be listed in the XREFFILE. Each entry in the XREFFILE or in the header of the surrogate files should have the format #SRGDESC=*id,name*. For the input surrogates, the headers of the surrogate files will be checked first. If the name is not found there, a warning will be printed and then the XREFFILE will be checked. If the name is not found in either place, srgmerge will quit with an error. Figure 9 shows an example cross reference file with surrogate IDs for airports, railroads, agriculture, population, housing, gap-filled airports, gap-filled airports, gap-filled railroads, and half pop half housing.

Figure 9. An example surrogate cross-reference file.

```
# Example of a surrogate cross reference file
#SRGDESC=2,Airports
#SRGDESC=5,Railroads
#SRGDESC=6,Agriculture
#SRGDESC=7,Population
#SRGDESC=8,Housing
#SRGDESC=201,Gapfilled Airports
#SRGDESC=204,Gapfilled Railroads
#SRGDESC=300,Half Pop Half Housing
```

Generating Surrogates Using Gap Filling

The next major feature added to the January 2005 Spatial Allocator addresses the case where the weight data sets do not contain values in all counties for which there are emissions in the emission inventory. If left unchecked, this can cause the emissions in those counties to be lost. The idea behind the gap-filling process is to allow the user to specify secondary, tertiary, and quaternary surrogates that will be used to fill in the data gaps with data that are less specific, yet still relevant. Thus, if a primary surrogate does not have a value for a given county, the value will be obtained from a secondary surrogate. If the secondary surrogate does not have a value, the value will be obtained from a tertiary surrogate, and so forth; hence the term "gap filling." Gap filling is a secondary function performed by the srgmerge program. Note that although srgmerge can do both merging and gap filling, they cannot be done during the same execution of the program—even for different surrogates. That ability is recommended as a future enhancement of the srgmerge program.

When gap filling, it is assumed that the most detailed (i.e., lowest-level) surrogate has entries for all counties of interest; the counties given for the output surrogate will therefore be the same as those available for the lowest-level surrogate. Each surrogate that is input to the gapfill function of srgmerge must be created before srgmerge is executed. To access the gapfill function of the srgmerge program, add the word "GAPFILL=" after the output surrogate name in the value for the OUTSRG keyword, then list the surrogates to use with their file names in the order of precedence from primary to quaternary. Otherwise, the input file format is the same as was discussed previously. Execution of srgmerge for gap filling works the same way it does for merging surrogates simply type the name of the program followed by the ASCII input file.

An example srgmerge input file that performs gap filling is shown in Figure 10. Note that the input file is very similar to the input file for the surrogate merging mode of srgmerge shown in Figure 8, only the OUTSRG line has changed. The first OUTSRG line in this file causes a new surrogate called Filled Airports to be created and placed in the output file. The values for this surrogate will come first from the Airports surrogate; for any counties missing from the Airports surrogate, the values will be

obtained from the Population surrogate. The second and third OUTSRG lines cause the original Airports and Population surrogates to be copied to the output file without any modification. The XREFFILE and OUTFILE keywords operate exactly as they do when merging surrogates. Note that there must be entries in the XREFFILE for the surrogates Filled Airports, Airports, and Population for the example shown in Figure 10 to work.

Figure 10. An example srgmerge input file to perform gap filling.

```
#Example Input file for gap filling using srgmerge
OUTFILE=output/new_srg.txt
XREFFILE=srg_xref.txt
OUTSRG=Filled Airports;GAPFILL=data/surrgs.txt|Airports;data/surrgs.txt|Population
OUTSRG=Airports;GAPFILL=data/surrgs.txt|Airports
OUTSRG=Population;GAPFILL=data/surrgs.txt|Population
```

Phase 1 Release Information

The January 2005 version of the Spatial Allocator includes the Phase 1 enhancements described above. It is available from the CEP web site at http://www.cep.unc.edu/empd/projects/mims/spatial/. The Spatial Allocator software is open source and has been released according to the terms of the GNU general public license. The release contains documentation, sample data sets, scripts to exercise the software, test scripts and data, and three executable programs:

- mims_spatial: the actual Spatial Allocator that generates surrogate files and performs other operations on Shapefiles;
- srgmerge: a program that reads existing surrogate files and performs gap-filling and weighted averaging of the surrogate fractions; and
- diffsurr: a program that compares two surrogate files and identifies any differences between them.

