Office of Statewide Health Planning and Development
Healthcare Workforce and Community Development Division

Title

From ZIP Codes to a Better Framework for Health Care Data

Abstract

ZIP Code level data is widely used to publish health care information for research and analysis. However, there are often too few data points from any one ZIP Code for public release under current federal law (HIPAA). More important, ZIP Code boundaries change frequently over time according to the needs of the United States Postal Service (USPS). Finally, ZIP Codes offer no consistent approach for delivering demographic indicators. This paper articulates a method of creating annual ZIP Code crosswalks to a more rigorous, census-based spatial framework--the Medical Service Study Area (MSSA). The approach makes use of weighted population centers, the MSSA and a probability score that a ZIP Code is properly matched. Development of these annual crosswalks enables data providers to maintain strict compliance with privacy rules, offers a more effective spatial unit for multiple year analysis, and supports meaningful demographic denominators for more robust patient care studies.

Introduction

ZIP Codes were created by the United States Postal Service (USPS) and first used in 1963. ZIP stands for Zoning Improvement Plan and the area is used as a tool to help the Postal Service more efficiently deliver mail. The first implementation of the ZIP, post World War II, was to identify the individual post office or metropolitan area delivery station associated with an address. At that point, the first three digits identified the delivery area of a sectional center facility or a major-city post office serving the delivery address area. The next two (the fourth and fifth) digits identified the delivery area of an associate post office, post office branch, or post office station. All post offices are assigned at least one unique 5-digit code. Since the original inception of the office location, the ZIP Code has taken on an implicit area definition. The implicit area is defined by the streets the Postal carriers travel in order to deliver the mail. In order for this to be efficient, the drive time from the Post Office (origin of incoming mail) to each address in the area should be minimized. This often places the Post Office in the center of commerce, or most populous area, but not always. Since the post WWII population migration from inner city to suburbia and now the increased movement to xurbia, the location of the post office no longer carries the same significance. Given these residence shifts, the US Post Office found it important to modify the coding scheme and implemented the ZIP+4 method. ZIP+4 is an enhanced code consisting of the 5-digit ZIP Code and four additional digits that identify a specific range of delivery addresses.

The problems of using ZIP Codes in GIS as an spatial framework for analysis and display stem from the fact that they are collections of street networks the carriers use to deliver mail. Sometimes these networks do not define contiguous or clearly definable regions and areas. Without a well defined boundary ZIP Codes are able to cross county and even state lines, including political jurisdictions. Furthermore there is no "'crosswalk' or relationship files that associate US postal service zips to any legal or statistical census geography. Most importantly, however, ZIP Codes are at the whim of a continuously migrating population. In places with little population change, this is not a problem. However, in California where the population is expected to nearly double over the next 50 years (from 32 million in 2000 to 50 million or more in 2050), the result is a vastly different ZIP Code geography over time. As an example, the City of Elk Grove is the 3rd fastest growing city in California[1]. Its primary ZIP Code, 95758, contained the geography depicted below.

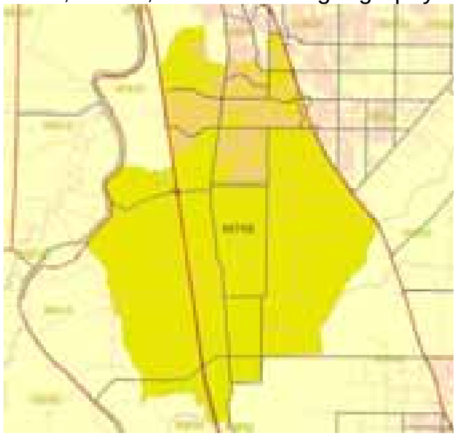


Figure 1 - ZIP Code 95758 in 2002

Since 2002 (just four years ago), its population has seen an annual population increase of nearly 8%, resulting in significant pressure on the Postal Service to deliver mail to those residents. The Postal Service therefore modified this ZIP Code, splitting the delivery zone creating a new ZIP Code (95757). Figure 2 displays the new geography of this area. One can easily see the significant boundary change as it applies to the region.

[1] California Department of Finance http://www.dof.ca.gov/HTML/DEMOGRAP/e-1press.pdf#search='elk%20grove%20dof%20fastest%20growing'

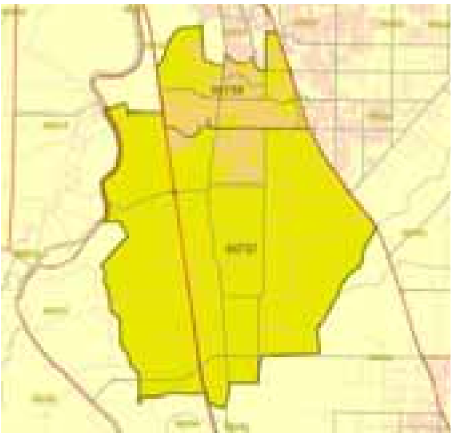


Figure 2 - New ZIP Codes as of 2005

Medical Service Study Area (MSSA)

Medical Service Study Areas (MSSA) are the defined geographic analysis unit for the Office of Statewide Health Planning and Development. MSSA's are reproduced on the decennial census and the boundaries are formally approved by the Healthcare Workforce Policy Commission. Moreover, the US Department of Health and Human Services, Health Resources and Services Administration (HRSA) formally recognizes California MSSA as the Rational Service Area (RSA) for medical service for California.

