

Geographic Information Systems Stack Exchange is a question and answer site for cartographers, geographers and GIS professionals. It's 100% free, no registration required.

Here's how it works:

Sign up

Anybody can ask a question

Anybody can answer

The best answers are voted up and rise to the top

ArcGIS: Create a line layer from a point layer and csv data file

UPDATED DECEMBER 20

I have a Regular point layer and a csv file that has the same ids as the point layer. The csv file is organized as

| ID | From | To | Weight |
|----|------|------|--------|
| -- | ---- | -- | ----- |
| 01 | 1234 | 1235 | 2 |
| 02 | 1234 | 1236 | 4 |
| 03 | 1234 | 1237 | 1 |
| 04 | 1235 | 1234 | 4 |
| 05 | 1236 | 1235 | 3 |

// etc

the point layer is organized in the following way:


MY_ID

1234
1235
1236
1237

the From / To fields correspond to a my_ID in the point layer. I'd like to create a new line layer, such that the From, To, Weight correspond to both the point and csv files

geoprocessing arcgis-9.3.1 point polyline-creation

edited Jan 23 '11 at 18:13

 whuber ♦
46.7k 9 116 186

asked Nov 22 '10 at 15:14

 dassouki
4,240 6 41 91

With spatial SQL (PostGIS) it could have looked like this: CREATE TABLE new_layer AS SELECT c.id, ST_Makeline(pfrom.the_geom, pto.the_geom) AS the_geom, c.weight FROM csv_table c INNER JOIN pointlayer AS pfrom ON c.from=pfrom.my_id INNER JOIN pointlayer AS pto ON c.to=pto.my_id; – Nicklas Avén Jan 6 '11 at 14:05

and that is affordable ;-) – Nicklas Avén Jan 6 '11 at 14:05

@Nicklas Avén - We're a strictly arcgis house, and I don't have permission to install postgis at work and I can't take the data home – dassouki Jan 6 '11 at 14:13

I know the situation :-(– Nicklas Avén Jan 6 '11 at 14:15

Shouldn't there be commas separating the values in your csv file? – Kirk Kuykendall Jan 6 '11 at 18:47