Precompiled executables are provided for Windows, Linux, and AIX. Makefiles are available for generating executables for other operating systems. The documentation includes an installation guide and information on how to use the features of the software. The software went through several levels of testing prior to its release. Testing was first performed by CEP staff and then by ICF staff. Automated unit tests were developed to test the details of the software code. Integration tests were performed to make sure that the software performed adequately with “real-world”-size input data sets. The outputs from the Spatial Allocator were compared to those produced by a commercial GIS. Values for functions that were computed by the Spatial Allocator were compared to those computed using a spreadsheet. Rational Purify Plus™ was applied to identify memory leaks in the software.

Next Release

Generic Spatial Allocation

As part of the Phase 2 enhancements, the Spatial Allocator will be refactored into three parts: (1) a library with core geospatial computation functions, (2) a program for creating surrogates, and (3) a generic Spatial Allocator program that handles the rest of the spatial allocation functions. The surrogate application will have the same user interface (with minor usability and performance-related adjustments) and produce the same results as the January 2005 version of the Spatial Allocator. The two new programs that will be created during Phase 2 will be called **srgcreate** for creating surrogates and **allocator** for the rest of the spatial allocation functions. The library that will contain the remaining functions will be called **libspatial**. Since the variables used by srgcreate will be comparable to the ones used by the January 2005 Spatial Allocator but with different names, a script will be provided to help existing users upgrade their scripts to use the new syntax.

The allocator program will accept an input data set associated with a set of points, polygons, or a grid, and allocate the data to a different set of polygons. Some examples of this are mapping county data to a grid, mapping gridded data to a county, gridding point observation data, regridding data from one grid to another, and allocating data up from county level to state level. In general, the spatial operations that will be supported by the allocator program are the following types of mapping:

1) a set of points to a set of polygons,
2) a set of polygons to a different set of polygons (no restrictions will be made regarding whether the input polygons are generally larger than the output polygons or vice versa),
3) a set of points to a regular grid,
4) a set of polygons to a regular grid,
5) a regular grid to a set of polygons, and
6) a regular grid to a different regular grid (e.g., for changes in resolution, map projection, or orientation).

Both srgcreate and the allocator program will be optimized to support efficient operation with, and easy specification of, regular grids. For example, the user will need to specify only the name of an input or output grid, and its details will be looked up in a GRIDDESC file. Both applications will be able to process large (e.g., 1- to 2-GB) data sets efficiently. For the srgcreate program, this memory optimization will be accomplished by limiting the number of data polygons (i.e., counties) that are processed at any given time; this may require that the input weight file be read multiple times. A partitioning algorithm will be implemented in both applications such that the entire spatial extent of the output domain (be it a grid or other set of polygons) does not need to be kept in memory at one time.

The allocator program has some different processing modes that the January 2005 Spatial Allocator. First, note that MIMS_PROCESSING environment variable has been changed to PROCESSING_MODE. The options for PROCESSING_MODE in the allocator program will be ALLOCATE, OVERLAY, CONVERT_SHAPE, and FILTER_SHAPE. Note that the old AGGREGATE and AVERAGE modes have been replaced by the new ALLOCATE mode, and the CONVERT_BELD mode will be removed. The new method for processing BELD3 data to create biogenic inputs for SMOKE is described below. The CONVERT_SHAPE and FILTER_SHAPE modes that exist in the January 2005 Spatial Allocator remain in the allocator program and use syntax similar to their current syntax except for some environment variables that are renamed. The new OVERLAY mode is described in more detail below.

The environment variables used for the new ALLOCATE mode are shown in Figure 11. Additions to the previous syntax are shown in **bold**. Variable names that are followed by an asterisk (*) existed conceptually in a prior version, but are being renamed in the new syntax. If the substituted names are new, the old name is shown in parentheses after the new one. Some examples of how the new allocator program can be used follow Figure 11.

Figure 11. Syntax for ALLOCATE mode of the allocator program.