The Song Brown Family Physician Training Act (1973) created the California Healthcare Workforce Policy Commission (CHWPC), conferring upon them the legal authority for determining areas of unmet need for physicians. Subsequently, the Garamendi Rural Health Services Act (1976) requires Commission to determine which areas of the state are rural and which urban, and which rural areas are deficient in medical services. Changes to the boundaries of MSSAs can only be made through motions adopted by the CHWPC via the formal MSSA Reconfiguration Adoption Process. Any such motions must be published agenda items of the CHWPC and generally are accompanied by letters of support from community officials and stakeholders.

The US Health Resources and Services Administration recognizes MSSAs as "Rational Service Areas" for purposes of determining Health Professional Shortage Areas (HPSA), Medically Underserved Areas (MUAs), and Medically Underserved Populations (MUPs.) While not performing as the key framework geography for healthcare planning and analysis in California, MSSAs have the potential for assisting in needs assessment, health planning and health policy development.

The following are the rules for defining MSSA geography:

- Each MSSA is composed of one or more complete census tracts.   MSSAs do not cross county lines. All census-defined places within the MSSA are within 30 minutes travel time to the largest population center within the MSSA, except in those circumstances where meeting this criterion would require splitting a census tract.

- Areas that, standing alone, would meet both the definition of an MSSA and a Rural MSSA, should not be a part of an Urban MSSA.
- Urban Definitions: Any Urban MSSA whose population exceeds 200,000 shall be divided into two or more Urban MSSA Subdivisions. Urban MSSA Subdivisions should be within a population range of 75,000 to 125,000, but may not be smaller than five square miles in area. If removing any census tract on the perimeter of the Urban MSSA Subdivision would cause the area to fall below five square miles in area, then the population of the Urban MSSA may exceed 125,000. To the extent practicable, Urban MSSA Subdivisions should reflect recognized community and neighborhood boundaries and take into account such demographic information as income level and ethnicity.
- Rural Definitions: A rural MSSA is an MSSA adopted by the Commission, which has a population density of less than 250 persons per square mile, and which has no census defined place within the area with a population in excess of 50,000. Only the population that is located within the MSSA is counted in determining the population of the census defined place.
- A frontier MSSA is a rural MSSA adopted by the Commission which has a population density of less than 11 persons per square mile. Any MSSA which is not a rural or frontier MSSA is an urban MSSA.

There are 541 MSSA's recognized in the State as of the writing of this paper. These areas will stay defined until the next Reconfiguration process in 2011. These units contain the full suite of functional demographic information from the US Census Bureau.

OSHPD and Health Data Publication

Hospitals are required to submit Patient Discharge Data semi-annually to the OSHPD. The data reported for every patient discharged from a California hospital: Patient demographic information, such as age, sex, county of residence and ZIP Code, and race/ethnicity; Diagnostic information, ICD-9-CM diagnostic codes, DRG & MDC groupings; Treatment information, ICD-9CM procedure codes; External Cause of Injury codes (E-codes) and total charges with expected principal source of payment. Hospital discharge data is maintained for in-patient, emergency room and ambulatory surgery visits. The data volume reported is roughly 20 million records per year.

The Enterprise Geographic Information System Project at OSHPD is developing tools to publish all key data holdings at OSHPD to its constituents. These constituents include residents, research institutions, education programs and policy makers. A fundamental unit for the delivery of health data is the geographic context of the data. Currently, that context is the ZIP Code. However, given the problems articulated above, OSHPD needs to develop a geographic framework for publishing all of it's health data for which it is the steward. OSHPD needs to reduce its risk of the geographic framework for its data misrepresenting the facts. Some of these risks could be mis-identifying an area as a cluster of cancer, failing to show an area as receiving inequity in care, and/or improperly labeling a region as having an adequate supply of healthcare workforce.

Due to the Health Information Portability and Accountability Act (HIPAA), OSHPD patient and other data are primarily tied to ZIP Codes as a geographic indicator. Again, given the problems mentioned above with ZIP Codes changing and no clear demographic indicators, we embarked on a process to try cross-walking ZIP Codes in California to the codified[2], stable Medical Service Study Areas. This analysis contained below, demonstrates our methods and discusses why we consider this an appropriate method of resolution.

