Arc Marine

GIS for a Blue Planet

Dawn J. Wright
Michael J. Blongewicz
Patrick N. Halpin
Joe Breman

Foreword by Jane Lubchenco

ESRI Press
REDLANDS, CALIFORNIA

ESRI Press, 380 New York Street, Redlands, California 92373-8100

10 09 08 07 1 2 3 4 5 6 7 8 9 10

Printed in the United States of America

Library of Congress Cataloging-in-Publication Data
Arc marine : GIS for a blue planet / Dawn J. Wright ... [et al.] ; foreword by Jane Lubchenco. – 1st ed.
p. cm.
Includes bibliographical references.
ISBN 978-1-58948-017-9 (pbk. : alk. paper)
1. Oceanography–Geographic information systems. I. Wright, Dawn J., 1961–
GC38.5.A73 2008
551.460285—dc22 2007000708

Ask for ESRI Press titles at your local bookstore or order by calling 1-800-447-9778. You can also shop online at www.esri.com/esripress. Outside the United States, contact your local ESRI distributor.

ESRI Press titles are distributed to the trade by the following:

In North America:
Ingram Publisher Services
Toll-free telephone: (800) 648-3104
Toll-free fax: (800) 838-1149
E-mail: customerservice@ingrampublisherservices.com

In the United Kingdom, Europe, and the Middle East:
Transatlantic Publishers Group Ltd.
Telephone: 44 20 7373 2515
Fax: 44 20 7244 1018
E-mail: richard@tpgltd.co.uk

Cover and interior design by Savitri Brant

Contents

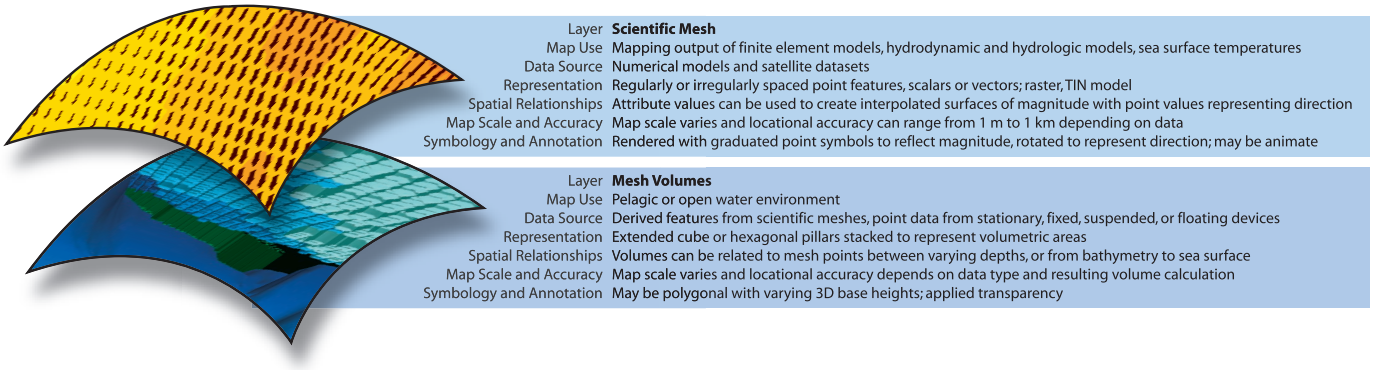
chapter seven

7

Model meshes

The Model Mesh dataset was added to the Arc Marine data model to support numeric modeling and modeling applications within the ArcGIS environment. While this book includes other case studies that illustrate the use of model data, they generally focus on storing and accessing model input data. This chapter focuses on storing and accessing model results, specifically 2D model data and 2D data with multiple values that vary over time. It also further examines data three dimensionally by stacking 2D data at different depths so that each layer represents a unique slice in space. This layered representation allows one to drill down through space to retrieve a single value or multiple values from any number of depths (layers) at a given time step.

This chapter revolves around the case study of the Federal Maritime and Hydrographic Agency of Germany, where Arc Marine has become an important component in its geodata infrastructure and warehouse, which is currently increasing from 80,000,000 scalar values to approximately 3,000,000,000 scalar values.



Introduction

Ocean numerical modeling is a main pillar of marine sciences and services, along with data collected during surveys. The marine community uses operational numerical models to predict currents, sea water temperatures, salinity, water levels, sea state, and other parameters in real or near real time. These important tools support storm surge warning, rescue operations, abatement of marine pollution, ship routing, integrated coastal zone management, and approval processes of offshore facilities. Numerical models are often used for supporting marine survey planning and marine data processing.

The integration of numerical modeling and GIS recently evolved from a relationship of mere exchanging of files to more intimate integration. This integration might consist of either having GIS capabilities embedded into the modeling software or having models executed from within a GIS using GIS-based data sources as the input. One of the principle reasons for this evolution is the ability to integrate or otherwise share the same data sources. GIS has long been able to preprocess and post-process model data and in some cases provide limited display capabilities of model results in conjunction with other spatial features. Historically, this was generally accomplished through an export/convert/import processing of the data by both GIS and modeling software. Now, this requirement of greater integration is more obvious as more input data for models is being collected and stored in large enterprise databases and is being generated and accessed in a client/server environment by multiple users with varying applications. Data sharing is the most basic and essential means of integration between GIS and numerical modeling software. If the software can operate on a common dataset, including model results, then integration advances and user efficiency greatly increases. Geodatabases offer data-sharing capabilities unavailable from earlier file-based data structures. For these reasons, Arc Marine aids the integration between the two by including the needed feature classes and object classes to support the storage of model data.

Numerical modeling

Numerical modeling is the use of sophisticated numerical algorithms with the aid of computers to solve differential equations simulating physical, chemical, and biological processes that occur in nature. For the purpose of this book, the examples are limited to the use of physical models. The accuracy of the numerical model results is dependent on the accuracy of the boundary and initial conditions used to provide the input and the modeler's ability to calibrate and validate the model. It is also dependent on the modeler's ability to determine the appropriate parameters for the model. Numerical modeling relies on different dimensionality—one dimension (1D), two dimensions (2D), and three dimensions (3D)—to accurately portray the physical world. Numerical modeling is generally grouped into categories of either steady state or dynamic.

Numerical models in 1D attempt to simulate or solve the flow along a line (e.g., rivers) with basic fluid mechanics algorithms. Points along that line and data associated with it are used to describe the state of the flow at that location and to interpolate to the next point down the line.

Two-dimensional modeling simulates flow through an area or surface (e.g., floods or statistical surfaces temperature of the water at a given depth). The geometry of a 2D model can be represented by either raster or vector data in the form of nodes and faces. Vector nodes can be used to build either a finite element mesh of irregularly shaped faces defined by three or four nodes, or a regularly spaced mesh of rectangular faces referenced by a linear or even curvilinear 2D coordinate system independent of the underlying projected or geographic coordinate system. The geometry of 3D model is volumetric in nature and is either regularly subdivided into cuboids referring to some linear or curvilinear 3D coordinate system or is a 3D mesh of finite elements (tetrahedrons) representing the volume of interest.

Numerical modeling additionally relies heavily on the element of time. The temporal capability of numerical modeling is the ability to replicate or otherwise forecast a certain phenomenon over time. Whether the model data is collected with recorders or calculated through an interpolation, how the values change or move over time is what modelers find most revealing.