PROCESSING_MODE***** – **ALLOCATE (replaces AVERAGE and AGGREGATE)**

GRIDDESC – the name of the GRIDDESC file that describes all grids

INPUT_FILE_NAME – the name of the file containing input data for spatial allocation

INPUT_FILE_TYPE – Shapefile (point or polygon), **PointFile, IoapiFile,** or **RegularGrid** (i.e., a special type of Shapefile that contains gridded data)

INPUT_FILE_MAP_PRJN – the map projection of the INPUT_FILE_NAME file

INPUT_FILE_ELLIPSOID – the ellipsoid of the INPUT_FILE_NAME file

INPUT_FILE_DELIM – the delimiter that is used for the PointFile (when applicable)

INPUT_FILE_XCOL – the name of the column containing x coordinates in the PointFile (when applicable)

INPUT_FILE_YCOL – the name of the column containing y coordinates in the PointFile (when applicable)

INPUT_GRID_NAME – the name of the input grid (if applicable)

ATTR_INFO_FILE – the name of a file that lists the attributes in the INPUT_FILE_NAME file to carry through to the output file and the algorithm by which to allocator the attributes (i.e., SUM, AVERAGE, DISCRETE_OVERLAP, or DISCRETE_CENTROID), or one of the constants ALL_SUM, ALL_AVERAGE, ALL_DISCRETE_OVERLAP, or ALL_DISCRETE_CENTROID,

OUTPUT_FILE_NAME* – the directory and name of the output file

OUTPUT_FILE_TYPE***** – RegularGrid, **Shapefile** (polygons only), or **IoapiFile**

OUTPUT_GRID_NAME*** –** the name of the output grid (if applicable)

OUTPUT_FILE_ELLIPSOID***** – ellipsoid of the output polygons

OUTPUT_FILE_MAP_PRJN***** – the map projection of the output file (if output is not gridded)

OUTPUT_POLY_FILE***** (was POLY_DATA) **–** the name of a Shapefile that specifies the geometry of the *output polygons* when the output is a Shapefile

OUTPUT_POLY_ELLIPSOID***** (was POLY_DATA_ELLIPSOID) – the ellipsoid of the output shapes

OUTPUT_POLY_MAP_PRJN* (was POLY_DATA_MAP_PRJN) – the map projection of the output shapes (note that these *must* be polygons - points or lines are not allowed)

X_GRID_PARTITIONS – the number of partitions for processing the output grid in the X direction (used when the OUTPUT_FILE_TYPE is RegularGrid or IoapiFile); see the detailed description below for more information

Y_GRID_PARTITIONS - the number of partitions for processing the output grid in the Y direction (used when the OUTPUT_FILE_TYPE is RegularGrid or IoapiFile)

MAX_OUTPUT_POLYS – the maximum number of output polygons to keep in memory for processing at one time (used when the OUTPUT_FILE_TYPE is Shapefile)

* The variable existed conceptually in the January 2005 version of the software.

In the following point-data-to-Shapefile example, the attributes of the point sources in pointSources2000.csv will be totaled by county and written to a Shapefile. The file pointSources2000.csv contains the columns Long, Lat, NOX, VOC, PM10, and SO2. The output Shapefile will contain only the attributes specified in emis_attrs.txt. Note that the values for some of the variables are special constants recognized by the program (e.g. ALLOCATE, COMMA, SPHERE).

```
set PROCESSING_MODE = ALLOCATE
set GRIDDESC = GRIDDESC.txt
set INPUT_FILE_NAME = pointSources2000.csv
set INPUT_FILE_TYPE = PointFile
set INPUT_FILE_DELIM = COMMA
set INPUT_FILE_XCOL = Long
set INPUT_FILE_YCOL = Lat
set INPUT_FILE_MAP_PRJN = +proj=lcc,+lat_1=33,+lat_2=45,+lat_0=40,+lon_0=-97
set INPUT_FILE_ELLIPSOID = WGS_1984
set ATTR_INFO_FILE = emis_attrs.txt
set OUTPUT_FILE_TYPE = Shapefile
set OUTPUT_FILE_NAME = ..\output\northeast_emis_by_county
set OUTPUT_POLY_FILE = northeast_counties
set OUTPUT_POLY_ELLIPSOID = SPHERE
set OUTPUT_POLY_MAP_PRJN = LATLON
set MAX_OUTPUT_POLYS = 200
```

The file emis_attrs.txt contains the following lines to specify that each of the three attributes should be summed over each output polygon (note that # denotes a comment):

```
# Attributes to include from the emissions data
NOX: SUM
SO2: SUM
PM10: SUM
```