Methods

This section describes the iteration process of cross-walking ZIP Codes to Medical Service Study Areas (MSSA's). The process outlined matches ***each ZIP Code to a single MSSA***. We also attach a code to each record which indicates the match confidence. Prior to generating the crosswalk, input data is pre-processed. Next the annual crosswalk, from ZIP Codes to a single MSSA, is generated. For the purposes of this exercise two Python scripts were developed. Script 1, called CreateMeanCenter.py (See Appendix) is the pre-process piece and creates a population weighted centroid for each ZIP Code area. Script 2, called MSSA_Zip.py (See Appendix) is the processing engine and generates a single crosswalk table for a given year. Each table represents an individual year of ZIP Codes cross-walked to MSSA's. The Python scripts are then stitched into the ESRI Geoprocessing framework as a tool, so it can be run interactively and called on by other processing components as needed.

Pre-Process

Prior to processing and generating yearly crosswalks, the following must be in place: 1) a ZIP Code point layer for enclosed zip codes for each year, 2) a ZIP Code polygon layer for each year, 3) a weighted mean center ZIP Code point layer representing the population mean center of the ZIP Code, and 4) a MSSA layer to on which to process. Listed below are the pre-processing steps.

1. Acquire ZIP points, which have a flag for 'enclosed'[3]. For the purposes of this paper and OSHPD business, our data comes from TeleAtlas.
2. Dissolve each ZIP Code polygon layer. The dissolve function creates one polygon per ZIP Code and ensures there are no 'extraneous' or floating zip polygons which will hinder the analysis process. Performing this for each year produces ZIP Code polygon layers. The dissolve ensures all non-contiguous same code ZIP Codes are maintained as one feature from which a mean weighted center is defined.
3. Create a population weighted center for each dissolved zip boundary in #2 above (See Appendix). The method for this layer is to measure the mean weighted center of all ZIP

[2] Medical Service Study Areas are codified in the Song-Brown Act 1976 in order to identify Primary Care Shortage areas. The boundaries are formally adopted by the California Healthcare Workforce Policy Commission

[3] TeleAtlas describes 'Enclosed' Zip codes as ENCLOSING ZIP For all point ZIPs, the Dynamap/ZIP Inventory file provides an enclosing ZIP to which data for that point ZIP can be mapped. The enclosing ZIP is identical for all entries that are located within the same ZIP. Enclosing ZIPs are particularly important for Boundary File users who want to map point ZIPs to enclosing ZIPs. To assign the enclosing ZIP, the actual location of the point is determined on a map. The boundary ZIP associated with the geographic location is the enclosing ZIP. NOTE: Universities, military bases, large hospitals, etc. may be defined as points or polygons.

Codes using the population value from each census block[4]. For this process we use the total population of Census Blocks entirely within each given ZIP polygon. This process is run for each year we have ZIP Code data. This process actually has a stepped logic under the following rules:

a. For each zip code determine the Census Blocks within that area
b. Select from those census blocks only those blocks with a total population greater than 0
c. Perform the MeanCenter function on those Census blocks.
d. Check to see if this population weighted mean center (output from 3.c) is contained within the original ZIP polygon
    i. If Yes, move to next polygon
    ii. If No, then use the CentralFeature function on the Census Blocks to determine a point that is actually within the ZIP polygon.

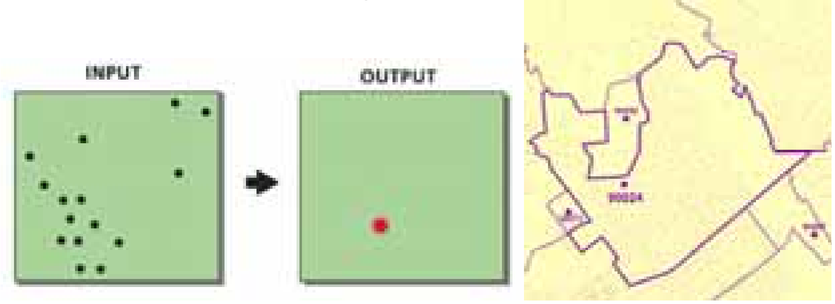


Figure 3 – Central Feature identifies the most centrally located feature in the selection set, ensuring the output is inside the original ZIP polygon. In the example on the right, the weighted mean center for ZIP Code 90024 would be outside the polygon boundary, so the Central Feature Function is run and places the ZIP point at the most central feature of the Census Block Points.

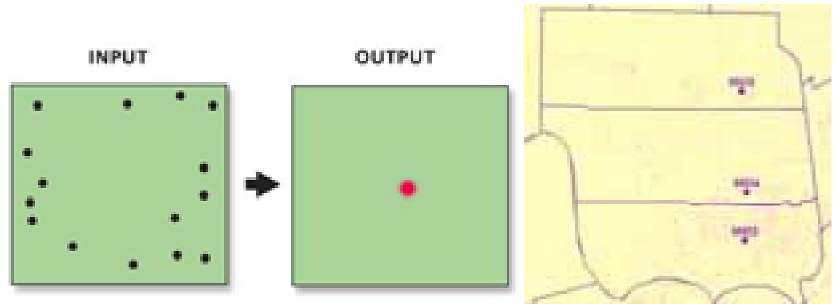


Figure 4 – Mean Center identifies the weighted geographic center of the set of features. In the example on the right, ZIP Codes 95835, 95834, and 95833 all have population centers in the southwest portion of the ZIP Code polygons (not in the center of the polygon), because that is where the majority of the population resides in each of these ZIP Codes.