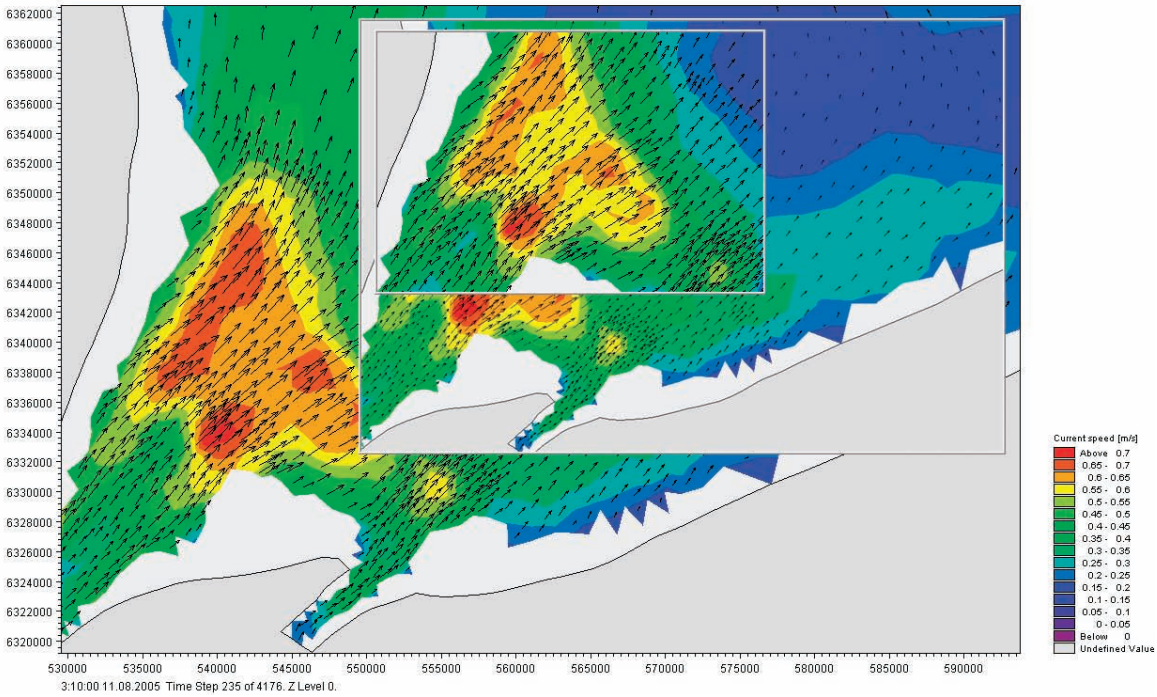


Figure 7.1 This diagram illustrates an example of 2D results from a numeric mesh model developed by Manitoba Hydro using DHI's MIKE 21 software.

Featured case study

The Bundesamt für Seeschifffahrt und Hydrographie (BSH, Federal Maritime and Hydrographic Agency) is a federal authority in Germany under the jurisdiction of the Federal Ministry of Transport, Building, and Urban Development. As a maritime partner to industry, science, and environmental organizations, the BSH provides a wide range of services in the fields of navigation and maritime transportation and the marine environment. BSH has used numerical models for many years to support the services of routine daily simulations and predictions of the North Sea and Baltic Sea dynamics, including current, water level, temperature, salinity, and dispersion of substances. The model system comprises several interacting computer programs that produce output data automatically without intervention. The model is 3D, consisting of two nested 3D models established in spherical coordinates with a horizontal regularly spaced square grid cell size of approximately 6 nautical miles and 1 nautical mile, extending to nine and five vertical depth levels, respectively. The coarser model covers the entire North Sea and Baltic Sea with a total number of 36,703 grid cells. The fine-resolution grid consists of 72,636 cells covering the German coastal areas. In addition, external surges entering the North Sea are computed by

a 2D model having a regular grid spacing of approximately 24 nautical miles. A detailed description of the BSH numerical model system can be found in Dick (2001) or Dick et al. (2001). The case study featured in this chapter uses data of the BSH fine-resolution 3D model grid.

Multiple dimensions

Model data and model results in 2D and 3D generally are represented as a raster or a stack of rasters. Arc Marine can support a vector representation for each of these dimensions, inclusive of a time element. The ProfileLine feature class, explained in detail in chapter 6, can support 1D numerical models. In Arc Marine, MeshFeature provides the feature and object classes necessary for storing data generated by 2D and 3D numerical models. Although not yet possible to define in UML, 2D rasters representing modeled surfaces can be included as part of the data model and subsequent geodatabases. This chapter focuses on the data model's vector representation of 2D and 3D data, the ability and means of accessing the data, and how to render the model data for analytical or presentational purposes.

Meshes and mesh features

MeshPoint features are points that describe data from a 2D or 3D numerical model as either a regularly spaced grid point or as an irregularly spaced node of a finite element mesh or other mesh type. MeshPoint features represent either the center of the cell of a regular 2D or 3D raster (GridPoint subtype), or they define the nodes (NodePoint subtype) of the finite element faces. The MeshPoint feature class is a subclass of MarineFeature and consequently inherits the FeatureID and FeatureCode attributes, which are used by several relationship classes. The MeshPoint feature class also adds attributes such as IPosition, JPosition, and KPosition that are traditionally used in modeling to describe the row, column, and depth location of a point within a mesh. The MeshID attribute is used as a key field for identifying in which Mesh a point participates. The PointType field is a subtype field for determining if the point is a GridPoint and used in a regular mesh or a NodePoint and used in an irregular mesh.

The MeshElement feature class is a polygonal feature class that inherits from MarineFeature for representing the face of an irregular mesh. At least three MeshPoint features must define a MeshElement, but it might also be defined by four MeshPoints. This feature class has four attributes in addition to those inherited from MarineFeature—Node1ID, Node2ID, Node3ID, and Node4ID. These attributes are used as key fields for identifying the FeatureID of the MeshPoint that marks the corners of the MeshElement. In this case, the PointType attribute of MeshPoint should be set to 2, identifying that the MeshPoint is subtyped as a NodePoint as opposed to a GridPoint.

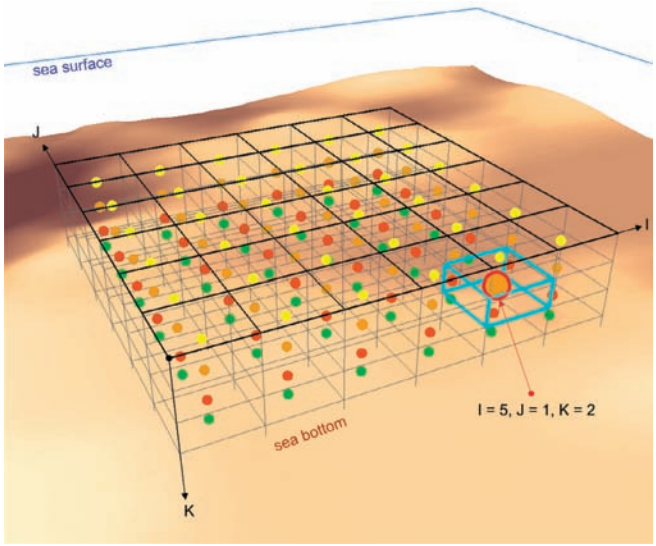


Figure 7.2 This diagram illustrates the 3D location of points of the MeshPoint feature class within a Mesh, where the points are symbolized based on their KPosition (depth layer) in the Mesh.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

MeshPoint features can participate in one or more meshes and meshes of different dimensionality, and can be referenced by either the coordinate property of their shape or their 3D location within a Mesh. The points, however, have no attributes that reflect information about the mesh itself. It is the Mesh object class that defines the Mesh and which points are used.

The Mesh object class defines the size, shape, and dimensionality of the mesh. The MeshID serves as a key field to uniquely identify the mesh. The relationship class named MeshHasPoints is used to link the mesh with the points in the MeshPoint feature class.

