GeoMedia

Data Integration and Validation

Validating Vector Data
The information contained in this document (the “Work”) is the exclusive property of Intergraph Corporation (“Intergraph”). The Work is protected under United States copyright law and other international copyright treaties and conventions, to include the Berne and Geneva Phonograms Conventions, the WIPO Copyright Treaty, and the World Trade Organization.

No part of the Work may be reproduced, stored, or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by Intergraph. All such requests should be sent to the attention of Manager of Customer Education of the Hexagon Geospatial Division of Intergraph (“Hexagon Geospatial”) at the following address:

Hexagon Geospatial
5051 Peachtree Corners Circle
Norcross, Georgia 30092-2500 USA

Phone: 770 / 776-3651
Fax: 770 / 776-3694
Support Services: 800 / 953-6643
Customer Education: 800 / 373-2713
Web: www.hexagongeospatial.com

Warning

The Work, and the software that is the subject of the Work, including icons, graphical symbols, file formats, and audio-visual displays, may be used only as permitted under license from Intergraph; contain confidential and proprietary information of Intergraph and/or third parties which is protected by patent, trademark, copyright and/or trade secret law; and may not be provided or otherwise made available without proper authorization. The Work may not be reproduced in any manner, including for resale or redistribution, without the prior written permission of Intergraph. Use of the software during the training course shall be in accordance with the End User License Agreement (“EULA”) delivered with the software. Neither the software nor the software documentation may be reproduced in any manner, without the prior written permission of Intergraph.

About This Manual

The Work is an instructional document designed to be an integral part of the training course offered by Hexagon Geospatial. Hexagon Geospatial believes the information in the Work is accurate as of its publication date. Any specifications cited in the Work are subject to change without notice. The information and the software discussed in the Work are subject to change without notice. Intergraph is not responsible for any errors that may appear in the Work.

Intergraph, the Intergraph logo, ERDAS, ERDAS IMAGINE, IMAGINE Essentials, IMAGINE Advantage, IMAGINE Professional, IMAGINE VirtualGIS, GeoMedia, ImageStation, Video Analyst, Stereo Analyst, TopoMouse, Hexagon Smart M.App, and Power Portfolio are trademarks of Intergraph Corporation. Hexagon and the Hexagon logo are registered trademarks of Hexagon AB or its subsidiaries. Other brands and product names are trademarks of their respective owners.

Acknowledgments

The data used in this tutorial was made available through the courtesy of United States Geological Survey (http://www.usgs.gov).
Table of Contents

This Tutorial........................................................................................................................................ vii

Section 1: Validating Vector Data........................................................................................................ 1
Exercise 1: Geometry Validation ............................................................................................................ 3
Exercise 2: Connectivity Validation ...................................................................................................... 13
Exercise 3: Automated Validation ...................................................................................................... 25
This Tutorial

Tutorial Objective
When accepting data from third party sources the data is not always perfect. GeoMedia includes tools to help you detect and correct vector data issues. This tutorial will familiarize you with those tools.

Tutorial Data Set
The exercises in this tutorial make use of vector data.

Tutorial Text Conventions
There are several conventions used throughout the tutorial:
- Ribbon bar items are shown as: On the Aaa tab, in the Bbb group, click Ccc.
- Dialog box names, field names, and button names are depicted using Bolded Text.
- Information to be entered, either by selecting from a list or by typing, is depicted using Italicized Text.

Tutorial Prerequisites
There are no prerequisites for making use of this tutorial; however, having a basic understanding of the GeoMedia Desktop will be helpful in expediting the execution of certain steps within this tutorial.
Section Objective
Data that is not geometrically accurate can lead to inaccurate analysis and conclusions as well as display wrong information. Validation is an important step in producing accurate datasets. The Validate Geometry tools in GeoMedia detect geometric anomalies on a single feature. In this section you will learn:

- How to utilize the geometry validation tools.
- How to utilize the connectivity validation tools.
- How to run an automated validation.

Tools Used
GeoMedia Advantage or GeoMedia Professional Vector Validation Tools.
Exercise 1: Geometry Validation

**Objective:**
Identify geometric anomalies in features from internal/external sources for evaluation and correction.

**Task 1: Validate Geometry**

Validate Geometry **detects anomalies** for all the features in one or more feature classes. The structure of individual features is validated.