[4] This mean center will be used as a late stage for determining the cross walk to a MSSA.

Process

The process is run in order from 1 below to 4 below. Each time a ZIP Code polygon finds a corresponding match to a given MSSA using the logic below it is removed from the set and not processed again.

1. Point in MSSA. Enclosed Zips (see pre-process #1) are matched directly to a MSSA if the zip point occurs within that MSSA. We assign a high degree of certainty to the crosswalk. Confidence Code = 1 (Very confident). Since these points are typically whole buildings or institutional populations, there is no variance in their location. These ZIP Codes are definitely within a single MSSA.

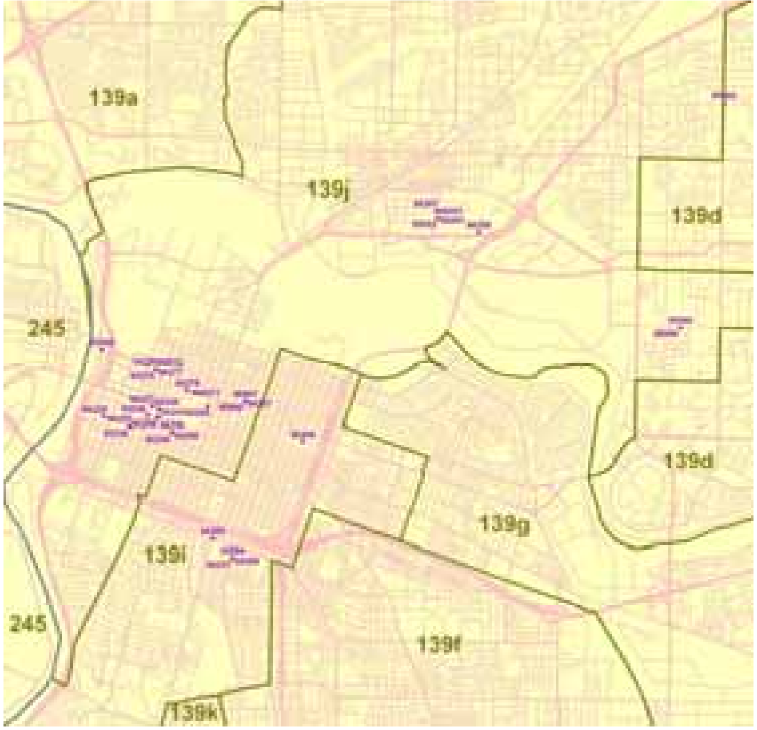


Figure 5 – Enclosed ZIP: This map illustrates enclosed zip locations (labeled in purple) within MSSA boundaries (labeled in olive green). Here ZIPs 94294, 94293, 95894, 94249, and 94297 are cross walked directly (with high confidence) to MSSA 139i.

2. Entirely contained within MSSA. For those ZIP Codes remaining, the processing checks to find those ZIP Codes whose polygon boundaries are entirely within a given MSSA. For each of these, we are very confident that the ZIP population is within a given MSSA and assign a confidence code of 2 for this match.

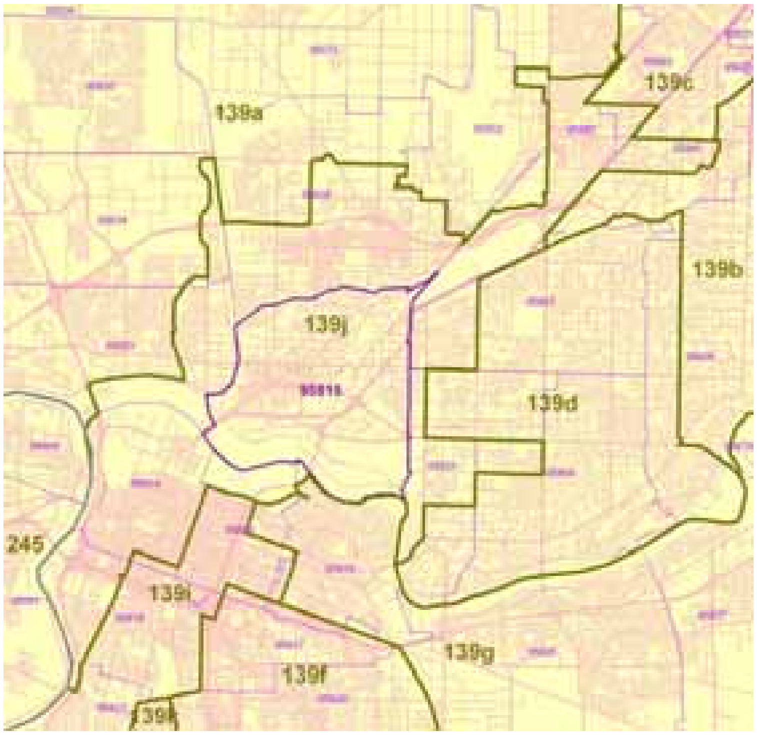


Figure 6 – Entirely Contained Within: This map shows how ZIP Code 95815 (bold purple outline) is entirely within MSSA 139j and therefore is cross walked to that MSSA with high confidence.