The MeshID is a user-defined identifier and is not an auto-generated value. The MeshHasPoints relationship class is a one-to-many relationship, allowing for one mesh to have many points. Furthermore, features from the MeshPoint feature class can participate in more than one Mesh, and Meshes of varying dimensionality can be created from the same feature class of MeshPoints. The TotalPoints attribute is simply the total number of points actually participating in the Mesh, while the attributes NoOfPointsI, NoOfPointsJ, and NoOfPointsK store the possible number of points in the respective three directions. The Dimension field is defined by a CodedValueDomain called MeshType to determine the dimensionality of the mesh as linear and 1D, a 2D area, or a 3D volume.

The TotalPoints is not necessarily the product of NoOfPointsJ by the NoOfPointsI by the NoOfPointsK, spanning the 3D mesh space. It is not necessarily true that the same number of points populates each depth layer defined by the mesh. In a simple example, a mesh could have a value of three for the NoOfPointsK, indicating that this volumetric mesh has three depth layers. The TotalPoints might be set to 22, where the first and second layers each have nine points and the third layer has only four points. In ocean numerical modeling, TotalPoints is often referred to as the number of wet grid points.

Vector and scalar quantities

Numerical models generally produce time-varying quantities of scalar or vector values. The VectorQuantity or the ScalarQuantity tables further define the Mesh Points so that data will be stored depending on its scalar or vector nature, respectively. The table entries also carry a time stamp so that the values of a particular quantity for a single location can vary over time. Points can have values in both tables, and these are managed through the MeshPointHasVectors and MeshPointHasScalars relationship classes. These relationships are one-to-many relationships, so a point can have one or more Vector Quantities and one or more Scalar Quantities.

The VectorQuantity table uses the FeatureID attribute as a key field for participating in the relationship associating vector values to specific points of the MeshPoints feature class and the ParameterID as a key field for participating in the relationship to the Parameter object class. The attributes, XComponent, YComponent, and ZComponent are used to define the three quantities of a vector. It will then be up to the user or a specific application to calculate the direction and magnitude of the vector (e.g., current direction and current speed). For storing the calculated magnitude and direction explicitly, users are free to add the necessary attributes to the VectorQuantity table. TimeValue is the attribute storing the associated time step of these values. The x, y, and z components could vary for every time step, indicating a FeatureID for a given point and a ParameterID for a given parameter.

The Scalar Quantities table also uses the FeatureID as the key field for participating in a similar relationship class with the MeshPoint feature class called MeshPointHasScalars. This table also has the ParameterID attribute used as the key field for the relationship class connecting the Parameter table. The DataValue attribute is used to store the actual value of the scalar, and the TimeValue attribute is used to store the time stamp of the data value.

Having vector and scalar data as attributes associated with the same set points, the same mesh can be rendered differently for each parameter type. Rendering vector data together with scalar data of the same mesh provides the ability to illustrate attributes in relation to each other. The BSH will often choose to look at current speed and direction in relation to sea temperature at varying depths and different time steps.

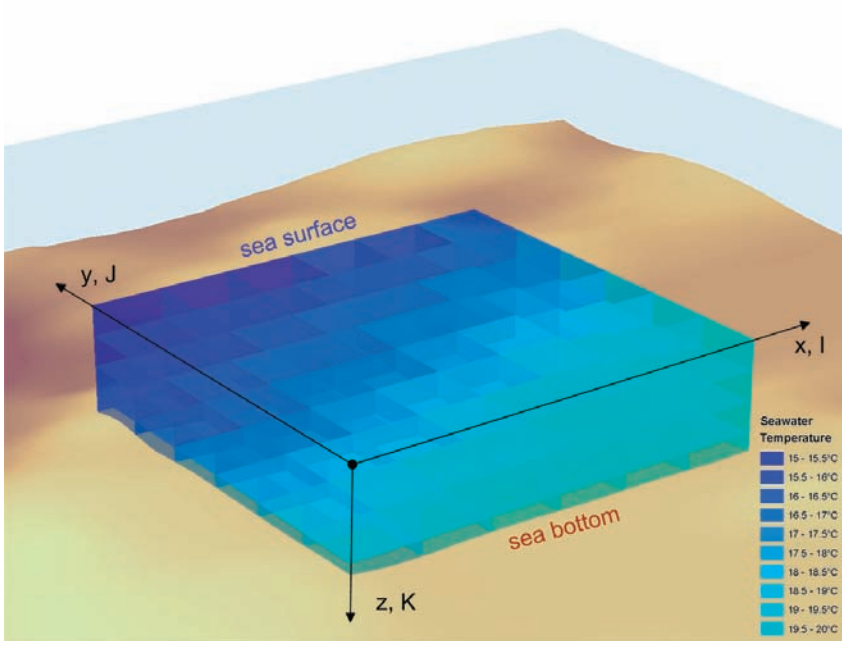


Figure 7.3 The capabilities of rendering a Mesh based on the scalar quantities of the Mesh Points.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

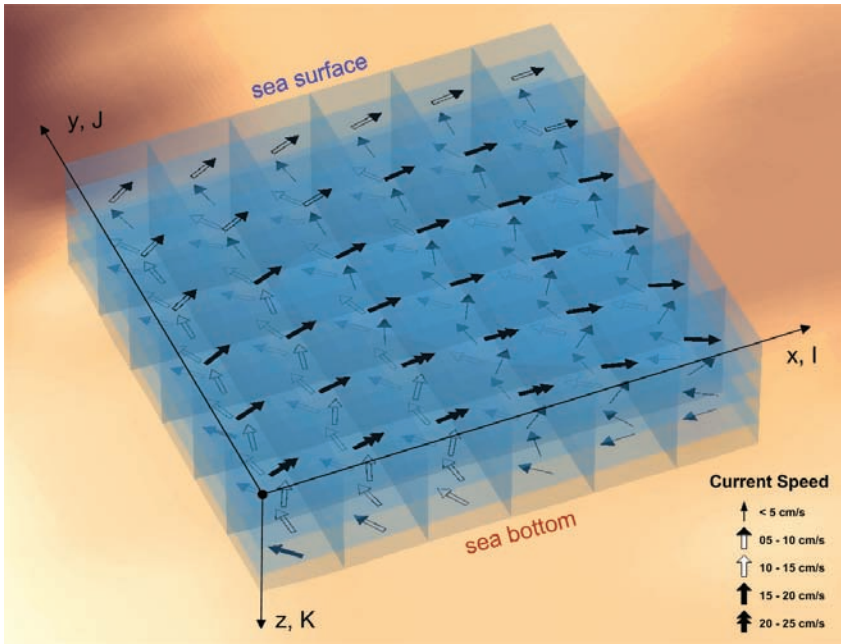


Figure 7.4 The capabilities of rendering a Mesh based on the vector quantities of the Mesh Points.