The following are other examples of operations that can be done with the new allocator program:

- Allocate polygons to polygons: the population data in us_censusTracts2000 can be allocated to states and the data output into a Shapefile.
- Allocate points to a grid: the observation data in july2001obs.txt can be allocated into grid cells and the data output into a RegularGrid-style Shapefile.
- Allocate polygons to a grid: the polygon-based land use data listed in us_landuse can be allocated onto a grid and the data output into an I/O API file.
- Allocate gridded data to polygons: the gridded land use data attributes listed in us_1km_landuse can be allocated to counties and the data output to a Shapefile.
- Allocate gridded data to another grid: the gridded air quality model data species listed in cmaq_attrs.txt from cctm_07112001_12km can be allocated to another grid and the data output to a regular grid Shapefile (alternatively we could have chosen IoapiFile as the output type).

Allocating Discrete Attributes

The allocator program must be able to perform allocation for both discrete and continuous attributes in the output data sets. Continuous attributes (e.g., length, square footage, or population) can be apportioned according to the area of overlap and transferred to the output data set accordingly. However, discrete attributes (e.g., urban vs. rural, or land use ID) can have only one value that corresponds to each shape in the output data set. Two options will be provided in the allocator program for determining the value of a discrete attribute for each output shape. One will be based on maximum area of overlap (i.e., the attribute from the input shape that has the largest overlap with the output shape will be used); the other will be based on centroid (i.e., the attribute from the input shape that lies at the centroid of the output shape will be used). Initially, allocation of discrete values will be supported only for polygon-to-polygon mapping, not point-to-polygon. Note that the centroid method does not make sense when applied to point or line data. For point data, the maximum overlap method would be similar to applying a counting algorithm (i.e., finding the most popular value in each output polygon), and for lines it would be similar to computing the total length that applied for each value of the attribute and picking the value that corresponds to the largest length.

The method for choosing the appropriate discrete value for a shape will be specifying an appropriate value in the attribute information file (ATTR_INFO_FILE). For example: If you want to know the predominant county for a grid cell, you could specify the DISCRETE_OVERLAP method in the attribute information file to allocate the FIPS code; alternatively, you could use the DISCRETE_CENTROID method to assign the FIPS code for the county that covers the centroid of the grid cell. Since the attribute information file allows values can be specified for each attribute, different choices could be made for each attribute if needed. Otherwise, the ALL_DISCRETE_OVERLAP or ALL_DISCRETE_CENTROID constant can be used. Note: To implement this feature, a centroid computation algorithm will need to be added to this software.

Printing Shapes That Are Overlaid by a Region

The allocator program will be able to print the shapes overlapped by a bounding box, grid, polygon, or shapes in a Shapefile. The new OVERLAY mode will support this feature. The mechanism for determining whether shapes overlay a region already exists in the software. Some additional output types will be added to support this feature: DelimitedFile and Stdout for "standard output" (i.e., to the screen). The following is an example of the syntax that would be used to print all of the attributes for observation stations in stations.txt that overlap the EAST_US_36 grid:

```
set PROCESSING_MODE = OVERLAY
set OVERLAY_TYPE=RegularGrid
set OVERLAY_SHAPE=EAST_US_36
set OVERLAY_ATTRS=ALL
set INPUT_FILE_NAME = stations.txt
set INPUT_FILE_TYPE = PointFile
set INPUT_FILE_MAP_PRJN = +proj=utm,+zone=17
set INPUT_FILE_ELLIPSOID = +ellps=MERIT
set INPUT_FILE_DELIM = COMMA
set INPUT_FILE_XCOL = Longitude
set INPUT_FILE_YCOL = Latitude
set OVERLAY_OUT_TYPE = DelimitedFile
set OVERLAY_OUT_NAME = east_us_36km_obs.txt
set OVERLAY_OUT_DELIM = SEMICOLON
set WRITE_HEADER = Y
set WRITE_DEBUG_OUTPUT = Y
```

Other examples of using the OVERLAP mode are the following:

- Print the abbreviations of the states that overlap the specified bounding box. If WRITE_DEBUG_OUTPUT is off, the only output will be the list of state abbreviations and could thus be used as part of a script to drive other processing.
- Print the names of the counties and states in the input Shapefile that are inside a polygon specified in a polygon file that represents a national park.
- Print the row and column numbers of the grid cells that overlap the polygons in a Shapefile (e.g., a state).
- Print modeled ozone values extracted from an I/O API file for the grid cells that cover a national park (as specified by a polygon file).