3. Mostly Contained. For those ZIP Codes remaining, if 70% of the ZIP Code area is within the MSSA, we match that ZIP Code to that MSSA. We assign a confidence code of 3 (somewhat confident) and tally the percent of the area overlapped.

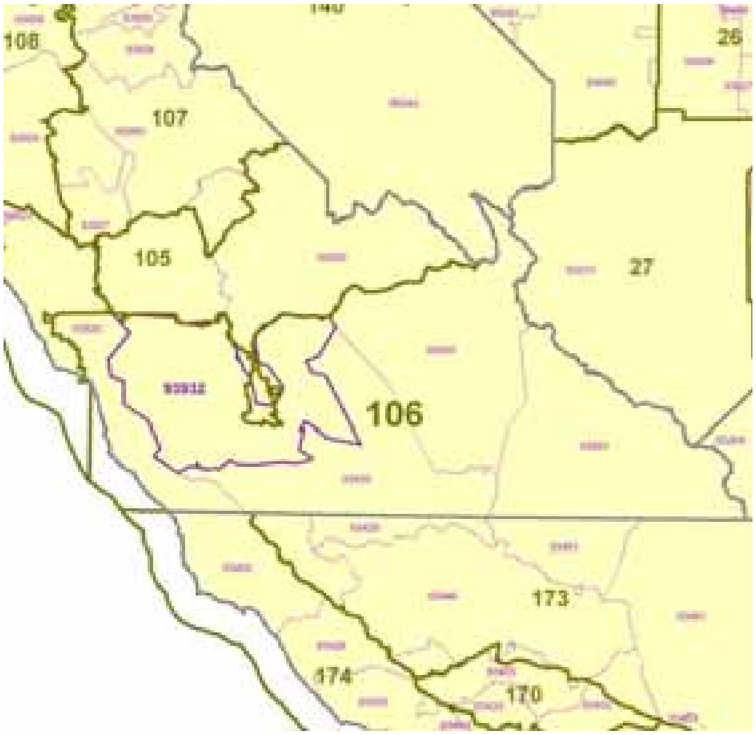


Figure 7 – Mostly Contained Within: This map shows ZIP Code 93932 (bold purple outline) has 97% of its area with MSSA 106. This ZIP Code is cross walked to MSSA 106 with some confidence

4. Center Point in MSSA. For those remaining, we check to see if the population weighted center of the ZIP Code is inside a given MSSA and assign a confidence code of 4 (somewhat less confident).

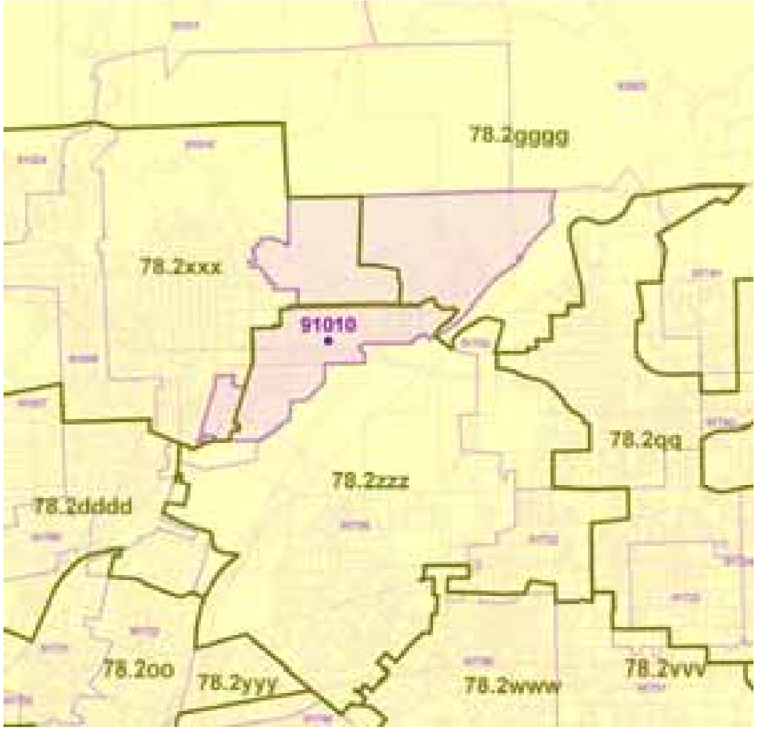


Figure 8 – Center Within. In this example, ZIP 91010 overlaps three MSSA's (78.2xxx, 78.2gggg and 78.2zzz) with roughly equal area. However, the population weighted center (and indeed the majority of the population) fall inside MSSA 78.2zzz and are therefore cross walked to that MSSA with some degree of confidence.

