Java Conflation Suite

# User Guide

Prepared by:

VIVID SOLUTIONS

# Document Change Control

| REVISION NUMBER | DATE OF ISSUE | AUTHOR(S) | BRIEF DESCRIPTION OF CHANGE |
|---|---|---|---|
| 1.0 | 15-Jan-2003 | Martin Davis | **Initial draft** |
| 1.1 | 15-Jan-2003 | Jon Aquino | **Formatting** |

# Table of Contents

# 1. OVERVIEW

This document explains how to use the JCS Conflation Suite and the JUMP Workbench to clean and conflate datasets.

## 1.1 RELATED DOCUMENTS

Related information is contained in the following documents:

| | |
|---|---|
| ***JCS Project Report*** | Details the design approach and algorithms investigated during the Java Conflation Suite project |
| ***JCS Developer Guide*** | Provides detailed information on using JCS conflation components programmatically |
| ***JUMP User Guide*** | A guide to using the features of JUMP |
| ***JUMP Developer Guide*** | A guide to using the JUMP API and to extending the JUMP Workbench through its plug-in framework |

# 2. FEATURE-LEVEL QUALITY ASSURANCE

Most spatial formats support representing many different kinds of geometrys (e.g. polygons with holes, multipolygons, linestrings which self-intersect). While working with spatial datasets, a couple of problematic situations can arise:

- A dataset contains geometric objects which are ill-formed (e.g. polygons which self-intersect)
- A dataset contains geometric objects which are valid, but which the user does not wish to have in the dataset (e.g. a dataset should not contain multipolygons or linestrings that self-intersect).

JUMP provides a facility for detecting these situations by validating the geometry of features in a dataset according to rules that the user can specify. It can check:

- basic geometric topology
- that line segments are longer than a specified amount
- that polygon areas are greater than a specified amount
- that angles in geometry linework are larger than a specified amount
- that the dataset contains only specified geometry types
- that polygons do not contain holes

For more details, see *Validating A Layer* in the JUMP User's Guide.

When invalid geometrys are detected, the user must decide how they wish to fix the problem. JUMP provides editing tools that allow easy correction of geometry linework.

# 3. CLEANING COVERAGES

## 3.1 COVERAGES AND COVERAGE ERRORS

A common kind of spatial dataset is a polygonal coverage. Coverages are datasets which contains polygons which must not overlap one another. In some cases the polygons are intended to completely cover a given area. In other cases, space or holes are allowed in the coverage to represent real-world entities (such as streets between blocks) (see Figure 1 below). In this case, however, the spaces are usually known to be larger than a certain amount.

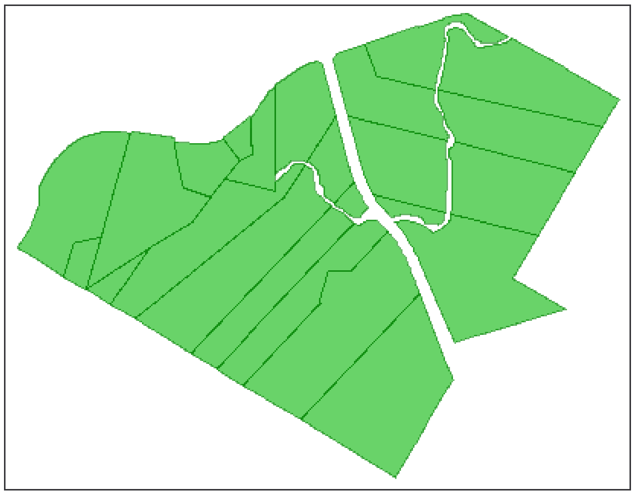


**Figure 1 - A coverage containing spaces between areas covered by polygons**

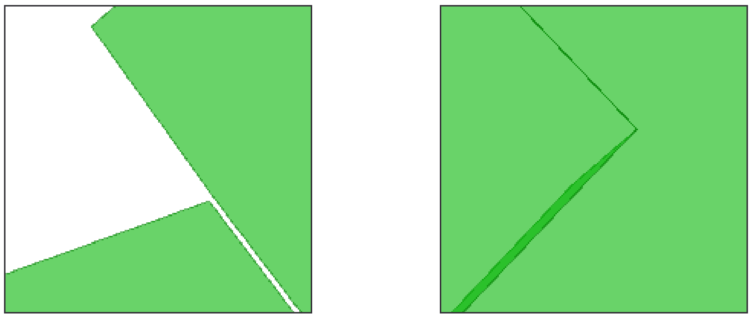
While a dataset may be intended to have coverage topology, not all spatial data formats and tools can enforce this constraint. It can happen that after edits or other processing is carried out on a dataset it will be left containing errors which violate the rules of coverage topology.

Often these errors are very small and are undetectable by visual inspection. Many GIS tools are not able to detect or display these problems. However, these gaps and overlaps are errors and should be removed from the dataset. JCS provides tools to detect, display, and remove gaps and overlaps.

Two different kinds of errors can occur in coverage datasets. These are:

- **Gaps** are places where two adjacent polygons are separated by a small amount along some or all of their boundary (see Figure 2a below). Since some coverages can contain legitimate spaces between polygons, gaps are always defined relative to a specified distance tolerance. Only spaces which have the adjacent line segments separated by less than the distance tolerance are considered to be gaps.

- **Overlaps** are places where two polygons overlap (see Figure 2b below). No distance tolerance is needed, since **any** overlap is always considered to be an error in a coverage.


(a) (b)
**Figure 2 – Examples of (a) a gap (b) an overlap**

## 3.2 COVERAGE CLEANING TOOLS

JCS provides tools to do the following tasks involved in coverage cleaning:

- QA coverages to detect errors (gaps and overlaps)
- Automatically fix most or all errors in a coverage
- Allow manual fixing of errors not fixed automatically

## 3.3 DATA PREPARATION

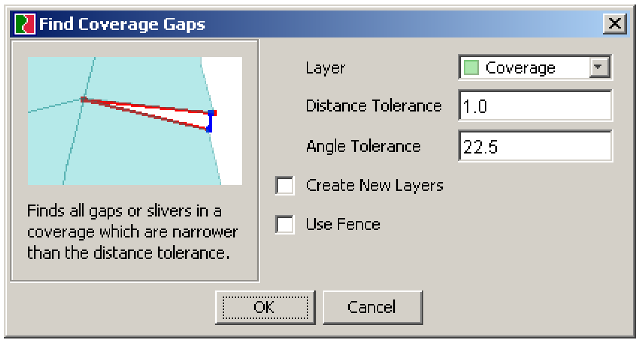
- Load the coverage dataset into the JUMP Workbench. The examples will use a layer loaded in the Reference category, but in fact any category will do. (See the *JUMP User's Guide* for more information on loading data)

## 3.4 FIXING GAPS

The first step in cleaning a coverage is to detect and remove gaps. Many overlaps are caused by gaps, so this will fix many or all of the overlaps as well.

### 3.4.1 Gap Detection

- Invoke the Gap Detection function by choosing **QA / Find Coverage Gaps...**
- In the **Find Coverage Gap** dialog (see Figure 3 below) select the layer to work on, and choose an appropriate **Distance Tolerance**. (A good idea is to choose a small tolerance to begin with and then make the tolerance larger in subsequent runs to find larger errors). The **Angle Tolerance** is less critical – the default value of 22.5 degrees is almost always sufficient.