**Validate Geometry Standard Anomaly types**

- **Empty Geometry** refers to features that have an empty geometry component. (e.g. A composite or collection without any components or a geometry that is returned as an empty variant).

- **Unknown Geometry Type** refers to those features whose geometry type does not correspond to a type recognized by GeoMedia.

- **Invalid Geometry** refers to those features whose geometry type does not match the geometry’s delineation.

- **Invalid Geometric Component** refers to those features whose geometry is syntactically correct (i.e. passed the check as specified by the Invalid Geometry anomaly), but whose specification does not define a valid geometric component. Included in the definition of invalid geometries are the following:
  - **Invalid Arc** – the radius, start point, and end point are not valid.
  - **Invalid Boundary** - the boundaries exterior or interior components are not polygons.
  - **Discontinuous Composite** - the composite does not form a connected path.

- **Uncontained Holes** refers to area features with inner boundaries (holes) that are not contained (totally or partially) within the outer boundary.
- **Unclosed Areas** refers to area features whose boundary does not close. In other words, the first and last point of the boundary do not have the same coordinate values.

![Unclosed Areas Diagram]

The process identifies the original feature geometry. Specifically, the first and last points of the boundary are identified. The anomaly geometry is a Geometry Collection containing two points corresponding to the first and last point of the area.

- **Overlapping Holes** refers to area features whose inner boundaries (holes) overlap one another.

![Overlapping Holes Diagram]

- **Zero-Length Lines** refers to linear features whose coordinates all occupy the same xy location. The process identifies the original feature geometry. The anomaly geometry is a polyline containing the points in the original geometry.

- **Zero-Coverage Areas** refers to area features that have no area (i.e. the vertices are all collinear). The process identifies the original feature geometry. The anomaly geometry is a polyline containing the points in the original geometry.

- **Loop in Area** refers to area features that contain a loop in either its outer or inner boundaries. No measurement is required as any loop is considered an anomaly.

- **Kickback** refers to linear or area features where the geometry doubles back on itself.
The line above is defined by a sequence of 10 vertices (A-B-C-D-D-E-F-G-H). The sequence D-C-D defines the kickback.

A kickback is corrected by removing the vertices that create the kickback. In the above example, the second set of vertices at C and D are removed. The line is defined by the sequence of 8 vertices (A-B-C-D-E-F-G-H). This is a simple case of a Kickback. The process identifies the original feature geometry. Specifically, the vertex or set of two vertices that defined the kickback. The anomaly geometry is either a single point for a duplicate point (corresponds to the location of the duplicate point) or a Geometry Collection containing two vertices that represent the kickback.
Validate Geometry Specialized Anomaly types

- **Kink** (or Spike) refers to linear or area features where there is a sudden divergence of the vertices that define the geometry. A measurement is required to distinguish between kinks and what are considered normal characteristics of the feature.

  ![Kink Example](image)

  In the above line, the vertex sequence A-B-C defines a sudden divergence of the line. This divergence can be a kink.

- **Loop in Line** refers to linear features whose geometry creates a loop whose area is greater than some specified tolerance.

  ![Loop in Line Example](image)

  Unlike Loop in Area, not all loops in line features are anomalies. Only those loops whose size is **less** than some specified parameter are considered anomalies.

  In the above definition, small loops are considered anomalies, while larger ones are considered a normal characteristic. Since the loop is considered an abnormal characteristic, the vertices placed to create the loop can be considered as invalid. Removing these vertices and thus removing the loop can be considered the proper way to correct this anomaly.

- **Short Vector** refers to line or area features with two sequential vertices less than some user-specified tolerance from one another.

  ![Short Vector Example](image)
In the above example, the line has two sequential vertices (A and B) that are less than some specified tolerance from one another. This defines the Short Vector Anomaly. Correcting the Short Vector anomaly requires removal of one of the vertices. Although subtle differences could result depending on which vertex is removed it appears that those differences can be ignored. In the above example, resolution is achieved by deleting point A.

![Diagram of Short Vector Anomaly](image1)

- **Fragmented Geometry** refers to a feature whose geometry is a collection of distinct geometries. The collection could be any combination of area, lines and/or points. The geometries belonging to the collection do not need to be discontiguous for it to be considered a fragmented geometry anomaly.