5. First Intersect. For those remaining, we find the first intersection of the ZIP Code polygon with the MSSA. This rarely happens, but mostly does with ZIP Code areas with a zero population. For these we calculate a confidence code of 5 (not confident). This happens very rarely when a ZIP Code area has no population.

A listing of our Confidence codes is contained in Table 1 below.

Table 1 – Structure of Confidence Codes for Crosswalk

Code	Confidence
1	Very Confident[5]
2	Very Confident[6]
3	Somewhat Confident
4	Less Confident
5	Not Confident

[5] Processes 1 and 2 are 100% confident that these ZIP Codes match directly to a single MSSA, so we assign the same confidence code. We distinguish them by applying a different processing Code.

[6] IBID

Stitching Tool into the Processing Framework

Stitching the scripts into the ESRI Geoprocessing framework, is a three step process; 1) Organize the place for the toolbox, script and help, 2) assemble the tool and 3) fill out documentation.

1. Organizing

In this step, we create a Tools directory with three subdirectories for all of our toolsets, not just the Crosswalk function.

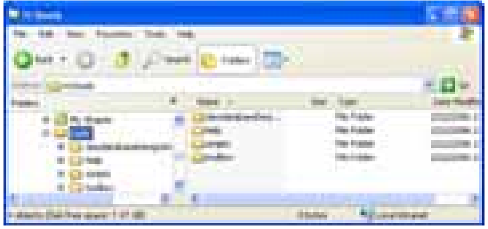


Figure 9 – Tools Directory

The three subdirectories are Help, Scripts and Toolbox. The Help subdirectory contains the exported help files (html) when step 3 is completed. The Scripts directory contains the native Python scripts for all the customized tools. The Toolbox directory contains individual 'tools' of personal GeoDatabases which contain Toolbox objects. These personal geodatabases are separated out for two primary reasons. First these GeoDB's hold only ESRI Toolboxes and no geospatial data, so they cannot be confused with other functions and are easily moved, documented and copied. Second, in an enterprise environment, these toolboxes are available to other staff, as opposed to the 'New Toolbox' method in the Toolbox window in ArcCatalog which places a new toolbox in an individual scratch space[7].

2. Assembling

Assembling the toolset requires a new script to be added to the appropriate Toolbox in one's personal geodatabase, associating that tool with a script in the Scripts directory and setting the properties of that script. To add a script to the toolbox in one's personal geodatabase using ArcCatalog, simply highlight the Toolbox, right click and choose the Add => script function. The resulting menu is shown below and allows a user to give the script a name, label and description[8]. The Second dialog allows one to point to the script in the Script directory. The third dialog requires one to define all of the arguments the script needs. Once this is filled out we have a working dialog in the Geoprocessing framework.

[7] Usually something like C:\Documents and Settings\mbyrne\Application Data\ESRI\ArcToolbox\My Toolboxes\Toolbox (1).tbx

[8] Make sure to check the 'store relative path names box, so the tool knows where the script is.

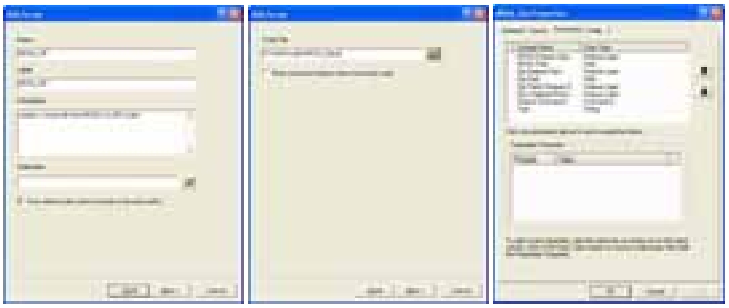


Figure 10 – Adding a Script Dialog

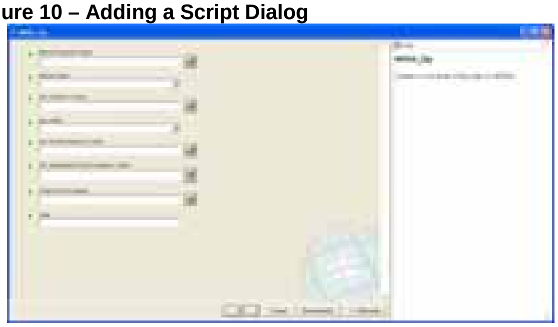


Figure 11 –The functioning Geoprocessing Script as opened from ArcCatolog Toolbox

3. Documenting

By right clicking on the script inside the Toolbox, we can use Edit Documentation to complete the process. The following dialog allows one to create a comprehensive script help file.