Finds all gaps or slivers in a coverage which are narrower than the distance tolerance.

Layer: Coverage
Distance Tolerance: 1.0
Angle Tolerance: 22.5
Create New Layers
Use Fence

**Figure 3 - Find Coverage Gaps dialog**

- Run the Find Coverage Gap function by clicking OK
- When the Find Coverage Gap function executes it will create two new layers in the QA category (see Figure 4 and Figure 5 below):

| **Layer** | **Description** |
|---|---|
| **Gap Size** | Shows the segments on either side of detected gaps |
| **Gap Segments** | Contains indicators showing the size of gaps |

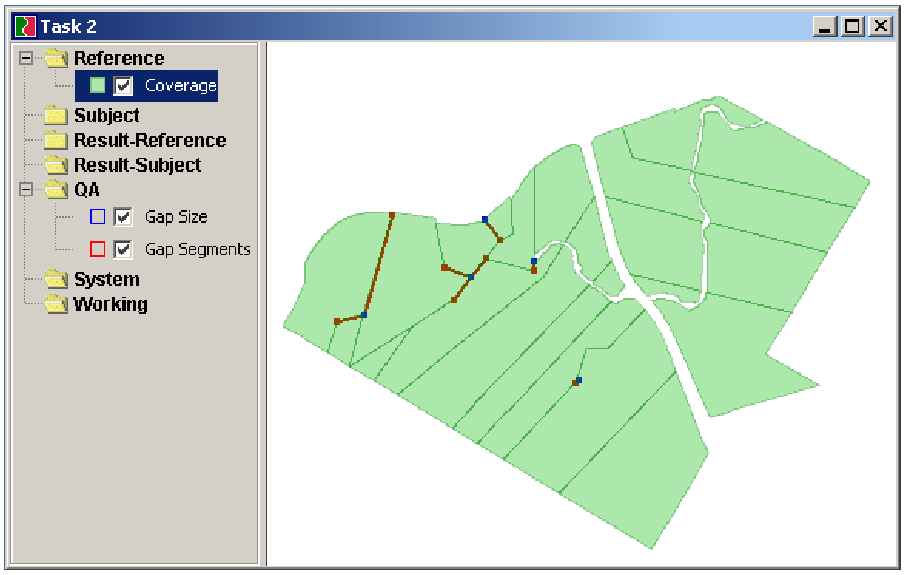


**Figure 4 - QA layers created by the Find Coverage Gap function**

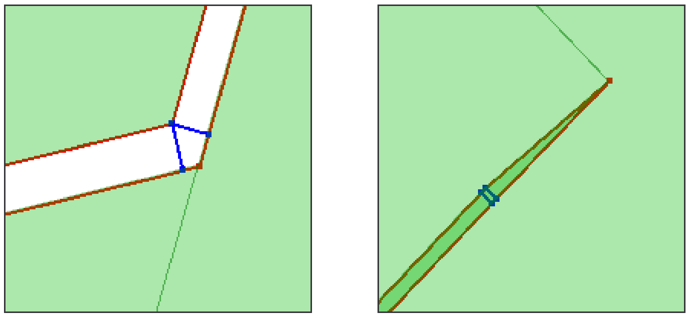
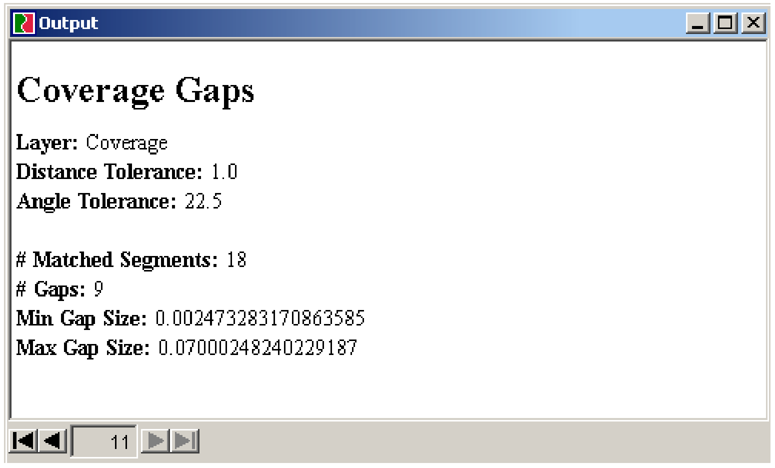


**Figure 5 - Closeups of gaps showingGap Segment indicators (red) and Gap Size indicators (blue)**

- View the report summarizing the results by clicking on the **Output Window** tool ( ).



Coverage Gaps

**Layer:** Coverage
**Distance Tolerance:** 1.0
**Angle Tolerance:** 22.5

**# Matched Segments:** 18
**# Gaps:** 9
**Min Gap Size:** 0.002473283170863585
**Max Gap Size:** 0.07000248240229187

**Figure 6 - Output Report for the Find Coverage Gap function**

- To see the gaps detected, either:
  - Use the **Zoom** tool ( ) and **Pan** tool ( ) to zoom to visible Gap Size indicators
  - View the Gap Size features in the Attribute View, select particular features and zoom directly to them using the Attribute View Zoom tool. You can sort by the size of the gaps to view larger ones first.
  - Sometime it is useful to see the precise values of the vertices forming the gap (for example, if two vertices are so close together they cannot be visually separated). To do this use the Show Vertices in Fence function. Use the **Fence** tool to draw a fence around the vertices, and then right-click and choose **Vertices In Fence**. A window will appear that shows the precise values of all vertices in the fence.

| **Hint** | It can be helpful to increase the transparency of the data layer in order to make the Gap Size and Gap Segments stand out more clearly. Use **Change Styles... / Transparency** to do this. |
|---|---|

| | |
|---|---|
| **Hint** | In order to easily see the size of gaps, use the **View / Scale Bar** function to display a scale bar on the screen. |

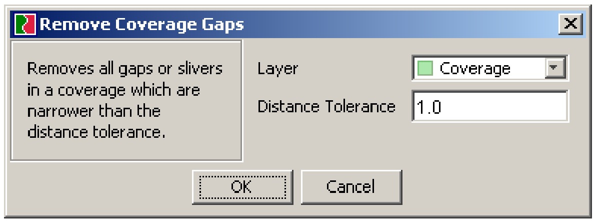
| | |
|---|---|
| **Hint** | Sometimes errors are so small that JCS is unable to display the geometry correctly. In this case simply zoom out until the surrounding geometries are correctly displayed. (Even if you can't view errors in the View window, JCS can still detect and correct them.) |

Find Coverage Gaps may be run as many times as required with the same or different Distance Tolerances.

### 3.4.2 Automated Gap Correction

To fix most or all coverage gaps automatically, perform the following steps:

- Invoke the Remove Coverage Gaps function by choosing **Clean / Remove Coverage Gaps…** (see Figure 7 below).


Remove Coverage Gaps

Removes all gaps or slivers in a coverage which are narrower than the distance tolerance.