9 Answers

We had a similar need, so I wrote up the following python script. It has been tested with CSV, shapefiles, PGDB*, and FGDB. It works on 9.3 and 10. It can be run from the OS command line or imported into toolbox and run from toolbox or the ArcGIS command line.

```
# -----  
# Table2Lines.py  
# Created: 2011-01-12  
# Author: Regan Sarwas, National Park Service  
#  
# Builds a line feature class from a line data table and a point feature class  
# The line table must have three fields for the line Id, point1 Id, and point2 Id  
# The point feature class must be a point (not multi-point) shape type and have a  
# field with the point id that matches the type and domain of the point ids in the line  
# table.  
# The names of all fields are provided in the input arguments
```

```

#
# usage:
# python Table2Lines.py path_to_line_table Line_ID_Field From_Point_ID_Field
#                               To_Point_ID_Field
#                               path_to_point_FC Point_ID_Field output_line_FC
#
# example:
# python Table2Lines.py "c:\tmp\lines.csv" "ID" "From" "To" "c:\tmp\pts.shp" "MY_ID"
# "c:\tmp\lines.shp"
#
# License:
# Public Domain
# Disclaimer:
# This software is provide "as is" and the National Park Service gives
# no warranty, expressed or implied, blah, blah, blah
# -----

import os, arcgisscripting
gp = arcgisscripting.create(9.3)
gp.Overwriteoutput = 1

#I use one search cursor and cache all the points in a dictionary.
#This avoids creating a search cursor for each point as lines are processed
#Assumes Python is more efficient and faster than ArcGIS. Should be tested.
def GetPoints(pointFC,pointIDField):
    points = {}
    pointDescription = gp.Describe(pointFC)
    pointShapeField = pointDescription.ShapeFieldName
    pointIDFieldDelimited = gp.AddFieldDelimiters(pointFC, pointIDField)
    where = pointIDFieldDelimited + " is not null"
    spatialRef = ""
    fields = pointIDField + "; " + pointShapeField
    sort = ""
    pts = gp.SearchCursor(pointFC, where, spatialRef, fields, sort)
    pt = pts.Next()
    while pt != None:
        points[pt.GetValue(pointIDField)] = pt.Shape.getPart()
        pt = pts.Next()
    return points

def MakeLine(pt1, pt2):
    """ pt1 and pt2 should be gp point objects or None """
    if (pt1 == None or pt2 == None):
        return None
    pts = gp.createObject("Array")
    pts.add(pt1)
    pts.add(pt2)
    line = gp.createObject("geometry", "polyline", pts)
    if (line == None) or (line.FirstPoint == None) or (line.LastPoint == None):
        return None
    return line

# Input field types must be in mapType (defined below).
# Point id type in both input data sets must map to the same type, i.e. OID and Integer

# Maps the string returned by gp.describe.Field.Type to the string required by
gp.AddField()
mapType = {"SmallInteger" : "SHORT",
           "Integer" : "LONG",
           "Single" : "FLOAT",
           "Double" : "DOUBLE",
           "String" : "TEXT",
           "Date" : "DATE",
           "OID" : "LONG",          #Not usually creatable with AddField() - use with Caution
           "Geometry" : "BLOB",     #Not usually creatable with AddField() - use with Caution
           "BLOB" : "BLOB"}

#GET INPUT
lineTable = gp.GetParameterAsText(0)
lineIDField = gp.GetParameterAsText(1)
fromPointIDField = gp.GetParameterAsText(2)
toPointIDField = gp.GetParameterAsText(3)
pointFC = gp.GetParameterAsText(4)
pointIDField = gp.GetParameterAsText(5)
lineFC = gp.GetParameterAsText(6)

#VERIFY INPUT (mostly for command line.  Toolbox does some validation for us)
lineIDFieldType = ""
fromPointIDFieldType = ""
toPointIDFieldType = ""
tableDescription = gp.Describe(lineTable)
for field in tableDescription.Fields:
    if field.Name == lineIDField:
        lineIDFieldType = mapType[field.Type]
    if field.Name == fromPointIDField:
        fromPointIDFieldType = mapType[field.Type]
    if field.Name == toPointIDField:
        toPointIDFieldType = mapType[field.Type]

if lineIDFieldType == "":
    raise ValueError("Field '" + lineIDField + "' not found in " + lineTable)
if fromPointIDFieldType == "":
    raise ValueError("Field '" + fromPointIDField + "' not found in " + lineTable)
if toPointIDFieldType == "":
    raise ValueError("Field '" + toPointIDField + "' not found in " + lineTable)

pointDescription = gp.Describe(pointFC)

