Composite Grid Generation Code for General 3-D Regions—the EAGLE Code

Joe F. Thompson*
Mississippi State University, Mississippi State, Mississippi

Abstract

A GENERAL three-dimensional grid generation code based on a composite block structure is discussed. The code can operate either as an algebraic generation system or as an elliptic generation system. Provision is made for orthogonality at boundaries and complete continuity at block interfaces. The code can operate in two or three dimensions or on a curved surface. The input is structured to be user-oriented, and arbitrary block configurations can be treated.

Contents

The construction of computational fluid dynamics (CFD) codes for complicated regions is greatly simplified by a composite block grid structure since, with the use of a surrounding layer of points on each block, a flow code is only required basically to operate on a rectangular computational region. The necessary correspondence of points on the surrounding layers (image points) with interior points (object points) is set up by the grid code and made available to the CFD solution code.

The EAGLE grid code is a general three-dimensional elliptic grid generation code based on the block structure. This code allows any number of blocks to be used to fill an arbitrary three-dimensional region. Any block can be linked to any other block (or to itself) with complete (or lesser) continuity across the block interfaces as specified by input. In the case of complete continuity, the interface is a branch cut, and the code establishes a correspondence across the interface using the surrounding layer of points outside the blocks. This allows points on the interface to be treated just as all other points so that there is no loss of continuity. The physical location of the interface is thus totally unspecified in this case, being determined by the code.

This code uses an elliptic generation system with automatic evaluation of control functions either directly from the initial algebraic grid and then smoothed, or by interpolation from the boundary-point distributions. In the former case the smoothing is done only in the two directions other than that of the control function. This allows the relative spacing of the algebraic grid to be retained but on a smoother grid from the elliptic system. In the latter case, the arc length and curvature contributions to the control functions are evaluated and interpolated separately into the field from the appropriate boundaries. The control function at each point in the field is then formed by combining the interpolated elements. This procedure allows very general regions with widely varying boundary curvature to be treated.

The control functions can also be determined automatically to provide orthogonality at boundaries with specified normal spacing. Here the iterative adjustments in the control functions are made by increments radiated from boundary points where orthogonality has not yet been attained. This allows the basic control function structure evaluated from the algebraic grid, or from the boundary-point distributions, to be retained and thus relieves the iterative process from the need to establish this basic form of the control functions.

Alternatively, boundary orthogonality can be achieved through Neumann boundary conditions which allow the boundary points to move over a surface spline, the boundary point locations being located by Newton iteration on the spline to be at the foot of normals to the adjacent field points. Provision is also made for mirror-image reflective boundary conditions on symmetry planes.

Although written for 3-D, the code can operate in a 2-D mode on either a plane or curved surface. In the case of a curved surface, the surface is splined and the generation is done in terms of surface parametric coordinates.

The code includes an algebraic three-dimensional generation system based on transfinite interpolation (using either Lagrange or Hermite interpolation) for the generation of an initial solution to start the iterative solution of the elliptic generation system. This feature also allows the code to be run as an algebraic generation system if desired. The interpolation, though defaulted to complete transfinite interpolation from all boundaries, can be restricted by input to any combination of directions or lesser degrees of interpolation, and the form (Lagrange, Hermite, or incomplete Hermite) can be different in different directions or in different blocks. The blending functions can be linear or, more appropriately, based on interpolated arc length from the boundary point distributions.

Blocks can be divided into sub-blocks for the purpose of generation of the algebraic grid and the control functions. Here, point distributions on the sides of the subblocks can either be specified or generated by transfinite interpolation from the edges of the side. This allows additional control over the grid in general configurations and is particularly useful in cases where point distributions need to be specified in the interior of a block, or to prevent grid overlap highly curved regions.

The composite structure is such that completely general configurations can be treated, the arrangement of the blocks being specified by input, without modification of the code. The input is user-oriented and designed for brevity and easy recognition. For example, the establishment of correspondence, i.e., a branch cut, between two blocks requires only the simple input statement

```
$INPUT ITEM = "CUT", START = ___,___,___,
END = ___,___,___, BLOCK = ___,
ISTART = ___,___,___, IEND = ___,___,___, IBLOCK = ___$
```

where START and END give the three indices of two opposite corners of the cut section on one block (BLOCK), while ISTART and IEND give the corners of the corresponding section on the other block (IBLOCK). The code sets up the point correspondence on the surrounding layers for complete continuity without additional input instructions.