Layer: Coverage
Distance Tolerance: 1.0

**Figure 7 - Remove Coverage Gaps dialog**

- In the **Remove Coverage Gaps** dialog, select the layer to work on, and enter an appropriate **Distance Tolerance** (usually the same one used during gap detection).
- Run the Remove Coverage Gap function by clicking OK

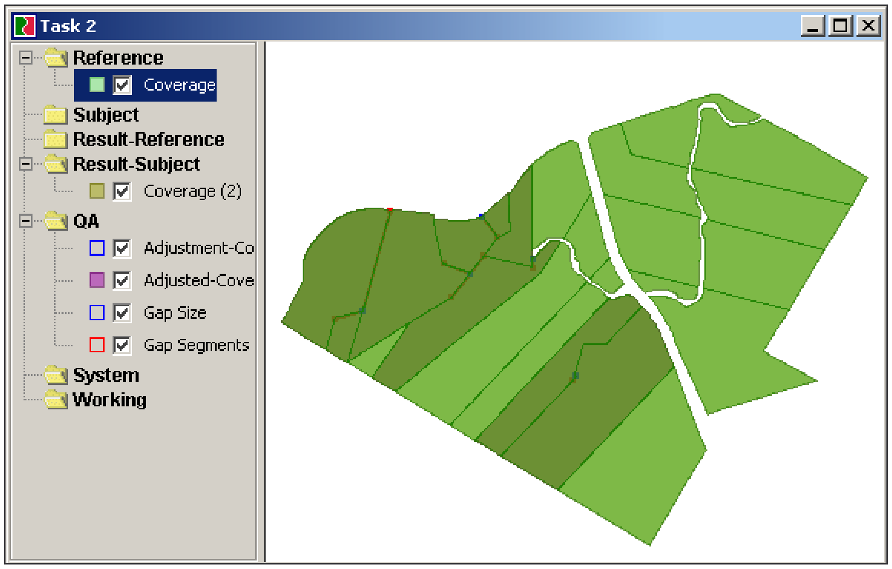


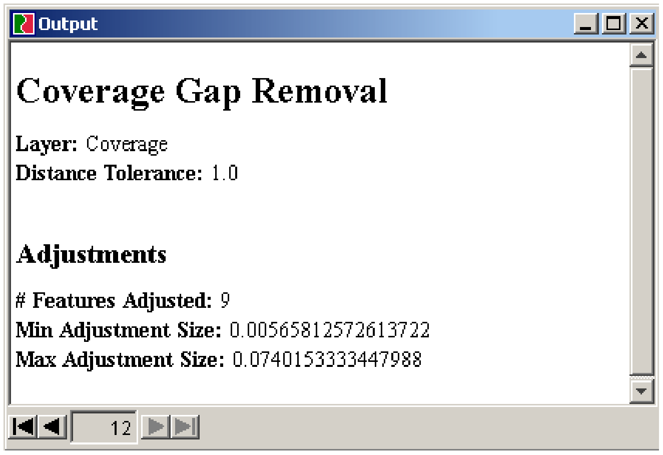
**Figure 8 - Layers created by the Remove Coverage Gaps function**

- When the Remove Coverage Gap function executes it will create a new, modified version of the input data layer as well as some layers containing features which allow you to QA the results of the process if desired (see Figure 8 above). The layers are created under two different categories. In the following table, ***dataset*** is the name of the input layer.

| Layer | Description |
|---|---|
| **Result-Subject / *dataset* (2)** | New version of the input dataset with gaps removed |
| **QA / Adjustment-*dataset*** | Size indicators for the adjustments performed |
| **QA / Adjusted-*dataset*** | Contains copies of the features that were adjusted |

  The Adjustment and Adjusted layers allow you to see what changes were made to your input dataset.

- View the report summarizing the results by clicking on the **Output Window** tool (🗔).



Coverage Gap Removal

**Layer:** Coverage
**Distance Tolerance:** 1.0

Adjustments

**# Features Adjusted:** 9
**Min Adjustment Size:** 0.00565812572613722
**Max Adjustment Size:** 0.0740153333447988

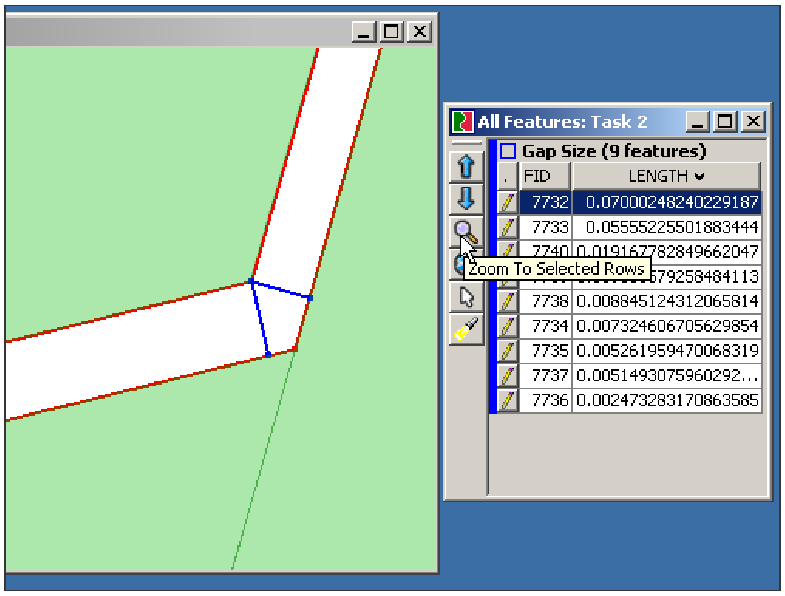
**Figure 9 - Output Report from Remove Coverage Gaps function**

The automated gap removal function can usually fix more than 95% of gaps in the coverage. However, there are some situations where it is difficult to automatically determine how to fix a gap. These situations will not be fixed by the Coverage Gap Removal function. In order to determine what (if any) gaps are still present in the coverage, re-run the Find Coverage Gap procedure on the new layer with the same Distance Tolerance (see *3.4.1 Gap Detection*). Gaps that remain will have to be fixed manually as described in the following section.

### 3.4.3 Manual Gap Correction

Any gaps that cannot be fixed automatically can be fixed using JUMP Workbench editing tools such as **Snap Vertices** () and **Delete Vertex** ().

- Put the new layer into edit mode by right-clicking on it and choosing **Editable**
- Use the **Zoom** tool () or the **Attribute View Zoom** tool as described above to zoom to coverage gaps (see Figure 10 below).



| . | FID | LENGTH |
|---|---|---|
| | 7732 | 0.07000248240229187 |
| | 7733 | 0.05555225501883444 |
| | 7740 | 0.019167782849662047 |
| | [illegible] | [illegible]679258484113 |
| | 7738 | 0.008845124312065814 |
| | 7734 | 0.007324606705629854 |
| | 7735 | 0.005261959470068319 |
| | 7737 | 0.0051493075960292... |
| | 7736 | 0.002473283170863585 |

**Figure 10 - Zooming to a gap using the Attribute View Zoom tool**

- Most gaps are caused by vertices that are not aligned together. To align vertices together use the **Snap Vertices** tool or the **Snap Vertices To Selected Vertex** tool. (See below for information on how to use these tools.)

- To verify that a gap has been fixed, draw a fence around the gap size indicator. (If the Snap Vertices tool was used to fix the gap the fence will already exist). Choose **Clean / Update Gap In Fence**. This re-runs the Find Coverage Gap algorithm on the gap(s) in the fence, and updates the **Gap Size** and **Gap Segments** layers. If the gap has been fixed, the indicators will disappear. If you are using the Attribute View to step through the gaps the gap will be removed from the table. If the gap does not disappear you may need to further adjust the geometries around the gap to eliminate it.