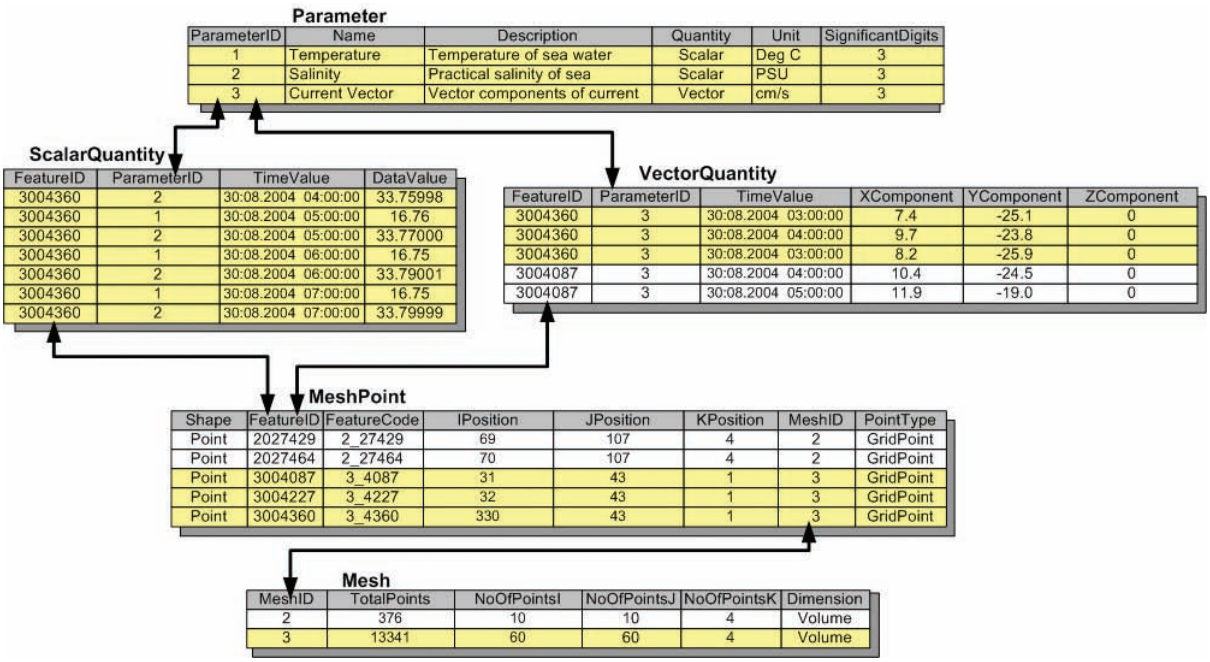
Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

Parameters

The Parameter object class is designed as a lookup table for all parameters for which the user has data in the geodatabase. The Parameter table stores basic attributes describing the parameters. Users can easily add attributes to this table to support other specific data types and applications. The Parameter table uses the ParameterID as the unique identifier for each parameter. The Name, Description, Unit, and SignificantDigits are then used to define the parameter. The Quantity attribute uses a coded value domain for defining whether it is a Scalar, Vector, or some other quantity.

This table can be approached in various ways. When users query the table for a specific parameter, access is provided through the relationship classes to the actual data values and ultimately the associated features. Likewise, when investigating specific features and data values, the same set of relationship classes provides access to the Parameter table and to information describing the data values.

Figure 7.5 illustrates the structure of the Mesh Features dataset. Its feature classes, object classes, and their relationships provide several access routes to the data. From the spatial perspective, starting with a Mesh (MeshID = 3) and going through the MeshHasPoints relationship class, all participating point features (e.g., FeatureID = 3004360, 3004227, 3004087)



Parameter

ParameterID	Name	Description	Quantity	Unit	SignificantDigits
1	Temperature	Temperature of sea water	Scalar	Deg C	3
2	Salinity	Practical salinity of sea	Scalar	PSU	3
3	Current Vector	Vector components of current	Vector	cm/s	3

ScalarQuantity

FeatureID	ParameterID	TimeValue	DataValue
3004360	2	30:08.2004 04:00:00	33.75998
3004360	1	30:08.2004 05:00:00	16.76
3004360	2	30:08.2004 05:00:00	33.77000
3004360	1	30:08.2004 06:00:00	16.75
3004360	2	30:08.2004 06:00:00	33.79001
3004360	1	30:08.2004 07:00:00	16.75
3004360	2	30:08.2004 07:00:00	33.79999

VectorQuantity

FeatureID	ParameterID	TimeValue	XComponent	YComponent	ZComponent
3004360	3	30:08.2004 03:00:00	7.4	-25.1	0
3004360	3	30:08.2004 04:00:00	9.7	-23.8	0
3004360	3	30:08.2004 03:00:00	8.2	-25.9	0
3004087	3	30:08.2004 04:00:00	10.4	-24.5	0
3004087	3	30:08.2004 05:00:00	11.9	-19.0	0

MeshPoint

Shape	FeatureID	FeatureCode	IPosition	JPosition	KPosition	MeshID	PointType
Point	2027429	2_27429	69	107	4	2	GridPoint
Point	2027464	2_27464	70	107	4	2	GridPoint
Point	3004087	3_4087	31	43	1	3	GridPoint
Point	3004227	3_4227	32	43	1	3	GridPoint
Point	3004360	3_4360	330	43	1	3	GridPoint

Mesh

MeshID	TotalPoints	NoOfPointsI	NoOfPointsJ	NoOfPointsK	Dimension
2	376	10	10	4	Volume
3	13341	60	60	4	Volume

Figure 7.5 This diagram illustrates the connectivity of the data. The structure allows users to approach the data from several directions, either spatially by querying a Mesh for points and then determining the data available, or to find all points of a certain parameter type.

form the selected dataset of the MeshPoint feature class. Within the feature class, the points can be further queried based on their depth position within the Mesh (KPosition = 1). All scalar data values associated with the selected points at this depth can be found, proceeding from the MeshPoint feature class through the MeshPointHasScalars relationship to the ScalarQuantities table. In the example, the point with the FeatureID = 3004360 has scalar values for two different Parameters (ParameterID = 2 and ParameterID = 1, Salinity and Temperature, respectively). Additionally, through the MeshPointHasVectors relationship class, the same set of points at this depth has vector quantities. Furthermore, for the scalar quantities and the vector quantities, each value for each of the points changes for each time step. This adds the fourth dimension to the data and another means of organizing specific values. That is, the values for these select points (or this mesh) for a specific parameter (Temperature) will change and can be rendered differently on a time-step-by-time-step basis. This allows for comparison of the data from one point in time to another. The results could be further narrowed by querying for a particular time period. The ParameterID is the key field by which the particular parameter (e.g., Salinity, Temperature) in the Parameter table can be identified.

Conversely, the path through the relationships and tables are in reverse of the previous example if the approach is to query by parameter type and to locate the mesh features or an entire mesh that represents a particular parameter. To find the type of quantity data present, the Parameter table can be viewed for all available types, either scalar or vector. If Current is selected, for example, through the ParameterHasVectors relationship class, all of the records in the VectorQuantity table are selected with the corresponding ParameterIDs. In this table, the FeatureIDs and their corresponding data values (XComponent, YComponent, and ZComponent) for the varying time steps can be seen. This data can be further queried for a particular range of values or time period. Through the MeshPointHasVectors relationship class, the actual point features that meet the criteria can be highlighted in the MeshPoint feature class. Finally, ArcMap can render and symbolize the entire mesh or the subset of the mesh that these points represent, based on one or more of these values.

Rendering mesh features

At the Marine Sciences department of the Federal Maritime and Hydrographic Agency (BSH) of Germany in Hamburg, Dr. Jürgen Schulz-Ohlberg and Kai Jancke are working with a team on the development of a customized, GIS-based marine data management, information, and analysis system. The system aims to achieve a more homogeneous and efficient treatment of disparate marine datasets comprising mainly oceanographic and marine chemical measurements as well as numerical model results. The BSH has contributed extensively to the design of the Mesh Features dataset and is using an extended version of Arc Marine. The BSH is implementing a centralized large-scale ArcSDE/Oracle geodatabase storing several hundred million objects, the majority of them in data resulting from daily numerical model predictions. In detail, holding 30 days of hourly BSH mesh data makes for approximately 72 million rows per parameter in the ScalarQuantity and

VectorQuantity tables, for the parameters Temperature, Salinity, and Current. Everyday after the forecast calculation is complete, three days worth of data are replaced in the geodatabase. Given that the forecast extends for three days into the future, two days are updated, one is added, and the oldest day is dropped.