An automated resolution is available for those feature recordsets with an auto-number key field. If the recordset has a key field or fields other than auto-number, no automatic resolution is available. New records are created for each separate geometry in the collection. Optionally, the original record can be kept and used as one of the new features created. The attribute values for the new records are copied from the original record and a geometry is created that corresponds with one of the distinct elements in the collection. If the original record is kept, its geometry is modified to correspond to one of the geometry components.

- **Duplicate Feature** refers to two features whose attributes match and whose geometries are either identical or have similar size, shape, and location within some specified tolerance. The example below illustrates two line and two area features whose geometries do not match, but which may still be considered duplicates.

![Diagram of Duplicate Feature](image2)

Although the two line and area geometries above do not match exactly, they have similar size, shape, and location, and may be considered duplicates. Determination of whether two features are duplicates can be made by creating zones around the features and analyzing the spatial relationship of the two zones to one another. The picture below
For points to be considered duplicates, one point only need be within the specified buffer distance of another for it to be considered a duplicate (i.e. no calculation of the overlap between the zones is required). The certainty value for a point being a duplicate of another is either 0 (not within buffer distance) or 100 (within buffer distance). Due to the complexity of analyzing Geometry Collections, they will be considered for exact duplicates only at this time. Geometry Collections will not be processed for near duplicates.

The attribute check is performed optionally. Either all attributes must match for a feature to be considered a duplicate, or no attributes are checked, in which case the geometry only is processed. When attributes are checked, all fields with the exception of key fields and geometry fields must have the exact same values.

- **Void Area** refers to a bounded region that does not belong to any area feature. Void Area would be an anomaly for map data that is required to be continuous coverage.

In the example above, two area features are coincident in such a way as to create a bounded region that does not belong to either area feature. This bounded region is an example of a Void Area Anomaly. The unbounded regions of the data under
consideration that do not belong to an area are not detected as a Void Area. The data would have to be modified so that these regions become bounded before they would be identified as a Void Area Anomaly.

The results are output as a query and can be displayed in a map window, data window, and/or queue. Any changes to the geometries of the features for which anomalies are detected will automatically update in any open map or data windows and any dynamic queues.

- **Superfluous Geometry** refers to those instances where the geometry of a feature contains a component which is not needed to portray the feature. Included in Superfluous Geometry are zero-length arcs and composites or collections with only one component.

1. Open Geoworkspace: C:\Fusion\Lessons\Lesson 2 Validating Vector Data \Validating Vector Data.gws.
2. **Add Legend Entries**: BuncombeCounty connection / Road feature class
3. **Toolbox Tab > Geometric Validation > Validate Geometry**.
4. **Input tab** – Select Buncombe County > Road feature class
5. **Anomalies tab** – Check all the **Standard** and **Specialized** entries (expand to view)

![Validate Geometry](image)

6. **Output tab**
7. Select **Output anomalies to queue**
   note: **Queue of Anomalies**
8. Select **Output anomalies as query**

![Image of Validate Geometry window]

9. Click **OK**

10. **Status will show in the lower left**

![Image of processing status]

11. When the processing concludes (it will take a few minutes), the following will open:
   - The Queued Edit Map Window
   - The Queued Edit Data Window
   - The Queued Edit control
12. Scroll through the **Queued Edit** list of Queue of Anomalies and observe that the following anomalies have been detected:
- Kinks
- Duplicate Features
- Fragmented Geometry
- Null Geometry

13. Examine some of them using the Queue Edit toolbar and QE Map Window

*Use the Queued Edit zoom controls, and advance buttons to cycle through the list of anomalies.*

14. Go to Toolbox Tab > Validate Geometry

15. Input tab > BuncombeCounty /Road feature class

16. Anomalies tab > Select:
- Kink
- Duplicate Feature
- Fragmented Geometry
- Null Geometry
17. Then set the **Anomaly properties**: **Auto correct** fields to **Yes** for each of the selected anomalies
18. Click **OK** (it will take a few minutes)
19. The Queued Edit Map Window, Queued Edit Data Window, and *Queue of Anomalies* are now redisplayed with **0 anomalies**
20. Select the original Queue of Anomalies and note that it is also **0**

**Use the Queued Edit toolbar drop down -> Dynamic Queues**

**The auto correct functionality has resolved the geometric anomalies.**

21. Close the Queued Edit Map & Data Windows and exit the Queued Edit command.

**hint: un-anchor and click x to close the Queued Edit toolbar.**
Exercise 2: Connectivity Validation

**Objective:**
Identify anomalies that exist in the spatial relationship *between* features from internal/external sources for evaluation and correction.