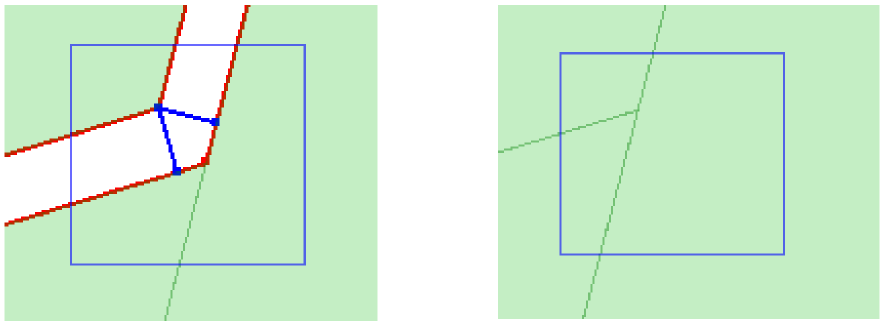


**Figure 11 - Using the Update Gap in Fence tool to verify that a gap has been fixed**

- Some gaps can be caused by holes that are too close to the shell of a polygon. To fix this, either delete the hole or move its vertices further away from the shell.

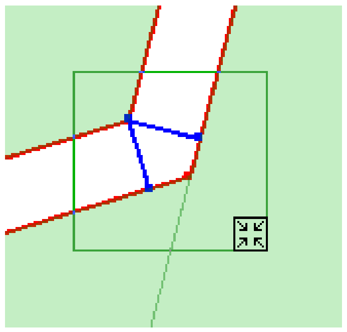
### 3.4.4 Using the Snap Vertices Tool

The **Snap Vertices** tool is the fastest way to snap vertices together. It automatically chooses the centre-most vertex to snap to (based on the position of the box drawn by the user). If it is desired to explicitly choose the vertex to snap to, the **Snap Vertices to Selected Vertex** tool can be used.

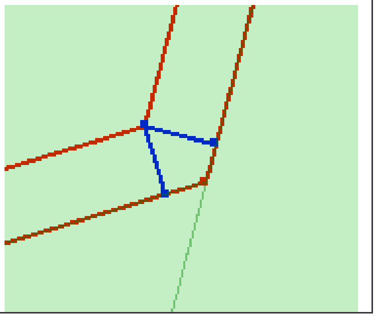
- Before starting to snap vertices, ensure that the layer is in Edit mode so that it is modifiable. The layer can be made editable by right-clicking on it in the layer list and choosing **Editable.** This will cause the Edit toolbox to appear.'

- Click on the **Snap Vertices** tool to enable it.

Draw a box around the vertices with the **Snap Vertices** tool



The vertices will snap together

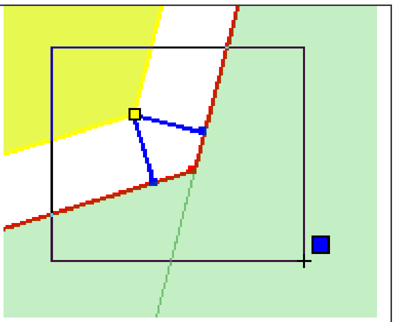
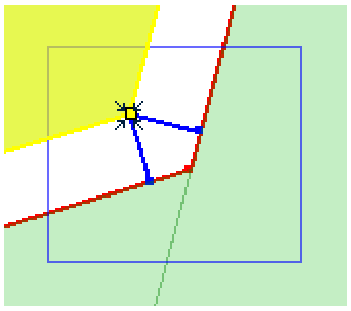
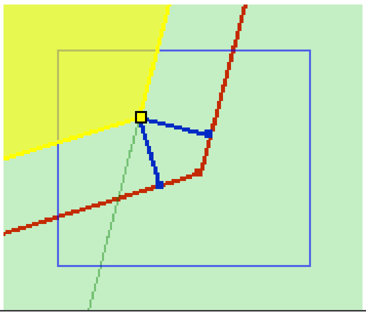


### 3.4.5 Using the Snap Vertices To Selected Vertex Tool

The **Snap Vertices To Selected Vertex** tool is the most precise way to snap vertices together. It allows choosing the exact vertex which is to be snapped to.

- Before starting to snap vertices, ensure that the layer is in Edit mode so that it is modifiable. The layer can be made editable by right-clicking on it in the layer list and choosing **Editable.** This will cause the Edit toolbox to appear.'

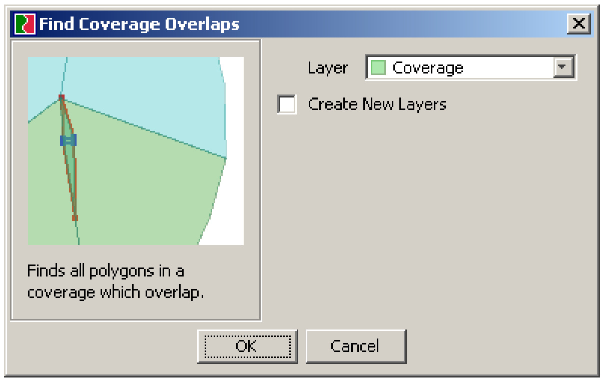
- Click on the **Snap Vertices To Selected Vertex** tool to enable it.

| | |
|---|---|
| Select the feature containing the vertex to be snapped to. Draw a Fence around the vertices to be snapped |  |
| Shift-click to choose the vertex to snap to. |  |
| The vertices in the fence are snapped together to close the gap |  |

## 3.5 FIXING OVERLAPS

### 3.5.1 Overlap Detection

- Invoke the Overlap Detection function by choosing **QA / Find Coverage Overlaps...** (see Figure 12 below).



Layer: Coverage
Create New Layers
Finds all polygons in a coverage which overlap.

**Figure 12 - Find Coverage Overlaps dialog**

- In the **Find Coverage Overlaps** dialog select the layer to work on. No Distance Tolerance is needed for this function, since it finds *all* overlaps regardless of size.
- Run the Find Coverage Overlap function by clicking OK
- When the Find Coverage Overlap function executes it will create three new layers in the QA category (see Figure 13 below):

| Layer | Description |
|---|---|
| **Overlap Size** | Contains indicators showing the size of overlaps |
| **Overlap Seg** | Shows the segments forming overlaps |
| **Overlaps** | Contains copies of the features which overlap |