```

```

if pointDescription.shapeType != "Point":
    raise ValueError(pointFC + " is a " + pointDescription.shapeType +
        " not a Point Feature Class.")

pointIdFieldType = ""
for field in pointDescription.Fields:
    if field.Name == pointIdField:
        pointIdFieldType = mapType[field.Type]
        break

if pointIdFieldType == "":
    raise ValueError("Field '" + pointIdField + "' not found in " + pointFC)
if (pointIdFieldType != fromPointIdFieldType or
    pointIdFieldType != toPointIdFieldType):
    raise ValueError("Field types do not match - cannot link points to lines.")

gp.AddMessage("Input validated")

# Create Feature Class...
outSpatialRef = pointDescription.SpatialReference
outPath, outName = os.path.split(lineFC)
gp.CreateFeatureclass_management(outPath, outName, "POLYLINE", "", "DISABLED", "DISABLED",
    outSpatialRef, "", "0", "0", "0")

gp.AddMessage("Created the output feature class")

# Add Fields...
lineIdFieldValid = gp.ValidateFieldName(lineIdField,outPath)
gp.AddField_management(lineFC, lineIdFieldValid, lineIdFieldType, "", "", "", "",
    "NON_NULLABLE", "REQUIRED", "")
fromPointIdFieldValid = gp.ValidateFieldName(fromPointIdField,outPath)
gp.AddField_management(lineFC, fromPointIdFieldValid, fromPointIdFieldType, "", "", "",
    "", "NON_NULLABLE", "REQUIRED", "")
toPointIdFieldValid = gp.ValidateFieldName(toPointIdField,outPath)
gp.AddField_management(lineFC, toPointIdFieldValid, toPointIdFieldType, "", "", "",
    "NON_NULLABLE", "REQUIRED", "")

gp.AddMessage("Added the fields to the output feature class")

points = GetPoints(pointFC,pointIdField)

fromPointIdFieldDelimited = gp.AddFieldDelimiters(lineTable, fromPointIdField)
toPointIdFieldDelimited = gp.AddFieldDelimiters(lineTable, toPointIdField)
where = fromPointIdFieldDelimited + " is not null and " + toPointIdFieldDelimited + " is
not null"
spatialRef = ""
#fields = lineIdField + "; " + fromPointIdField + "; " + toPointIdField
fields = ""
sort = ""
#Create the input(search) and output(insert) cursors.
lines = gp.SearchCursor(lineTable, where, spatialRef, fields, sort)
newLines = gp.InsertCursor(lineFC)

line = lines.Next()
while line != None:
    pt1Id = line.GetValue(fromPointIdField)
    pt2Id = line.GetValue(toPointIdField)
    try:
        lineGeom = MakeLine(points[pt1Id],points[pt2Id])
    except:
        lineGeom = None
    if lineGeom == None:
        gp.AddWarning("Unable to create line " + str(line.GetValue(lineIdField)))
    else:
        # Create a new feature in the feature class
        newLine = newLines.NewRow()
        newLine.Shape = lineGeom
        newLine.SetValue(lineIdFieldValid, line.GetValue(lineIdField))
        newLine.SetValue(fromPointIdFieldValid, pt1Id)
        newLine.SetValue(toPointIdFieldValid, pt2Id)
        newLines.insertRow(newLine)
    line = lines.Next()

#Closes the insert cursor, and releases the exclusive lock
del newLine
del newLines
gp.AddMessage("Done.")

```

*output to PGDB does not work due to a schema lock problem

Also, it does not copy any attributes from the line table, so those must be joined after the line feature class is created.

answered Jan 13 '11 at 1:36



Regan Sarwas
1,412 8 18

I was late to get to work today; other wise, I would've given you the bounty :(– [dassouki](#) Jan 13 '11 at 13:38

My fault for not responding sooner. Glad it helped. – [Regan Sarwas](#) Jan 13 '11 at 16:55

dassouki, can you work with Hawth's Analysis Tools for ArcGIS 9.x?