Users access data through the Parameter table for the GIS application based on the Arc Marine data model being developed at the BSH. This application is an ArcGIS tool that extracts mesh data based on user-specified values. It uses the Parameter table and its relationship classes to populate the content of the dialog elements, for example, selection lists, with the available data. In the example illustrated in figures 7.6 and 7.7, the application presents available data in the existing geodatabase associated with the two meshes described previously. From these two meshes, the user can query and create a new point feature class representing either a particular scalar or vector parameter at a specified depth and for a specific time step. In figure 7.6, a MeshPoint feature class is created from the points at the second depth layer of a selected Mesh (MeshID = 3) and at a specified time step for the parameter of Current Vector. Likewise in figure 7.7, the same Mesh (MeshID = 3) is being queried for all temperature values at depth layer 3 for a particular moment in time.

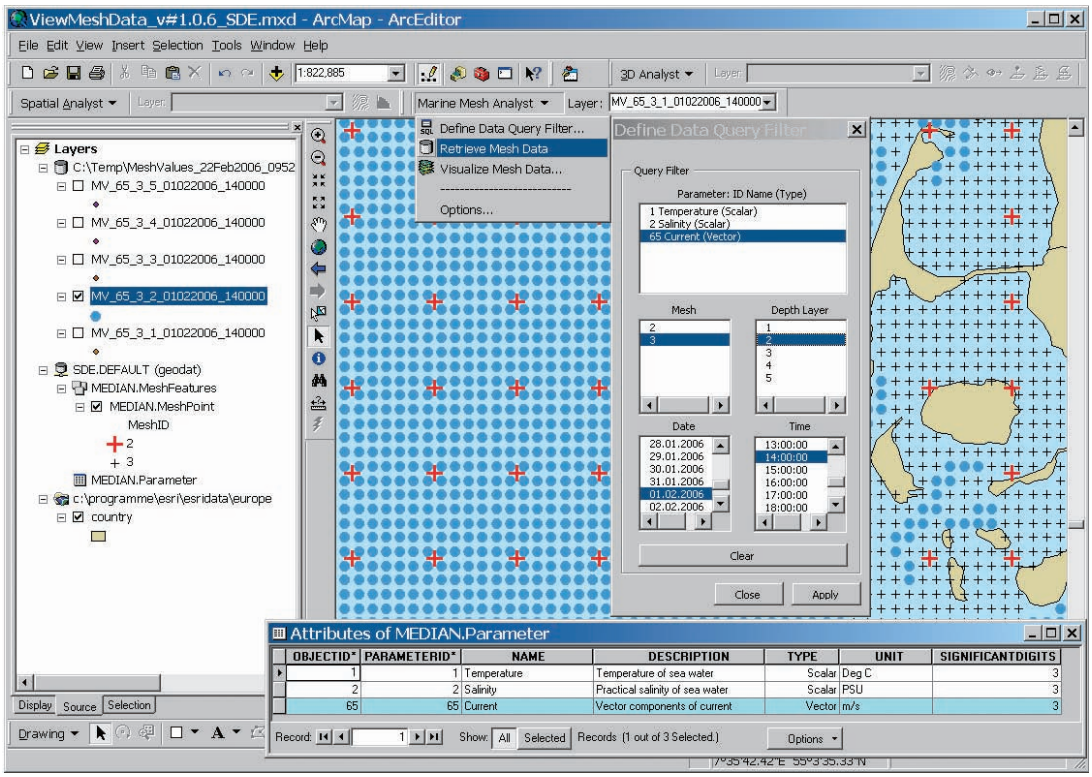


Figure 7.6 At BSH, MeshPoint features can be extracted from the geodatabase by specifying the Current Vector parameter, the second depth layer (Kposition = 2), and a specific date and time.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

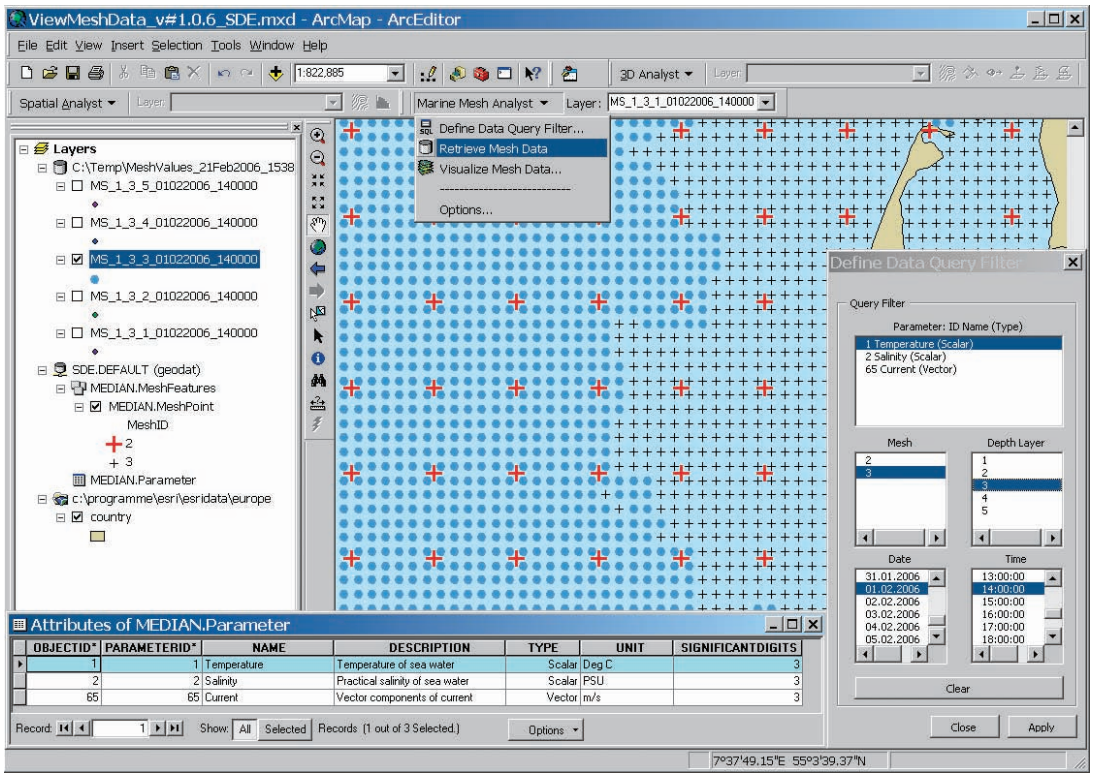


Figure 7.7 Mesh features and their associated temperature values may be extracted from the geodatabase using the BSH application by specifying the Temperature parameter and the values from the third depth layer (KPosition = 3) and a specific data and time (time step) and an existing mesh (MeshID = 3).