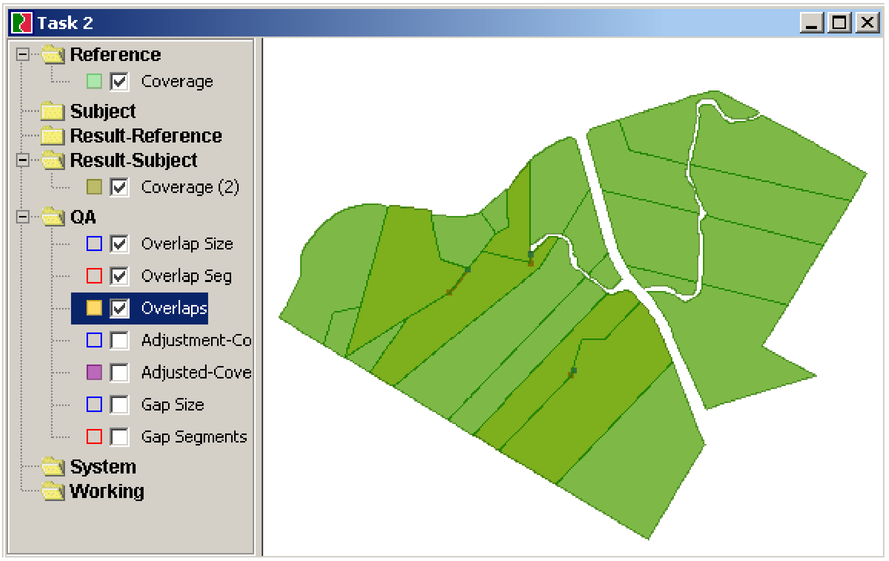
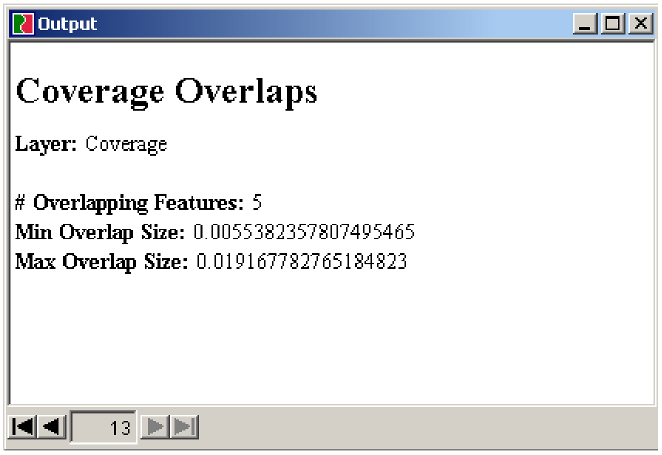


**Figure 13 - Layers created by the Find Coverage Overlaps function**

- View the report summarizing the results by clicking on the **Output Window** tool (🗔).



Coverage Overlaps

Layer: Coverage

# Overlapping Features: 5
Min Overlap Size: 0.0055382357807495465
Max Overlap Size: 0.019167782765184823

**Figure 14 - Output Report for the Find Coverage Overlaps function**

- To see the overlaps detected, either:
    - Use the **Zoom** tool () and **Pan** tool () to zoom to visible Overlap Size indicators
    - View the Overlap Size features in the Attribute View, select particular features and zoom directly to them using the Attribute View Zoom tool. You can sort by the size of the gaps to view larger gaps first.

**Note:** Sometimes the errors are so small that JCS is unable to display the geometry correctly. In this case simply zoom out until the surrounding geometries are displayed.

### 3.5.2 Manual Overlap Correction

Most small overlaps should have been detected and fixed using the Gap correction tools previously discussed (see *3.4.3 Manual Gap Correction*). Overlaps that remain may be due to two things:

- The **Distance Tolerance** used in gap cleaning may have been too small to correct the gap due to the overlap. If possible, increase the Distance Tolerance value to allow the gap to be corrected. However, do not increase the Distance Tolerance to the point at which valid spaces between polygons begin to be affected.
- Overlaps may be due to duplicate features in the datasets. In this case, it may be possible to simply remove one of the features. (The best way to do this is to use the **Feature Info** tool to view all features at a specified point. One of the features can be selected in the **Feature Info** view and deleted using **Cut Selected Items**)
- Overlaps may be present in the input dataset for a business reason. In this case it is necessary to decide how to deal with the overlap manually.

### 3.6 SAVING THE CORRECTED DATA

- When the data has been cleaned satisfactorily, save the layer to a dataset by right-clicking on the layer and choosing **Save Layer As…** (see the JCS User's Guide).

# 4. ALIGNING COVERAGES ALONG THEIR BOUNDARY

In the same way that coverages can contain gaps and overlaps internally, coverages that represent adjacent areas can also contain gaps and overlaps along the shared boundary. JCS provides tools to detect and automatically fix these kinds of errors. The process of fixing boundary errors is known as Boundary Alignment.

# 5. FINDING GEOMETRY DIFFERENCES

A common problem in spatial data processing is to find differences in the geometries in two datasets. Geometric differences can be caused by:

- Features being added or deleted
- Geometry linework being altered
- Data format conversion or other applications altering the values of coordinates slightly
- The precision being reduced by conversion to and from other formats

A common situation is that one dataset is an updated version of the other, and it is required to determine the location and number of changes made to the original dataset.

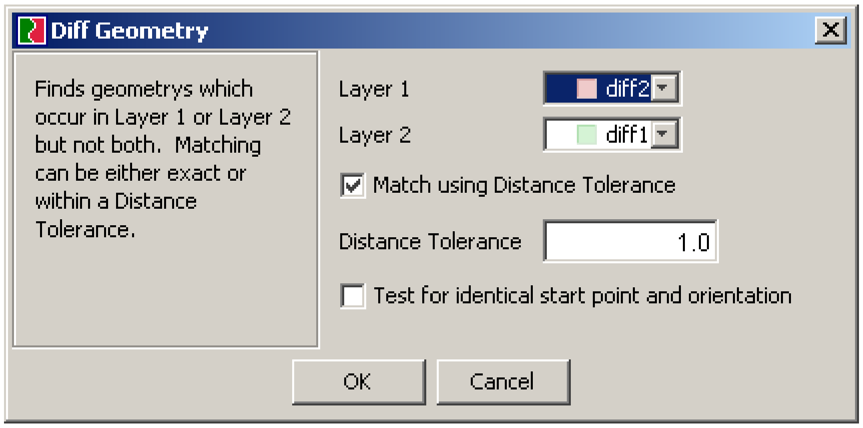
These changes can be detected by means of a spatial difference algorithm. Difference algorithms work by matching geometries between the two datasets, and reporting the geometries in each dataset which do not have a match in the other dataset. Matching can be carried out in two different ways:

- **Exact Matching** tests if geometries are the same on a point-by-point basis. This is useful if the datasets are known to be very similar and only small changes have occurred.
- **Matching within a Distance Tolerance** tests if each point in the geometries lies within a given distance of some other point on the other geometry. This has the effect of ignoring changes smaller than the distance tolerance. This is useful in situations where many points have been shifted by small amounts (e.g. by a change in precision caused by format conversion) and only large-scale differences are of interest.

## 5.1 DIFF GEOMETRY

JCS provides a function called Diff Geometry to detect the differences between the geometry of features in two datasets. (The term "Diff" is a reference to a well-known Unix application for detecting the differences between two files. The JCS Diff Geometry function performs a similar function for spatial datasets.)

- Load the two datasets into two layers in the JUMP GUI
- Invoke the **QA / Diff Geometry...** function
- In the **Diff Geometry** dialog (see Figure 15) select the layers to process.
- If desired, a **Distance Tolerance** can be used. Using a distance tolerance will match Geometrys if all points on the Geometrys are within the distance tolerance of each other. This option is useful when only differences greater than a certain size are of interest.
- The option **Test for identical start point and orientation** applies only if a Distance Tolerance is not used. It indicates that the lists of points in Geometrys must be completely identical for the Geometrys to match. If this option is not selected (the usual case) the lists must contain the same points in the same order, but not necessarily starting at the same point or with the same orientation (in the case of rings).


Diff Geometry

Finds geometrys which occur in Layer 1 or Layer 2 but not both. Matching can be either exact or within a Distance Tolerance.