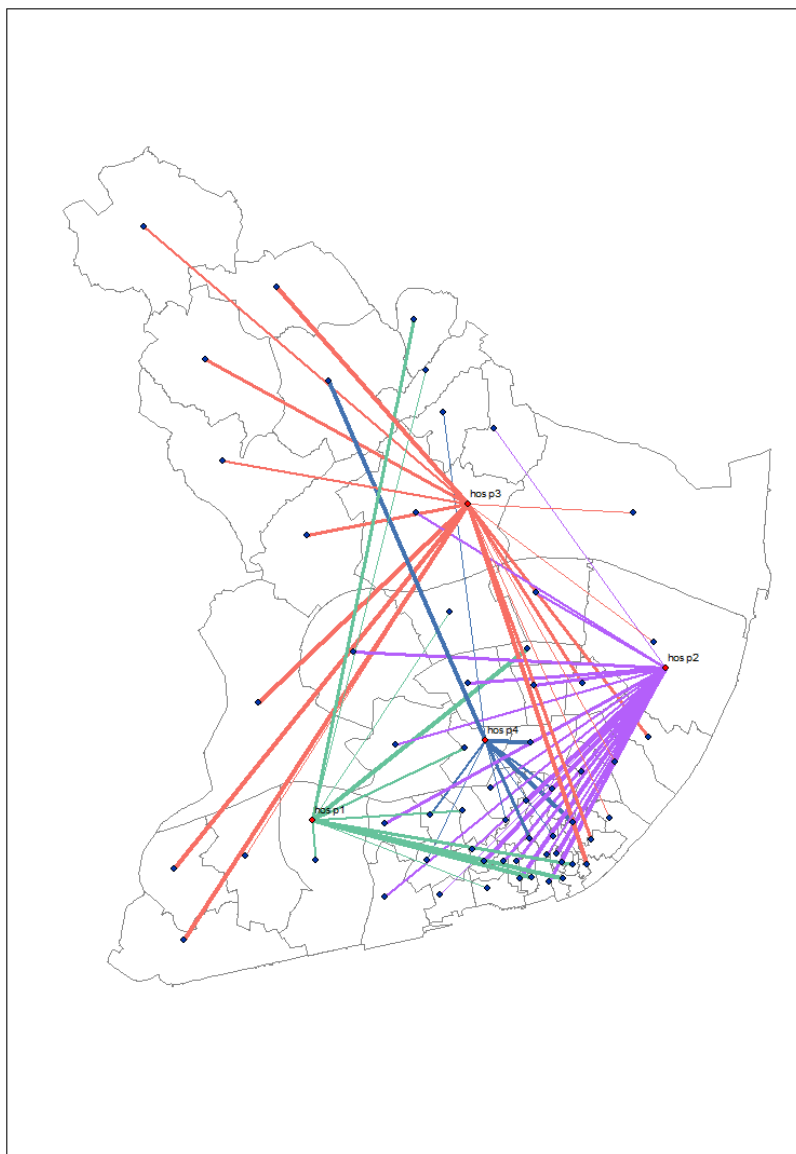
--EDIT--

with "Add XY to Table (points)" (in "Table tools" from Hawth's bar), create the fields with the coordinates from each point from the point layer.

in excel, for example, open the csv file and add 4 fields: fromX, fromY, toX, toY and copy the coordinates from each point (work with filters, is quite fast) and then save the csv file

in arcmap, use "Add XY line data from table" (in "Table tools" from Hawth's bar) in a line layer using the csv file as source and the various connections are created

in the line layer symbology, use "Multiple attributes - Quantity by category", using Weight field
this is the result of my test:



edited Jan 7 '11 at 0:18

answered Jan 6 '11 at 16:31



Marinheiro

655 3 9 15

Yes, I have it installed already – dassouki Jan 6 '11 at 17:10

Take a look at the sample Python script in the link below (from ArcGIS Help). It is designed to turn a text file of lat/long into points, then into lines.

I think has all the concepts there that could modify in order to accomplish your task. If you haven't used Python, its fairly easy to learn and you will find a lot more uses for it as you continue learning more.

http://webhelp.esri.com/arcgisdesktop/9.3/index.cfm?TopicName=Writing_geometries

answered Jan 6 '11 at 20:15



RyanDalton

14.2k 5 60 128

Add your csv file to arcmap (from1234 to1235 are coordinates?)


add your point layer.

merge both to a new point layer

use Point To Polyline from GeoWizards based on ID to generate a line with all the attribute data.