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

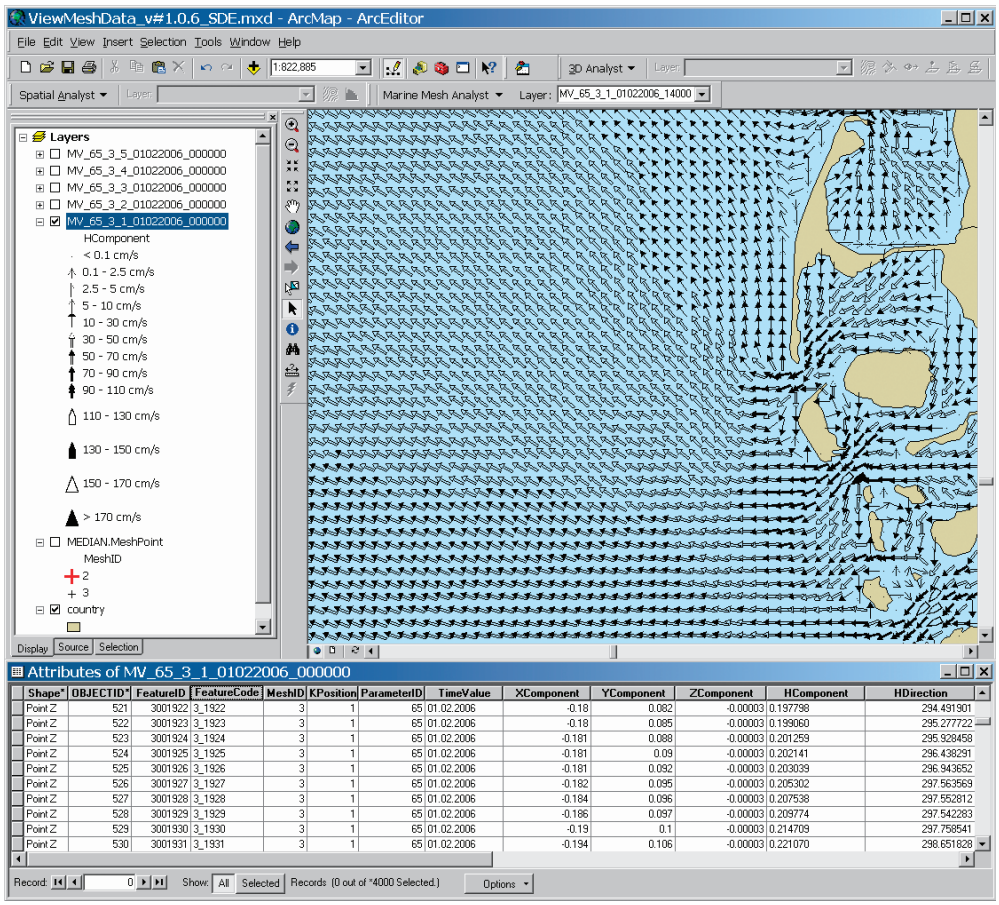


Figure 7.8 The sea surface current distribution near the German coast for a specific date and time uses vector data extracted from the geodatabase. The speed and the direction of the current are rendered as graduated and rotated symbols.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

These feature classes can then be further rendered in various ways to visualize and present the model values. In the example illustrated in figure 7.8, the data from the first query is rendered using the vector values of the parameter Current. The newly created MeshPoint feature class holds the XComponent, YComponent, and ZComponent and is joined with selected attributes of the VectorQuantity object class. Using the BSH application, the vector properties of magnitude and direction are automatically calculated from the x-, y-, and z-components and stored in the attributes HComponent and HDirection. The magnitude (speed) and the direction values determine the graduated size and the angle of each symbol. If this process repeats for each possible time step, the user can see how the data evolves over time at the specified depth. And if the complete procedure repeats for each depth layer, then the user can visualize the variation in the values over time and over varying depths.

Furthermore, statistical surfaces, which might be more revealing and intuitive than rows and columns of points, can be interpolated from mesh points associated with scalar values. Users can interpolate these values to a raster resulting in a distribution of values across the

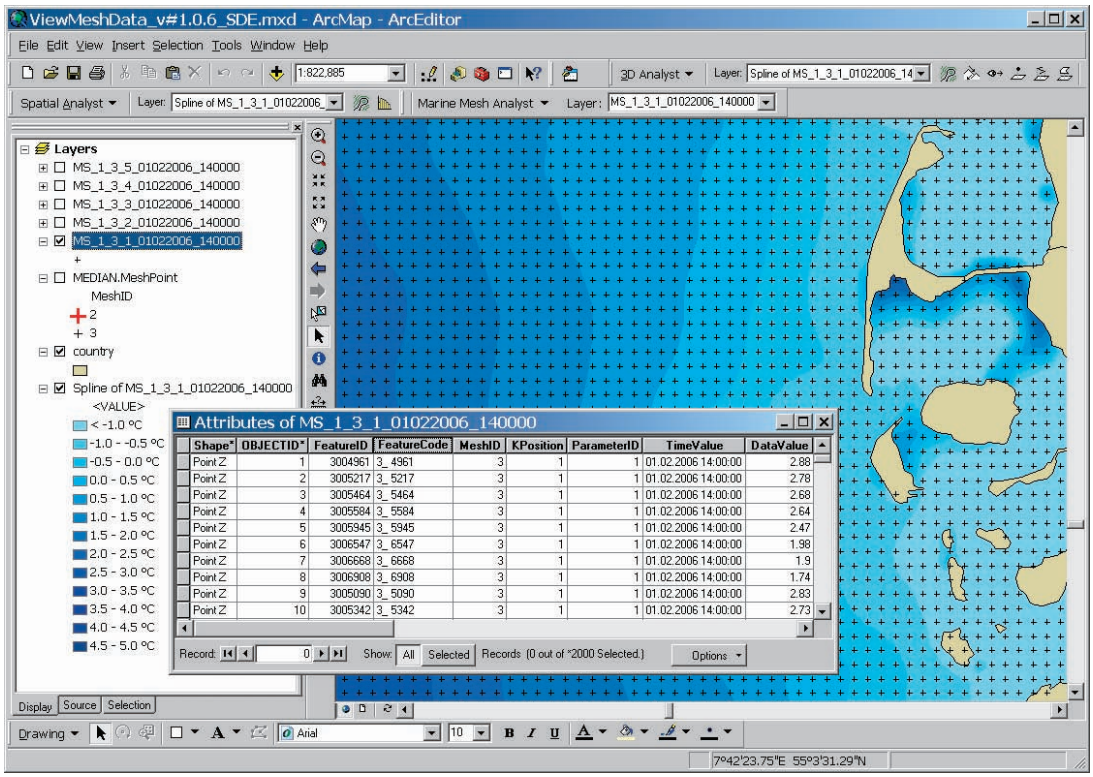


Figure 7.9 The color-coded sea surface temperature distribution near the German coast represents a specific date and time. The underlying raster layer is an interpolated surface of temperature data values extracted from the MeshPoints feature class using the BSH application.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

surface. Continuing the example of the second query using temperature data in figure 7.7, the temperature values associated with the points are used in a spline interpolation showing the temperature distribution at a given depth and at a specified date and time. This effect is illustrated in figure 7.9.

Because the mesh points generating the temperature surface also have vector values for the Current, the two data types can be graphically overlaid in ArcMap. The resulting map displays the color-coded temperature distribution and the speed and direction of the current rendered as rotated graduated point symbols.

This data becomes even more informative when rendered in 3D. This can be achieved in different ways. In one approach taken at the BSH, a scalar mesh quantity (temperature) may be interpolated to a raster (as shown in figure 7.9 and described above) repeatedly for all depth layers of a single mesh. ArcScene can display the resulting set of raster layers, one stacked on the other with the base height of each raster layer set to the depth value of the corresponding model layer. Controlling the transparent display property of the rasters gives a pseudo 3D impression of the horizontal and vertical temperature distribution.

Another approach produces a similar looking result but avoids the intermediate steps of the raster interpolation. This approach works when the cuboid cells of a model mesh have already been built as a regular 3D raster, which may be regarded as a vertical stack of 2D rasters (compare to figure 7.2). Instead of using a representation by a number of points stored in the MeshPoint feature class, each of these 2D rasters can be represented by an identical number of conjoined regular rectangles describing the upper horizontal faces of the cuboid cells in the numerical model. Like the points in the MeshPoint feature class, these rectangles need to be created only once in advance and may be retained in an additional polygonal feature class (referred to as MeshFace below). In this feature class, the FeatureID is obtained from the ID of the point coincident with the center of the polygon. The depth of each rectangle should be stored as the z-coordinate property of the feature's geometry enabling rendering in 3D. Using the temperature data as an example again, the temperature values associated with the points for a single mesh for all depth levels at a specified date and time can easily be joined to the MeshFace feature class via the FeatureID attribute. ArcScene can then render the MeshFace feature class using graduated colors and transparent display for the temperature values (figure 7.10). As with interpolated rasters, this type of representation reveals—while not fully 3D—the horizontal and the vertical distribution of the mesh values.