Layer 1: diff2
Layer 2: diff1
[✓] Match using Distance Tolerance
Distance Tolerance: 1.0
[ ] Test for identical start point and orientation

**Figure 15 – Diff Geometry dialog**

- Run the Diff Geometry function by clicking OK
- When the Diff Geometry function executes it creates four new layers in the QA category. These layers can be used to inspect the differences between the two datasets. The first pair of layers show all the unmatched Geometries in the input datasets. The second pair of layers show the segments in the unmatched Geometries which cause the Geometries to be different.

| **Layer** | **Description** |
|---|---|
| **QA / Geometry Diffs – *dataset_1*** | Geometrys in dataset_1 which have no match in dataset_2 |
| **QA / Geometry Diffs – *dataset_2*** | Geometrys in dataset_2 which have no match in dataset_1 |
| **QA / Segment Diffs – *dataset_1*** | Segments in the unmatched geometries in dataset_1 which have no match in dataset_2 |
| **QA / Segment Diffs – *dataset_2*** | Segments in the unmatched geometries in dataset_2 which have no match in dataset_1 |

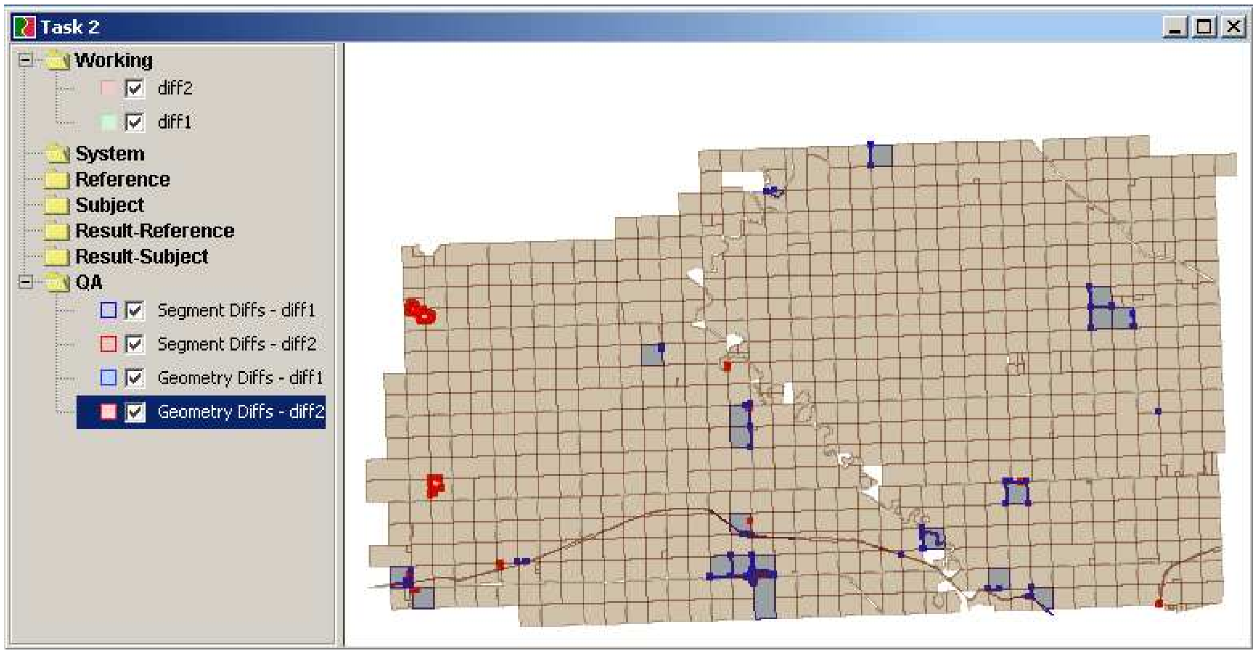
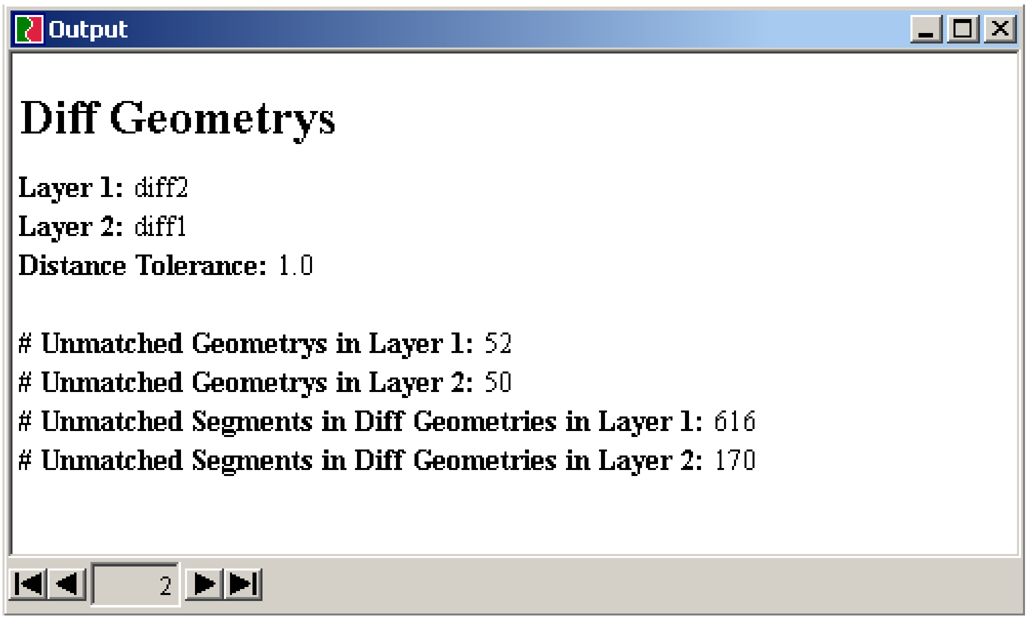


**Figure 16 - Layers produced by Diff Geometry**

- View the report summarizing the results by clicking on the **Output Window** tool (□).


Output

Diff Geometrys

**Layer 1:** diff2
**Layer 2:** diff1
**Distance Tolerance:** 1.0

**# Unmatched Geometrys in Layer 1:** 52
**# Unmatched Geometrys in Layer 2:** 50
**# Unmatched Segments in Diff Geometries in Layer 1:** 616
**# Unmatched Segments in Diff Geometries in Layer 2:** 170

2

**Figure 17 - Output Report for the Diff Geometry function**

# 6. FINDING OFFSET BOUNDARY CORNERS

A problem that sometimes occurs in adjacent datasets is that the corners of adjacent polygons are incorrectly offset by a small amount. An area where this is of particular concern is in cadastral datasets, where parcels from different jurisdictions may not align exactly.

It is important to note that this problem is not the same as Boundary Alignment. There is not necessarily any misalignment between the vertices of an offset corner. The datasets may form a valid coverage, correctly noded with no gaps or overlaps, and yet they may still contain offset corners. The concept of an offset corner is at the macro level of the arrangement of the polygon edges, not at the micro level of misaligned vertices.