GeoWizards (lot of free function including Point To Polyline) <http://www.ian-ko.com/>

answered Nov 22 '10 at 16:31



Mapperz ♦


38.9k 5 41 103

1234 and 12345 are point ids ... the same ids from the point layer – [dassouki](#) Nov 22 '10 at 17:30

The other issue is, i have multiple points that are destinations and multiple that are only origins. – [dassouki](#) Nov 22 '10 at 19:24

- at 10.0 i'd do a model or py script with this sequence of tools
- 1) AddXY on the points to get XY coordinates at attributes
 - 2) CopyRows to load the csv into a gdb table (for it to be valid for steps 3 & 4)
 - 3) AddJoin based on the From to get xy of the from point (bring the weight field you're at it)
 - 4) AddJoin based on the To to get the xy of the to point (names will be x_1 and y_1 i think)
 - 5) XYToLine tool will turn this type of table into a line feature class

answered Nov 22 '10 at 19:51



gotchula

2,420 8 12

I can't afford 10.0 right now :(– [dassouki](#) Nov 22 '10 at 20:05

this process should work in 9.3.1 also. It doesn't have to be scripted does it? Just manually follow the steps. But I think I would add an export to txt between 4 and 5. I have a txt file example and the command line at work. I will post them tom. – [Brad Nesom](#) Jan 7 '11 at 4:57

Here's one idea: This might not be the simplest solution, but have you tried the 'Create Features from Text File' tool in ArcToolbox? I'm running 9.3 and in my toolbox it's listed under 'Samples' -> 'Data Management' -> 'Features'. Looks like it could be part of a solution. You'd first have to get x,y for each point, then you'd need to join the x,y data with your 'from - to' data in the csv file using your my_id. Then you'd need to do some reorganization of your data - probably by running it through a script that would take a table that looks like this:

| ID | From | From_x | From_y | To | To_x | To_y | Weight |
|----|------|--------|--------|------|------|-------|--------|
| -- | ---- | ----- | ----- | -- | ---- | ----- | ----- |
| 01 | 1234 | 1.0 | 2.0 | 1235 | 3.0 | 2.0 | 2 |
| 02 | 1234 | 1.0 | 2.0 | 1236 | 3.5 | 2.0 | 4 |

And create a text file that looks like this: (the arcgis help on the tool explains the formatting)

```
Line
01 0
0 1.0 2.0 0 0
1 3.0 2.0 0 0
02 0
0 1.0 2.0 0 0
1 3.5 2.0 0 0
```

END

You'd need to do another join to the features created from the text file to the original CSV get your 'weight' value in the lines.

answered Jan 6 '11 at 15:36



Not sure if this will work in your case, but could this method work?

answered Nov 22 '10 at 20:47


 Chad Cooper
9,649 3 24 60

If Geo-Wizards works (I just tried 2 ArcScripts that didn't work), you just need to make a Cartesian Result in some sort of SQL (SQL Server Express, MySQL . . .)

(Your point layer I named pt and csv file = PTCSV and I made up some x,y values)

```
SELECT id, my_id, X, Y, [From], [To]
FROM PTCSV AS T1
      CROSS JOIN
pt AS T2

WHERE MY_ID = [FROM]
OR MY_ID = [TO]
ORDER BY id
```

You will get the following result:

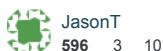
| id | x | y | From | To | Weight |
|----|-----|-----|------|------|--------|
| 01 | 5 | 5 | 1234 | 1235 | 2 |
| 01 | 10 | 20 | 1234 | 1235 | 2 |
| 02 | 5 | 5 | 1234 | 1236 | 4 |
| 02 | 5 | -10 | 1234 | 1236 | 4 |
| 03 | 5 | 5 | 1234 | 1237 | 1 |
| 03 | -10 | -30 | 1234 | 1237 | 1 |
| 04 | 5 | 5 | 1235 | 1234 | 4 |
| 04 | 10 | 20 | 1235 | 1234 | 4 |
| 05 | 10 | 20 | 1236 | 1235 | 3 |
| 05 | 5 | -10 | 1236 | 1235 | 3 |

Then, if Geo-Wizards works how the other 2 scripts I tried were supposed to, just group by id. [You may have to create a shapefile 1st (Display X,Y Data and export)]