Creation of SMOKE-BEIS3 Inputs

Inputs for SMOKE-BEIS3 will be created using the BELD3 data. These data are available at ftp://ftp.epa.gov/amd/asmd/beld3 as ASCII or ArcInfo datasets. BELD3 covers most of North America at a 1-km resolution. Due to its size, the data is split into 24 adjacent "tiles". In BELD3, each 1-km pixel is associated with a country code and a county via a FIPS code. The percentages of forest, agriculture, water, and other are also listed for each pixel. These should add up to 100% for each pixel. For each county, there are data regarding the percentages of each land use type in that county; one file contains agricultural types and the other contains forest types. The two data sets are brought together to compute the land use fractions for each of the 230 land use types for the pixel. The allocated land use types should add to 100% for each pixel.

SMOKE-BEIS3 requires three I/O API files that contain land use information to do its biogenic processing. These files are known as A, B, and TOT and must contain data for the grid on which the air quality model will be executed. To create these files, the software will need to determine which of the 24 BELD3 tiles intersect the air quality model grid. The data from the portions of the tiles that overlap the

grid will need to be converted to the map projection of the air quality model grid and aggregated up to the resolution of the air quality model grid.

A new executable program called **beld3smk** will be developed to create the SMOKE-BEIS3 inputs. This executable will require four input data sources that will be created one time and distributed with the program:

1) compressed A, B, and TOT files for each BELD3 tile in the I/O API format,
2) a GRIDDESC file with descriptions of the grids for all of the BELD3 tiles,
3) empty A and B files with headers that contain all 230 BELD3 land use types from which variable names and descriptions will be retrieved, and
4) a Shapefile with shapes that correspond to the 24 BELD3 tiles.

The compressed A, B, and TOT files will be created using existing programs. These files will be "compressed" in that they will contain data only for the subset of the 230 possible land use types that have nonzero values within the corresponding BELD tile. If the land use types in the tile can fit into a single file instead of needing A and B files, only a single file will be available. Once these data files are available, the new program will be implemented as follows:

1) The program will use the OVERLAY mode of the allocator program with the Shapefile of the BELD3 tiles to determine which tiles intersect the modeling grid.
2) For each tile that intersects the modeling grid, beld3smk will use the grid-to-grid allocation function available from in the allocator program to create a temporary I/O API file aggregated to the level of the modeling grid for that tile. Note that the map projection of the BELD3 data may not match the map projection of the modeling grid. However, the grid-to-grid conversion process will automatically change the map projection of the input data as the data are read in, so any map projection changes will be accounted for during the conversion.
3) Beld3smk will read the temporary files that were created for each tile, sum them for each grid cell in the modeling grid, and output the final A, B, and TOT I/O API files that will be read in by SMOKE-BEIS3. The purpose of this summing is to take into account the case where a modeling grid cell is split between two tiles. If the sum of the percentages for a grid cell is not within a reasonable tolerance of 100%, then a warning will be issued.
4) The user will have the option of stating whether to include any land use types that do not exist in the modeling grid in the A and B files. SMOKE-BEIS3 will be updated so that it can accept an incomplete set of land use types in that it will assume that any land use types that are not specified do not occur in the modeling grid.

The beld3smk program will respond to the following environment variables:

- OUTPUT_GRID_NAME – the name of the modeling grid for which the output A, B, and TOT files should be created
- GRIDDESC – the name of the GRIDDESC file that contains the info for the output grid
- INPUT_DATA_DIR – the name of the directory that contains the four input data sources needed by the program
- WRITE_ALL_TYPES – specify Y to output all 230 land use types; otherwise, only the land use types that actually exist in the modeling grid will be output (all land use types will be required for older versions of SMOKE-BEIS3)
- OUTPUT_FILE_PREFIX – a prefix for the three output files that will be created (_A.ncf, _B.ncf, and _TOT.ncf will be added to this)

- OUTPUT_FILE_ELLIPSOID – the ellipsoid of the output files

Specifying a Discretization Interval

The allocator program will provide an option for specifying a maximum line segment length. Any line segments in lines, polygons, and generated grid cells will be split into segments that are not longer than the specified length. This will improve accuracy when converting shapes between map projections. To accomplish this task, a line-splitting algorithm will be incorporated into the software. The length of the discretization interval will be specified using an environmental variable, such as `set MAX_LINE_SEG_METERS=1000`. If the output projection is LATLON, the user should specify MAX_LINE_SEG_DEGREES instead of MAX_LINE_SEG_METERS. Automated tests for this feature will be developed for lines, polygons, and grids. The results will be compared against results from a commercial GIS.