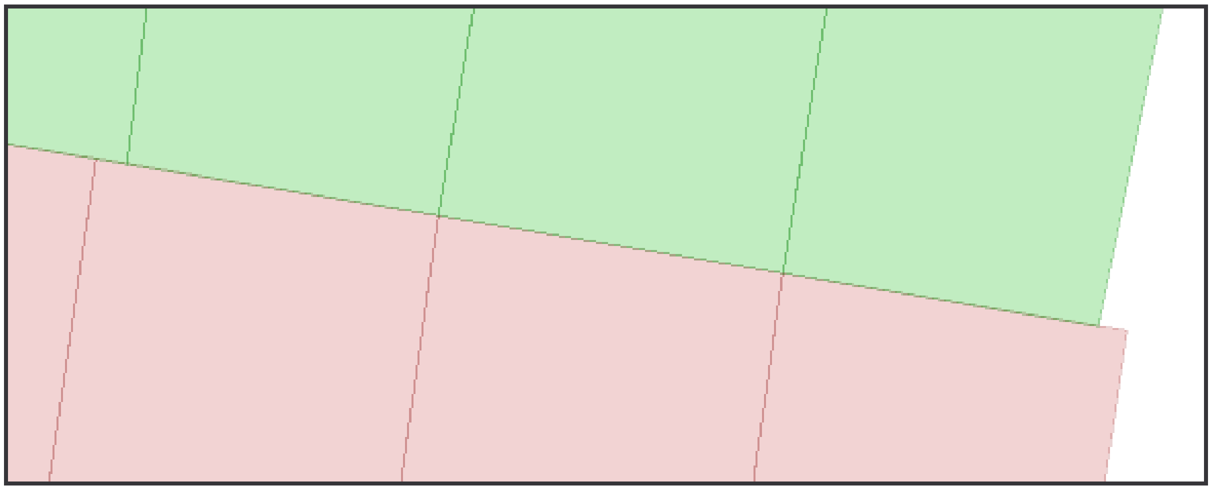


**Figure 18 - Examples of Offset Boundary Corners**

To identify offset problems JCS provides the **Offset Boundary Corner Finder** function. It allows the user to define a tolerance which specifies the boundary between datasets, an angle which defines which vertices will be considered as corners, and a tolerance for the size of offsets to report. The function produces a layer containing indicators allowing easy visualization of the location and size of offsets.

If it is desired to fix the offset corners, this is easily accomplished using the **Snap Vertices** and **Snap Vertices to Selected Vertex** tools.

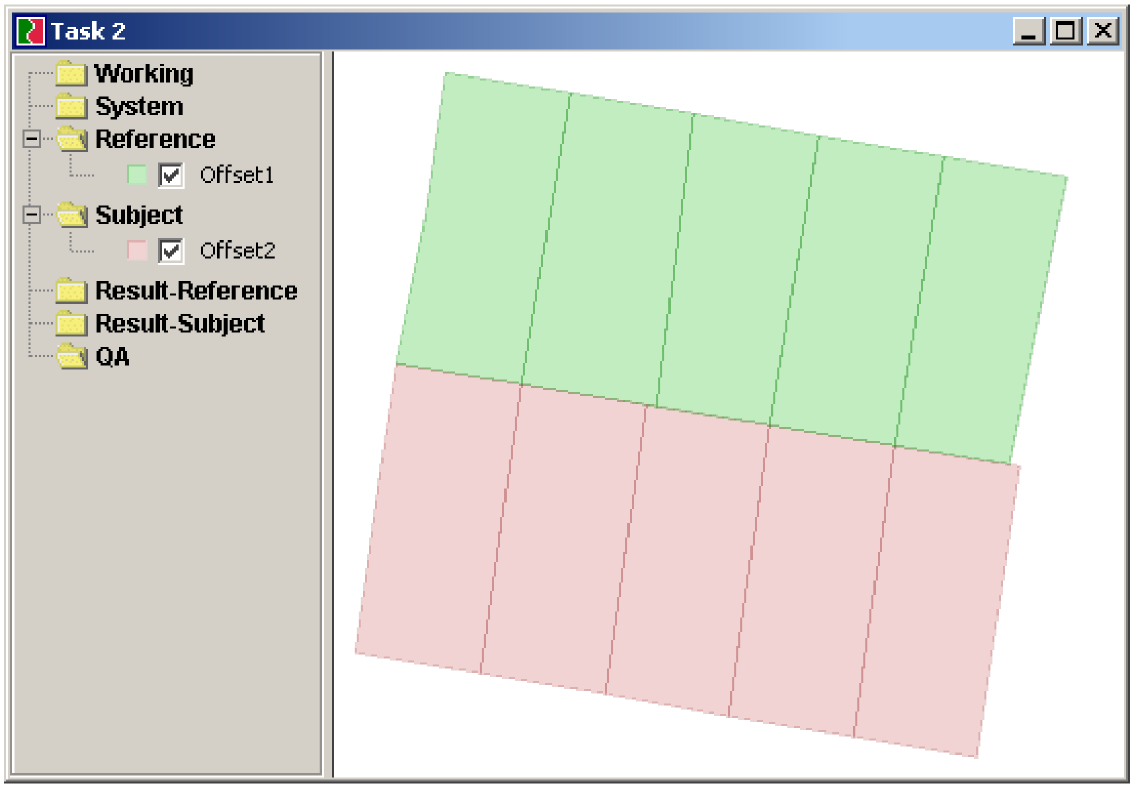
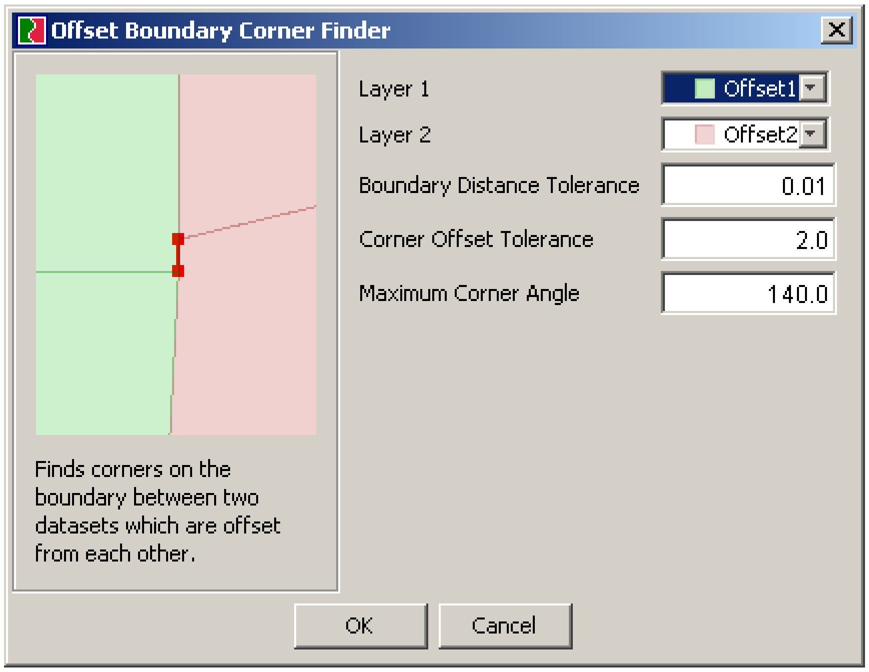


**Figure 19 - Input layers for the Offset Boundary Corner Finder**

- Load the two datasets into two layers in the JUMP GUI
- Invoke the **QA / Find Offset Boundary Corners...** function
- In the **Offset Boundary Corner Finder** dialog select the layers to process.
- Specify a **Boundary Distance Tolerance**. This defines which vertices are considered to be on the boundaries of the datasets by their proximity to the other dataset. Only boundary vertices will be tested for being corners and offsets.
- Specify a **Corner Offset Tolerance**. Only offsets below this length will be reported.
- If desired, specify the **Maximum Corner Angle** (in degrees). Vertices whose neighbour segments meet at this angle or less will be considered to form corners. This angle should always be less than 180 degrees.