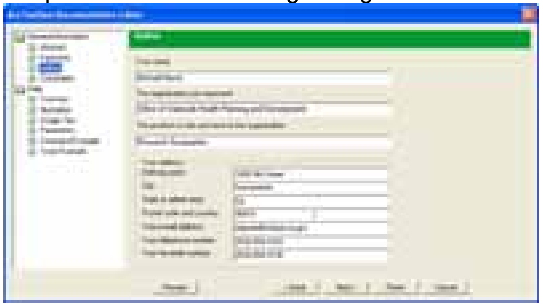
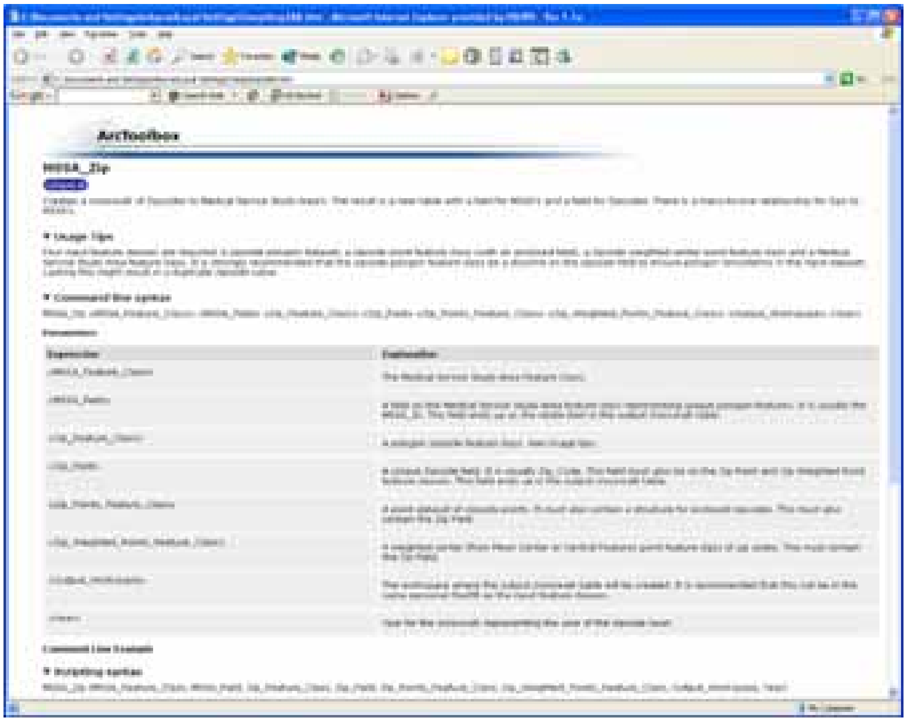


Figure 12 – Script Documentation Dialog

Prior to finishing this dialog, the documentation is exported to an HTML file, which is stored in the 'Help' directory. An example of the help for this script is listed below.



ArcToolbox

MSSA_Zip

[illegible]

▼ Usage Tips

[illegible]

▼ Command line syntax

MSSA_Zip <MSSA_Feature_Class> <MSSA_Field> <Zip_Feature_Class> <Zip_Field> <Zip_Points_Feature_Class> <Zip_Weighted_Points_Feature_Class> <Output_Workspace> <Year>

Parameters

Expression	Explanation
<MSSA_Feature_Class>	The Medical Service Study Area Feature Class.
<MSSA_Field>	[illegible]
<Zip_Feature_Class>	A polygon zipcode feature class. See Usage tips.
<Zip_Field>	[illegible]
<Zip_Points_Feature_Class>	[illegible]
<Zip_Weighted_Points_Feature_Class>	[illegible]
<Output_Workspace>	[illegible]
<Year>	[illegible]

Command Line Example

▼ Scripting syntax

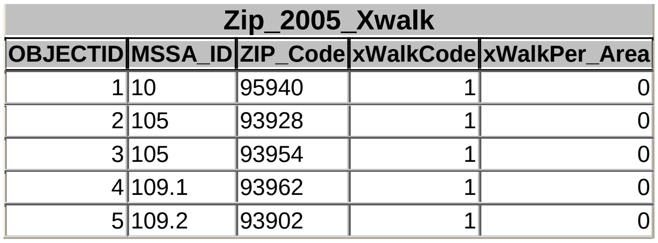
MSSA_Zip (MSSA_Feature_Class, MSSA_Field, Zip_Feature_Class, Zip_Field, Zip_Points_Feature_Class, Zip_Weighted_Points_Feature_Class, Output_Workspace, Year)

Figure 13 – Finished Help File

Results

An example output table is shown below. The resulting table has a unique identifier (ObjectID), a MSSA foreign key (MSSA_ID), a ZIP Code foreign key (ZIP_Code), a Confidence Code (xWalkCode), and for those with greater than 70% area inside a single MSSA, a value of the area overlap (xWalkPer_Area).