To allow for a combined analysis or comparison of temperature and current values in 3D as represented in figure 7.11, the current vector data values are extracted from the geodatabase for each of the depth layers represented by the NoOfPointsK attribute. These new MeshPoint values are joined to the points coincident to the polygons of the MeshFace feature class and graphically overlaid with the stacked temperature layers in ArcScene. The current data is rendered using graduated arrow symbols as illustrated in figure 7.11. The rotation angle of the arrows indicates the ocean current's direction, and the size of the arrows is scaled according to the ocean current's speed. This results in a true rendering of scalar and vector values for the same mesh point at each of the depth layers. By adding the fourth dimension of time, the scene can be animated to show the change in values of depth over time.

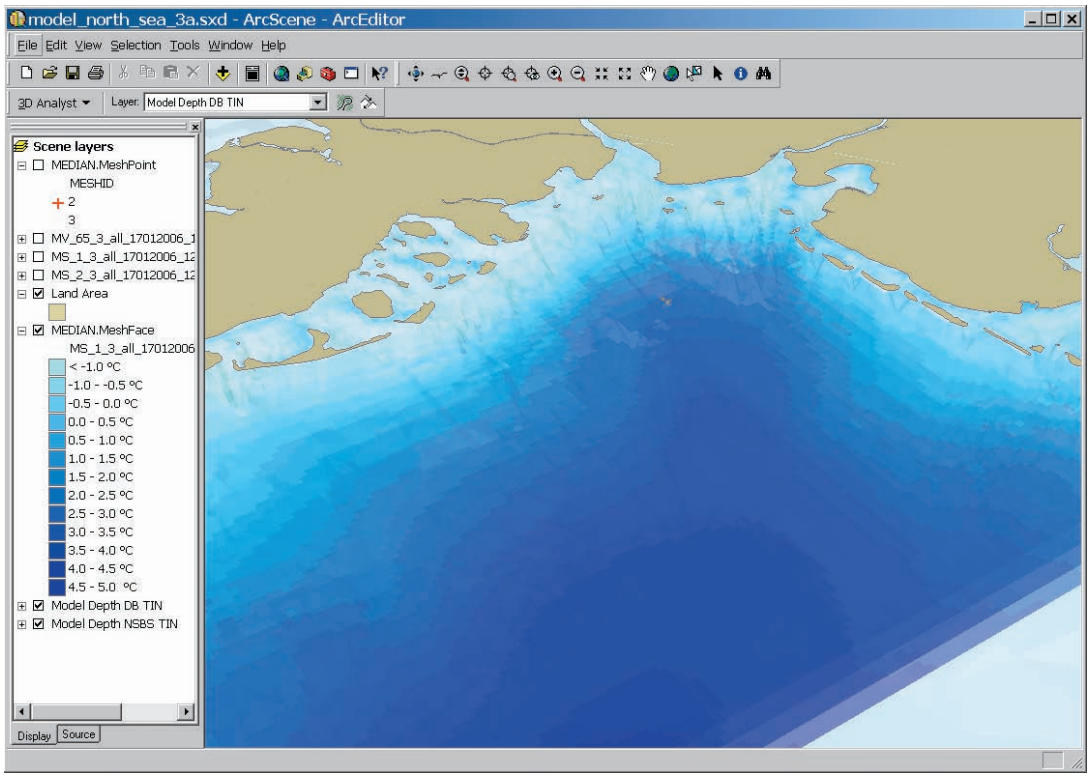


Figure 7.10 This 3D view of vertically stacked polygonal layers simulates the raster layers, color coded by temperature value in the German Bight for a specific date and time (for further explanation, see text).

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

This data can be furthered queried or accessed and then presented using standard ArcGIS tools. Figure 7.12 shows that once the raster data layers for temperature are created for each depth layer in a Mesh, the graphing tools from the ArcGIS 3D Analyst extension can be applied for extracting data. In this example, the tools extract data along a line for a certain depth layer. The data is graphed to display the change in temperature along that line. Additionally, the tools create a drilling effect through the various raster layers to see the varying temperatures for the existing depths at a given location.

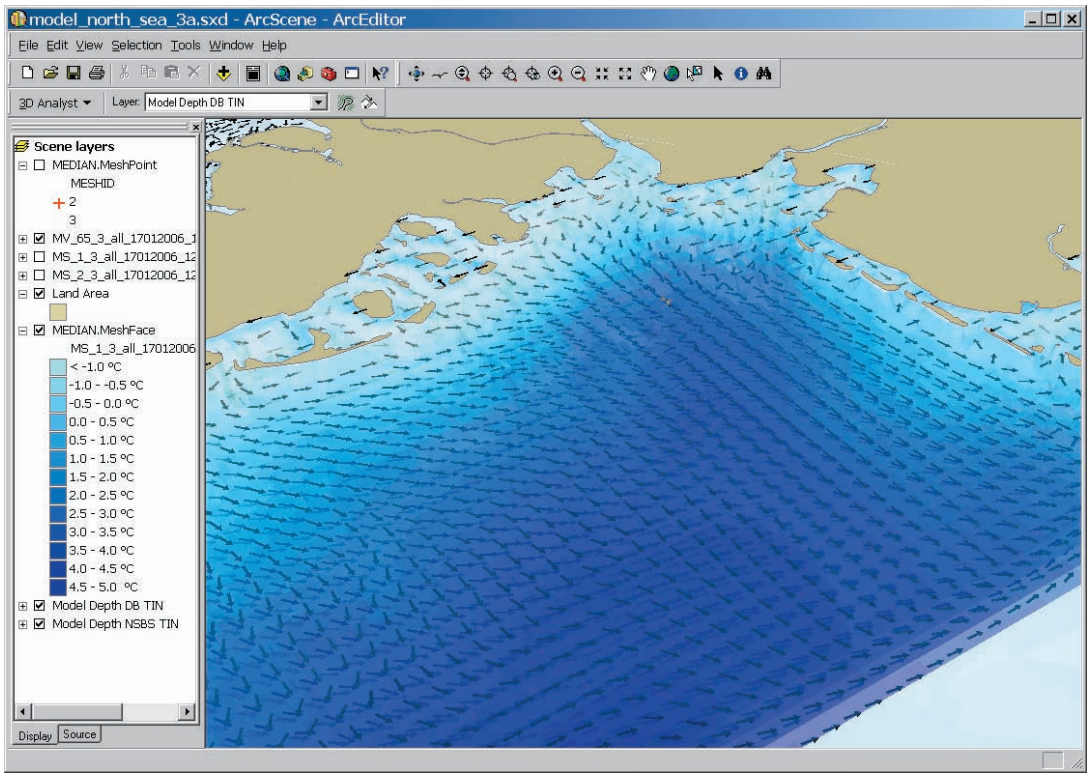


Figure 7.11 The combined 3D view of stacked temperature and current distribution in the German Bight is shown for a specific date and time, with the current data being rendered at all model depth levels using graduated and rotated arrow symbols.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