**Task 1: Validate Connectivity**

Validate Connectivity *detects anomalies* for all the features in one or more feature classes.

**Undershoots**
Validate Connectivity measures the distance from the endpoint to the intersection of another segment along the original trend of the line. Feature A is being compared against feature B for an undershoot. Validate Connectivity measures the distance along the original trend of the line (shown in green).

If the distance from the endpoint to this intersection is less than the tolerance, an undershoot is reported.

In another example, the endpoint of line A is within the specified distance (undershoot tolerance) of line C. Therefore, the relationship of line A to line C represents an undershoot. Line B is not within the specified tolerance of line C; therefore it is not an undershoot. The undershoot distance is calculated by projecting the line (Line A or Line B) along its original direction until it meets the undershot line (Line C).
Undershoots can be corrected by extending the line to meet the undershot line. (In this case, extending A to meet C). However not all situations are this straightforward, which could require the user to review and edit the features in order to resolve them.

The two features and the location are identified where the undershoot occurs. The endpoint of the undershooting feature along with location on the undershot feature where projection would occur.

**Overshoots**
In the example (below), line A extends past line C a distance less than the specified distance (undershoot tolerance). Therefore, the relationship of line A to line C represents an overshoot. Line B extends past line C a distance greater than this tolerance, and therefore does not represent an overshoot.

Overshoots can be corrected by deleting the part of the line that extends past the overshot line. (In this case, deleting the part of line A that extends past C). However, not all situations are this straightforward, which could require the user to review and edit the features in order to resolve.

An overshoot could also be a case of Unbroken Intersecting Geometry. If this is the case, the overshoot should be corrected, which will in effect correct Unbroken Intersecting Geometry. An overshoot that is also an Unbroken Intersecting Geometry should be identified solely as an Overshoot and corrected as such.

An overshoot could and will in all likelihood be also Non-coincident. However correcting the overshoot will correct the Non-coincident Intersecting Geometry.
In the above example, node B belongs to feature class 1, and is within the tolerance of both node A and node C of feature class 2. Node A and Node C are also within the node mismatch tolerance. Connectivity Validation processes the feature classes together, comparing FC1 to FC2 and FC1. This results in a single anomaly being reported with all three nodes as the mismatch. This allows for the mismatching 3 nodes to be resolved as a group and they are corrected by averaging the coordinates of all three points so that the three lines meet at a common point.

Here is another example of endpoints of lines, intersections, and/or point features that are within a specified distance of one another.

In the above example, three lines terminate within a specified distance of one another. Node mismatches can be corrected in a couple of different ways. First, by averaging the coordinate values of all the nodes in question and moving the offending nodes to that location.
If one feature takes precedence over the others, then the endpoints of the other features can be moved to the endpoint of higher precedence. This may require the operator to make a choice. In our example, nodes B and C are moved to A, which is the node that is of higher priority.

An automatic option is available to correct the simpler cases. Only cases where the anomalous features are the endpoints of line or point nodes are corrected. If one of the nodes is defined by an intersection, no automatic resolution is possible.

Also, no secondary mismatches can exist. In other words, for a group of features that mismatch, each feature in that group must meet the mismatch criteria with every other feature in that group and must not mismatch with features outside the group. When this occurs, the node mismatch can be corrected by averaging the coordinates of all nodes and moving the nodes to this new location.

**Unbroken Intersecting Geometry**

Unbroken Intersecting Geometry refers to a linear feature that crosses over another linear feature or area boundary instead of being broken into separate features at the intersection point.
Unbroken Intersecting Geometry can be corrected by splitting the features that cross over into multiple features. In the above example, Line 1 and Line 2 are both split into two lines. The intersection point serves as the point where the two lines are split.

Non-coincident Intersecting Geometry
When two lines cross and both are non-coincident Connectivity Validation will report this as a single queue item identifying that both features need to have a vertex added at the intersection. Non-coincident Intersecting Geometry refers to the situation where a line or areas cross over one another without corresponding vertices at the intersection points.

Non-coincident Intersecting Geometry can be corrected by inserting vertices on the features at the location of the intersection.
Non-coincident intersecting geometry anomalies not only occur at the point of intersection between two features, but also occur when features share a linear segment. If points on the shared segment don’t belong to all features that share that segment, these points are considered anomalies.