If this would work, but you don't have access to SQL, you could accomplish the same thing in ArcMap by Joining the layer to the .csv `From = MY_ID` and then export as a table. Repeat with `To = MY_ID` and just copy and paste the results in Excel or something, add them together, and bring them back in to ArcMap

1 of the scripts that I tried <http://arcscrips.esri.com/details.asp?dbid=15828> worked as far as producing the lines, but the attributes came out wrong.

answered Jan 6 '11 at 19:40



I did something similar in VBA long ago. I've changed up and abbreviated the code to better apply it to your question. Run this macro in the Visal Basic Editor window while in an edit session for the line layer you wish to update.

```
Sub CreateLines()

On Error GoTo ErrorHandler

Dim document As IMxDocument
Set document = ThisDocument

' Get the points layer, first layer in TOC
Dim pointsLayer As IFeatureLayer
Set pointsLayer = document.FocusMap.Layer(0)

' Get the lines layer, second layer in TOC
Dim linesLayer As IFeatureLayer
Set linesLayer = document.FocusMap.Layer(1)
```

```

' Open the CSV data
Open "C:\Temp\test.csv" For Input As #1
Do While Not EOF(1)

    ' Get values for current csv line/row
    Dim row As String
    Line Input #1, row
    Dim values As Variant
    values = Split(row, ",")
    Dim lineID As Integer
    lineID = CInt(values(0))
    Dim fromID As Integer
    fromID = CInt(values(1))
    Dim toID As Integer
    toID = CInt(values(2))
    Dim weight As Integer
    weight = CInt(values(3))

    ' Find the from point
    Dim fromFilter As IQueryFilter
    Set fromFilter = New QueryFilter
    fromFilter.WhereClause = "PointID = " & fromID
    Dim fromCursor As IFeatureCursor
    Set fromCursor = pointsLayer.Search(fromFilter, False)
    Dim fromPointFeature As IFeature
    Set fromPointFeature = fromCursor.NextFeature
    Dim fromPoint As IPoint
    Set fromPoint = fromPointFeature.ShapeCopy

    ' Find the to point
    Dim toFilter As IQueryFilter
    Set toFilter = New QueryFilter
    toFilter.WhereClause = "PointID = " & toID
    Dim toCursor As IFeatureCursor
    Set toCursor = pointsLayer.Search(toFilter, False)
    Dim toPointFeature As IFeature
    Set toPointFeature = toCursor.NextFeature
    Dim toPoint As IPoint
    Set toPoint = toPointFeature.ShapeCopy

    ' Create the line feature
    Dim newLineFeature As IFeature
    Set newLineFeature = linesLayer.FeatureClass.CreateFeature
    newLineFeature.Value(newLineFeature.Fields.FindField("LineID")) = lineID
    newLineFeature.Value(newLineFeature.Fields.FindField("FromPointID")) = fromID
    newLineFeature.Value(newLineFeature.Fields.FindField("ToPointID")) = toID
    newLineFeature.Value(newLineFeature.Fields.FindField("Weight")) = weight

    ' Set the new line geometry
    Dim lineGeometry As IPolyline
    Set lineGeometry = New Polyline
    lineGeometry.fromPoint = fromPoint
    lineGeometry.toPoint = toPoint
    Set newLineFeature.Shape = lineGeometry
    newLineFeature.Store

    ' Clean up
    Set fromFilter = Nothing
    Set fromCursor = Nothing
    Set fromPointFeature = Nothing
    Set fromPoint = Nothing
    Set toFilter = Nothing
    Set toCursor = Nothing
    Set toPointFeature = Nothing
    Set toPoint = Nothing

Loop
Close #1


ErrorHandler:
Close #1

End Sub

```

NOTE: this code has not been tested and has no real error checking, so keep that in mind. Also, VBA is certainly on its way out so you might want to consider digging to Python for a longer term solution.

answered Jan 7 '11 at 19:44

 Brenner256
486 2 7