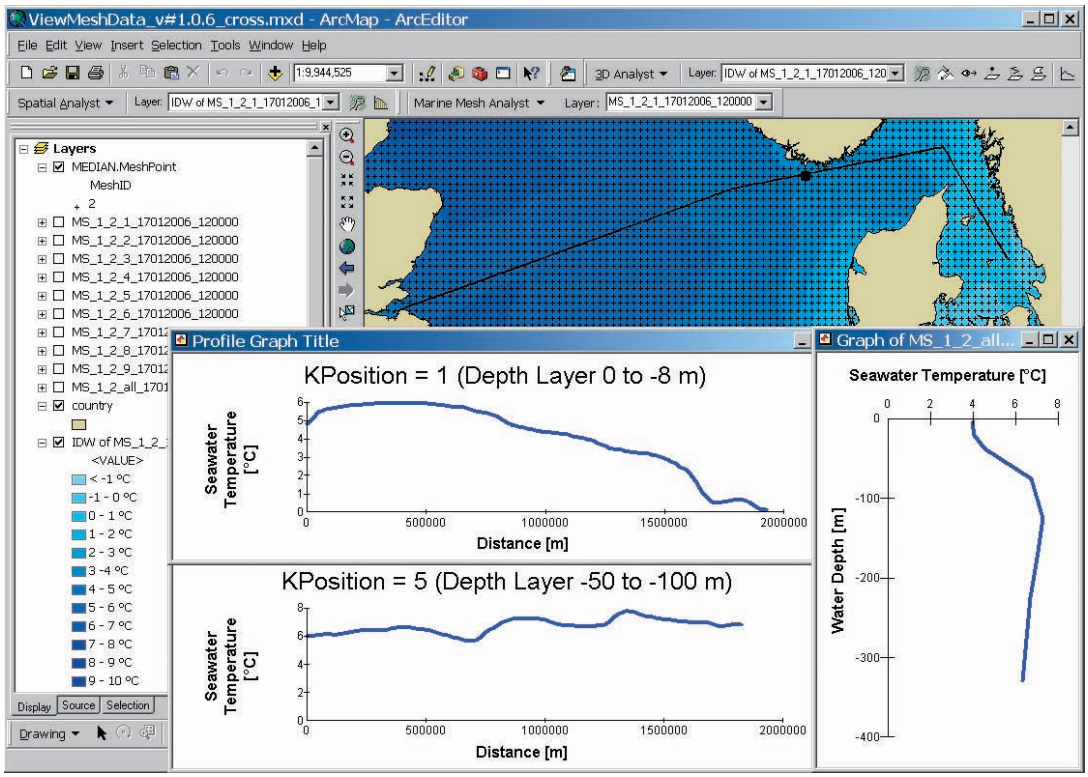


Figure 7.12 The use of tools in the ArcGIS 3D Analyst extension allow for data extraction, either along a line or at a given point from the raster layers interpolated from various depth layers of a Mesh.

Provided by Jürgen Schulz-Ohlberg, Federal Maritime and Hydrographic Agency, Germany. Used with permission.

Conclusion

Debate continues over the logic of storing model results, which can be voluminous, in a database. However, no one argues about the querying capabilities of a relational database. Users can apply the same capabilities to model results by storing the data as a normalized vector structure in a database, especially a spatially enabled database such as the geodatabase. As seen in the case study provided by the BSH, the Mesh Features dataset of Arc Marine provides a good means for storing the data so that it can be queried and accessed from varied perspectives for different applications.

Arc Marine class definitions featured in this chapter

FEATURE CLASSES	**MeshPoint**—A feature class representing point features from a 1D, 2D, or 3D numerical model as either an equally spaced mesh or irregularly spaced mesh, where the values of each point will change over time.			
	Subtype	GridPoint NodePoint		
	Notes			
	Properties	HasZ = True		
	Fields	FeatureID	Integer	A geodatabase-wide unique identifier and key field for participating in relationships
		FeatureCode	String	A user-defined code used for identifying a feature
		IPosition	Integer	A feature's location in the I direction for a given mesh
		JPosition	DateTime	A feature's location in the J direction for a given mesh
		KPosition	Double	A feature's location in the K direction for a given mesh
		MeshID	Integer	A user-defined identifier for a given mesh
		PointType	Coded Value Domain	A coded value domain defining the subtype to be one of the following: 1 = GridPoint 2 = NodePoint
	MeshElement—A polygonal feature class representing a face of a finite element.			
	Subtype	None apply		
	Properties	None apply		
	Notes	MeshElements are polygonal features comprised of a combination of either three or four nodes		
	Fields	FeatureID	Integer	A geodatabase-wide unique identifier and key for participating in relationships
		FeatureCode	String	A user-defined code used for identifying a feature
		Node1ID	Integer	A key field for identifying the FeatureID of a MeshPoint that represents a corner of the MeshElement
		Node2ID	Integer	A key field for identifying the FeatureID of a MeshPoint that represents a corner of the MeshElement
		Node3ID	Integer	A key field for identifying the FeatureID of a MeshPoint that represents a corner of the MeshElement
		Node4ID	Integer	A key field for identifying the FeatureID of a MeshPoint that represents a corner of the MeshElement

OBJECT CLASSES				
	Mesh—Stores necessary information for defining the dimension and structure of a Mesh grid.			
	Notes	A Mesh is a collection of points forming either a line, area, or volume features in nature.		
	Fields	MeshID	Integer	The unique identifier for a mesh
		TotalPoints	Integer	Total number of points participating in the Mesh
		NoOfPointsI	Integer	Total number of points available in the I direction within a Mesh
		NoOfPointsJ	Integer	Total number of points available in the J direction within a Mesh
		NoOfPointsK	Integer	Total number of points available in the K direction within a Mesh
		Dimension	Coded Value Domain	A coded value domain defining the subtype to be one of the following: 1 = Linear 2 = Area 3 = Volume
	VectorQuantity—This table stores the vector values associated with a MeshPoint.			
	Notes			
	Fields	FeatureID	Integer	A key field for relating this table to a feature class
		ParameterID	Integer	A key field for relating this table to the Parameter table
		XComponent	Double	The Vector x component value
		YComponent	Double	The Vector y component value
		ZComponent	Double	The Vector z component value
		TimeValue	DateTime	The time stamp assigned to the vector value
	ScalarQuantity—This table stores the scalar values associated with a MeshPoint.			
	Notes			
	Fields	FeatureID	Integer	A key field for relating this table to a feature class
		ParameterID	Integer	A key field for relating this table to the Parameter table
		DataValue	Double	The measured scalar value
		TimeValue	DateTime	The time stamp assigned to the scalar value

OBJECT CLASSES (cont'd)	**Parameter**—Stores some basic information about the represented parameters.			
	Notes	This table is designed to be a lookup table of parameter types. It can be used as a mechanism for querying a geodatabase for a specific parameter and then finding values of a particular type in related tables. Alternatively, it can be used as a lookup table of parameter types for a particular value.		
	Fields	ParameterID	Integer	The unique identifier of a specific parameter
		Name	String	The name of a parameter
		Description	String	The description of a parameter
		Quantity	Double	The quantity type for a parameter. A coded value domain defining the subtype to be one of the following: 1 = Other 2 = Scalar 3 = Vector
		Unit	String	The unit of measure for a parameter
		Significant Digits	Integer	The number of significant digits defining the precision of this parameter

RELATIONSHIPS			
	MeshHasPoints	1 : *	One Mesh has one or more MeshPoints
	MeshPointHasScalars	1 : *	One MeshPoint can have zero or many Scalar values
	MeshPointHasVectors	1 : *	One MeshPoint can have zero or many Vector values
	ParameterHasScalars	1 : *	One Parameter can have zero or many Scalar values
	ParameterHasVectors	1 : *	One Parameter can have zero or many Vector values

References

Dick, S. 2001. Operational ocean modeling forecasting and applications. In *Operational Oceanography, Scientific Lectures at JCOMM-I.* Akureyri, Iceland: WMO/TD-No. 1086, JCOMM Technical Report No. 14.

Dick S., E. Kleine, S. H. Müller-Navarra, and H. Komo. 2001. The Operational Circulation Model of BSH (BSHcmod), model description and validation. Berichte des BSH Nr. 29/2001. Hamburg, Germany: Bundesamt für Seeschifffahrt und Hydrographie (BSH, Federal Maritime and Hydrographic Agency).

Case study content

Michael Blongewicz, DHI Water & Environment
Jürgen Schulz-Ohlberg, Bundesamt für Seeschifffahrt und Hydrographie (BSH)
Kai Jancke, Bundesamt für Seeschifffahrt und Hydrographie (BSH)