Warning! Validate Geometry must be performed first and any issues resolved prior to running Validate Connectivity.

1. Select Toolbox > Geometric Validation > Validate Connectivity.
2. Select Road as the input feature class.
3. Select the **Anomalies** tab and check all of the **Standard Anomaly types** as shown.

![Anomalies tab](image)

4. Check the **Standard Anomaly** types and change the **Overshoot**, **Undershoot**, and **Node Mismatch Tolerance** values to 20.

![Standard Anomaly types](image)

5. Click the **OK** button on the **Advanced Validate Connectivity** dialog to begin processing.

6. Note that 470 anomalies are detected and queued in the **AnomaliesQueue** queue, and the anomalies are also loaded in the activated **Queued Edit Map Window** and **Queued Edit Data Window**.
7. Select **Legend > Add Legend Entries** and add the *Road* feature class to the *Queued Edit* Map Window. The *Road* feature class will now be displayed with the anomaly in the *Map Window*.

The symbology of **Road** feature class and the AnomalyGeometry of **Anomalies Queue** item may have to be changed so the two entries are legible in the *Queued Edit Map Window*. 
8. Scroll through the *AnomaliesQueue* queue and review the various connectivity anomalies, such as Overshoots, Undershoots, and Unbroken Intersections.

9. The Queued Edit Options icon > Queuing Options > General tab > Show Description box may have to be selected so the Anomaly description can be reviewed.
10. Select Toolbox > Geometric Validation > Validate Connectivity, select the Road feature class, select the Anomalies tab, and change all of the 5 Standard Anomaly type Auto correct entries to Yes.

11. Click OK on the Advanced Validate Connectivity dialog to begin processing the Road features with the Auto correct entry set to Yes.

12. Note that 5 unbroken intersections and 4 node mismatches remain. This is because Road cul-de-sacs connected to an endpoint are considered to be an unbroken intersection. The node mismatches can’t be resolved because there are 3 or more nodes detected and Validate Connectivity doesn’t know which items to resolve.
Exercise 3: Automated Validation

**Objective:**
Setup and execute an automated validation.

**Task 1: Run an Automated Connectivity Validation**

Automated Validation allows you to create, edit, save, and execute advanced connectivity rules and geometry rules. After you create the rules, you save them as a group to a validation rule set XML file. You then execute the rule set file to validate features. The processing results for each rule in the rule set are stored in a separate queue. Connectivity results are stored in static queues, and geometry results are stored in dynamic queues.

1. Select **Toolbox > Geometric Validation > Automated Validation**.

2. The **Automated Validation** dialog will appear.

3. Select the **New Connectivity...** button on the **Automated Validation** dialog.

4. The **New Connectivity Rule** dialog will appear.
5. Type HydroLine in the **New Connectivity Rule >Name:** field.

6. Click the **Properties…** button in the **Query Names files** field.

7. Select the *Buncombe County* connection HydroLine feature class in the **Query Names File Properties** dialog.

8. Click the **OK** button on the **Query Names File Properties** dialog.
9. The **Save Query Names File As** dialog will appear.

![Image of the Save Query Names File As dialog]

10. Navigate to the `C:\Training\Buncombe` folder and type in *QueryNames.xml*.

11. Click the **Properties…** button next to the **Anomaly Definitions file:** field.

12. The **Connectivity Anomaly Definition File Properties** dialog will appear.

![Image of the Connectivity Anomaly Definition File Properties dialog]

13. Click the **Load…** button on the bottom left of the **Connectivity Anomaly Definition File Properties** dialog.

14. Navigate to the `C:\<Training path>\Fusion\Lessons\Lesson 2 Validating Vector Data` folder and select the file of *AnomalyDefinitions.xml*. 

![Image of the Connectivity Anomaly Definition File Properties dialog]
15. Select the BuncombeCounty connection in the Output anomalies to static queue frame.

16. Type HydroLineQueue in the QueueName field.

17. Click the OK button on the New Connectivity Rule dialog.

18. Check the box next to the HydroLine entry in the Rules to apply: frame.
19. Click the **Apply** button on the bottom right of the **Automated Validation** dialog.

20. The **Save Rule Set** dialog will appear.

21. Save the name to **C:\Training\Buncombe\QueryNames2.xml**.

22. The **Validation.log file** is generated with processing information:

23. The **Automated Validation > Status** field will have a value of **Successful**.

24. Close the workspace when done.