Layer 1: Offset1
Layer 2: Offset2
Boundary Distance Tolerance: 0.01
Corner Offset Tolerance: 2.0
Maximum Corner Angle: 140.0

Finds corners on the boundary between two datasets which are offset from each other.

**Figure 20 - Offset Boundary Corner dialog**

- Run the Offset Boundary Corner Finder function by clicking OK
- When the Offset Boundary Corner Finder function executes it creates a layers containing indicators which show the location and size of corner offsets.

| **Layer** | **Description** |
|---|---|
| **QA / Offset Indicators** | Indicators which show the location and size of any corner offsets found |

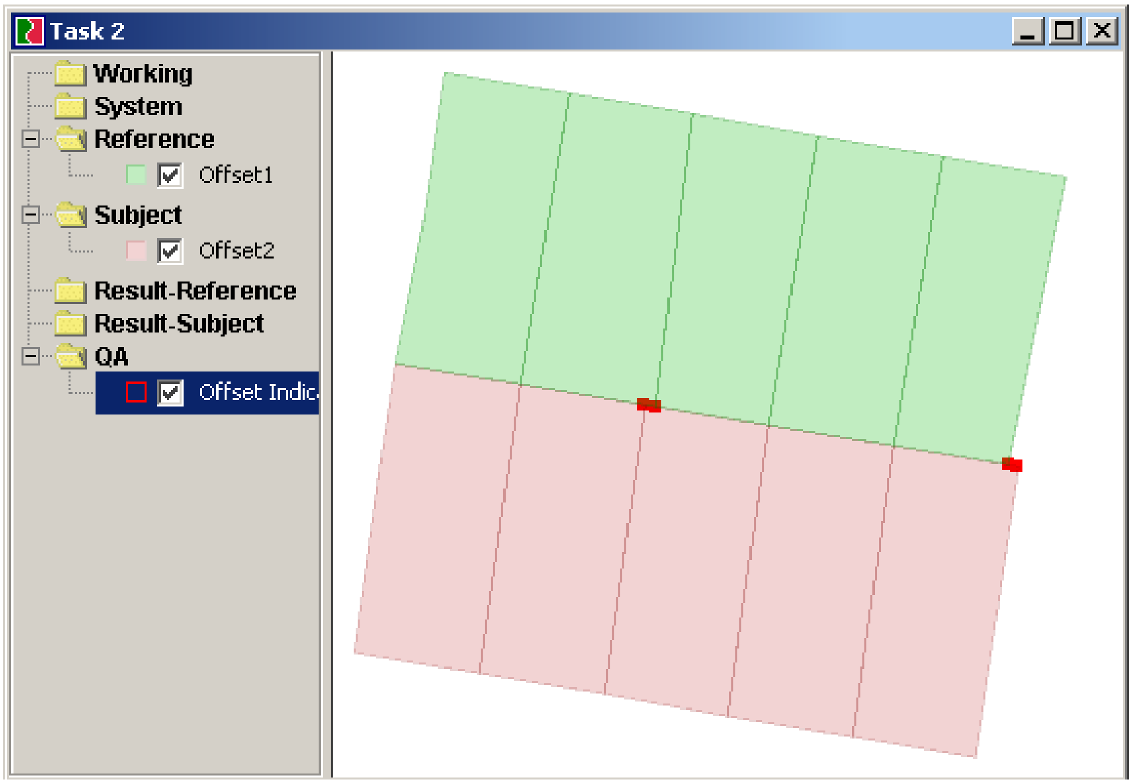


**Figure 21 - Output from the Offset Boundary Corner Finder**

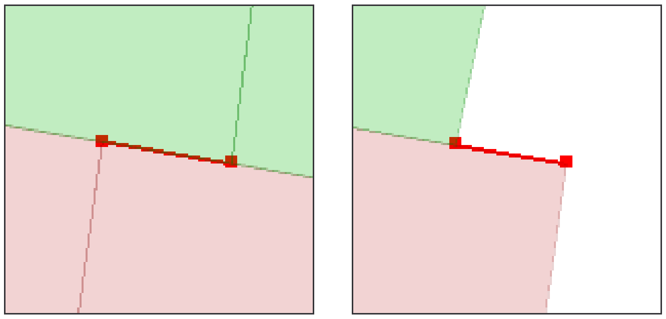
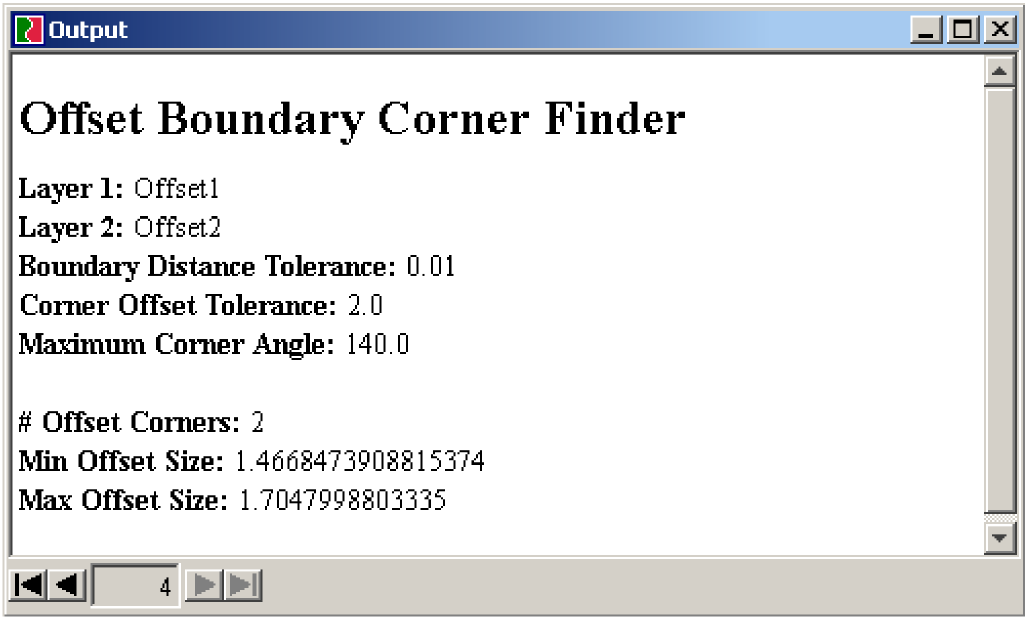


**Figure 22 - Closeup of the Offset Corner indicators**

- View the report summarizing the results by clicking on the **Output Window** tool (🗔).



Offset Boundary Corner Finder

**Layer 1:** Offset1
**Layer 2:** Offset2
**Boundary Distance Tolerance:** 0.01
**Corner Offset Tolerance:** 2.0
**Maximum Corner Angle:** 140.0

**# Offset Corners:** 2
**Min Offset Size:** 1.4668473908815374
**Max Offset Size:** 1.7047998803335

**Figure 23 - Output Report for the Offset Boundary Corner Finder**