Possible Additional Enhancements

The following are potential future enhancements for the Spatial Allocator that have been identified, but are not explicitly included in the scope of work for the current project:

- Create and make available a system based on the Spatial Allocator to reproduce the over 60 surrogates currently in use by the EPA. Verify that the surrogates are properly reproduced.
- Support gap filling and merging in the same execution of srgmerge.
- Allow the value for OUTSRG= lines in the srgmerge input file to span multiple lines.
- Accept the current srgdesc format used by SMOKE as an alternative format for the surrogate cross-reference file. Other tweaks to cross-referencing may also be required as the modeling community starts making use of this feature.
- When computing a function of multiple surrogates using srgmerge, if there are no surrogate fractions for a particular county for one of the input surrogates, gap-fill the values for that county using a specified gap-filling surrogate.
- Enhance the comments in the output surrogate file to include the names of the input data and weight files, the value of the WEIGHT_FUNCTION, and the contents of the filter file.
- Optionally add an extra surrogate line for counties on the edge of the grid for grid cell “0, 0” to represent the amount of the surrogate that fell outside the grid. This would cause surrogates for counties on the edge of the grid to sum to 1.
- Support line-to-polygon and line-to-grid operations in the allocator program.
- Support an ASCII-delimited line format similar to the PointFile format.
- Create netCDF files that follow standard conventions so they can be visualized by NCAR’s Integrated Data Viewer.
- Support allocation of discrete attributes when allocating point data to polygons/grids.
- Support the DISCRETE_COUNT method for polygons as an alternative to OVERLAP and CENTROID.
- For OVERLAP mode, support Shapefile as an output type and add an option to OVERLAP_IN_REVERSE.
- Support reading data from the new ArcGIS database format.

CONCLUSIONS

Substantial enhancements to the surrogate creation functionality of the MIMS Spatial Allocator have been incorporated into the January 2005 version of the software. It should now be able to reproduce most of the surrogates currently in use at EPA, although a system to reproduce the surrogates has not yet been developed. The additional enhancements that will be made available in a June 2005 version of the Spatial Allocator will further reduce the need for a GIS for air quality modeling. In addition, some unique features will be included, such as grid-to-county mapping and printing of data values from grid cells that overlap a polygon. These new features should find many uses in the air quality modeling community.

BIBLIOGRAPHY

Eyth, A. “The MIMS Spatial Allocator: A Tool for Generating Emission Surrogates without a GIS”. In *Proceedings, Emission Inventories: Applying New Technologies*; U.S. Environmental Protection Agency: San Diego, CA, 2003.

Vukovich, J. “Implementation of BEIS3 within the SMOKE Modeling Framework”. In *Proceedings, Emission Inventories: Partnering for the Future*; U.S. Environmental Protection Agency: Atlanta, GA, 2002.

ACKNOWLEDGMENTS

This work was funded under US EPA contract number GS-35F-4121D. Software development support was provided by Benjamin Brunk and Rajasooriya Partheepan of the CEP. Testing and documentation support was provided by Kimberly Hanisak of the CEP, and Mark Lee and Rebecca Murphy of ICF Consulting.

DISCLAIMER

The research presented here was performed under the Memorandum of Understanding between the U.S. Environmental Protection Agency (EPA) and the U.S. Department of Commerce's National Oceanic and Atmospheric Administration (NOAA) and under agreement number DW13921548. This work constitutes a contribution to the NOAA Air Quality Program. Although it has been reviewed by EPA and NOAA and approved for publication, it does not necessarily reflect their policies or views.

KEYWORDS

Spatial Surrogates
Geographic Information System (GIS) Alternative
SMOKE
BEIS3
BELD3