Detailed discussion of both the use and the operation of the code is given in Ref. 1. The code is written in modular form so that components can be readily replaced. The code is vectorized (CRAY-XMP) wherever practical and includes pro-

Presented as Paper 87-0275 at the AIAA 25th Aerospace Sciences Meeting, Reno, NV, Jan. 12-15, 1987; received April 3, 1987; synoptic received May 11, 1987. Copyright © American Institute of Aeronautics and Astronautics, Inc., 1987. All rights reserved. Full paper available at AIAA Library, 555 W. 57th St., New York, NY 10019. Price: microfiche, $4.00; hard copy, $9.00. **Remittance must accompany order.**

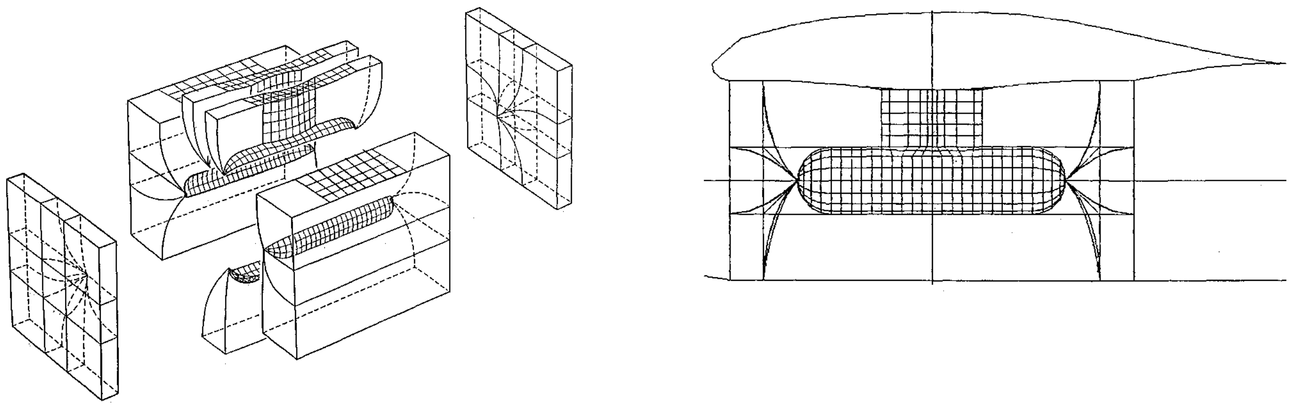
Fig. 1 Exploded view of a 27-block system for a pylon-store.

vision for separate storage of each block on the CRAY solid-state disk (or conventional disk) to allow very large grids to be generated.

As an example, an exploded view of a 27-block system for a pylon-store, constructed so as to transition from a 0-type grid on the store to a rectangular macro-block that can be inserted into a C-type grid about a wing, is shown in Fig. 1. The dotted lines here indicate subblock boundaries.

An auxiliary front-end (Ref. 2) has also been written to set up boundary data for input to the grid code. This auxiliary code builds boundary segments in response to a series of input commands that again are designed to be user-oriented, brief, and easily recognized. The following features are included: 1) generation of generic plane conic-section or cubic curves, 2) generation of cubic space curves, 3) generation of generic conic-section surfaces, 4) generation of cubic surfaces, 5) generation of surfaces by stacking, rotating, or blending curves, 6) extraction and concatenation of surface segments, 7) transformation of surfaces by translation, rotation, and scaling, 8) reversal or switching of point progressions on surfaces, 9) establishment of point distributions by curvature and with specified end, or interior, spacings, 10) establishment of surface parametric grids by transfinite interpolation, 11) generation of tensor-product surfaces, 12) generation of surfaces by transfinite interpolation, 13) generation of grids on curved surfaces, and 14) intersection of surfaces.

Acknowledgment

The development of this code was sponsored by the U.S. Air Force Armament Laboratory, Eglin Air Force Base, under Grant F08635-84-0228; Dr. Lawrence E. Lijewski, monitor.

References

[1]Thompson, J.F., *Project EAGLE Numerical Grid Generation System User's Manual, Vol. III: Grid Generation System*, U.S. Air Force Armament Lab., Technical Rept., AFATL-TR-87-15, 1987.

[2]Thompson, J.R., *Program EAGLE Numerical Grid Generation System User's Manual, Vol. II: Surface Generation System*, U.S. Air Force Armament Lab., Technical Rept., AFATL-TR-87-15, 1987.