Table 2 - Example from 2005 cross walk



Zip_2005_Xwalk

OBJECTID	MSSA_ID	ZIP_Code	xWalkCode	xWalkPer_Area
1	10	95940	1	0
2	105	93928	1	0
3	105	93954	1	0
4	109.1	93962	1	0
5	109.2	93902	1	0

Individual year crosswalk results have tabulated below. The first section (Overall Results) shows the number, percent of the total and additive percent of each Confidence Code match in the ZIP year analysis. This section illustrates the total amount of crosswalks for each Confidence Code. The second section (Results from Area Greater than 70% - Code 3) shows results for only those matches where greater than 70% of the ZIP Code area overlaps an individual MSSA. This section illustrates how much ZIP Code area overlaps a single MSSA for those with greater than 70% area overlap. The final section (ZIP Code Mean Center Distance from MSSA Edge) shows the distance from the mean center of each ZIP Code to the edge of the closest MSSA. This section illustrates the quality of using a single mean center match for those with many overlapping MSSA's.

Table 3 – 2005 Crosswalk Results

Overall Results			
Confidence Code	Number	% of Total	Additive %
1	998	37.18%	37.18%
2	155	5.77%	42.96%
3	1012	37.70%	80.66%
4	519	19.34%	100.00%
5	0	0.00%	62.82%
Total (ZIP Codes)	2684		

Results from Area Greater than 70% - Code 3					
			Range	Count	Percent
Max Area	99.9999996		70% - 75%	94	9.29%
Min Area	70.00		75% - 80%	80	7.91%
STD Area	9.50		80% - 85%	68	6.72%
Mean Area	92.14		85% - 90%	89	8.79%
Median Area	97.64		90% - 95%	102	10.08%
			95 - 100%	579	57.21%
			Total	1012	100%

ZIP Code Mean Center Distance from MSSA Edge				
	All Other Match Codes (1, 2, 3)	Less than 70% Area Match		
		Frontier	Rural	Urban
Count	1,152	19	380	115
Max	53,437	11,616	6,343	12,097
Min	3	35	0	10
Std	4,234	3,638	968	2,062
Mean	3,909	4,260	821	2,455
Median	2,414	3,381	517	2,145

Table 4 - 2004 Crosswalk Results

Overall Results			
Confidence Code	Number	% of Total	Additive %
1	1004	37.21%	37.21%
2	171	6.34%	43.55%
3	1090	40.40%	83.95%
4	429	15.90%	99.85%
5	4	0.15%	100.00%
Total (ZIP Codes)	2698		

Results from Area Greater than 70% - Code 3					
			Range	Count	Percent
Max Area	99.9999998		70% - 75%	82	7.52%
Min Area	70.05		75% - 80%	63	5.78%
STD Area	8.97		80% - 85%	66	6.06%
Mean Area	93.45		85% - 90%	85	7.80%
Median Area	98.68		90% - 95%	91	8.35%
			95% - 100%	703	64.50%
			Total	1090	100.00%

ZIP Code Mean Center Distance from MSSA Edge				
	All Other Match Codes (1, 2, 3)	Less than 70% Area Match		
		Frontier	Rural	Urban
Count	1,237	12	345	72
Max	53,419	1,792	4,836	7,307
Min	4	229	1	28
Std	4,207	2,953	992	1,918
Mean	3,853	3,127	806	2,305
Median	2,347	1,814	494	2,109

Table 5 – 2003 Crosswalk Results

Overall Results			
Confidence Code	Number	% of Total	Additive %
1	1004	37.38%	37.38%
2	172	6.40%	43.78%
3	1074	39.99%	83.77%
4	436	16.23%	100.00%
5	0	0.00%	100.00%
Total (ZIP Codes)	2686		

Results from Area Greater than 70% - Code 3					
			Range	Count	Percent
Max Area	99.9999996		70% - 75%	77	7.17%
Min Area	70.08		75% - 80%	63	5.87%
STD Area	8.91		80% - 85%	61	5.68%
Mean Area	93.56		85% - 90%	85	7.91%
Median Area	98.87		90% - 95%	93	8.66%
			95% - 100%	695	64.71%
			Total	1074	100.00%

ZIP Code Mean Center Distance from MSSA Edge				
	All Other Match Codes (1, 2, 3)	Less than 70% Area Match		
		Frontier	Rural	Urban
Count	1,230	13	347	76
Max	53,419	2,900	5,745	7,306
Min	4	229	1	28
Std	4,210	2,826	999	1,873
Mean	3,852	3,093	808	2,282
Median	2,345	1,836	494	2,029

Table 6 – 2002 Crosswalk Results

Overall Results			
Confidence Code	Number	% of Total	Additive %
1	1013	37.66%	37.66%
2	168	6.25%	43.90%
3	1075	39.96%	83.87%
4	434	16.13%	100.00%
5	0	0.00%	100.00%
Total (ZIP Codes)	2690		

Results from Area Greater than 70% - Code 3					
			Range	Count	Percent
Max Area	99.9999987		70% - 75%	78	7.26%
Min Area	70.07		75% - 80%	65	6.05%
STD Area	8.95		80% - 85%	65	6.05%
Mean Area	93.42		85% - 90%	83	7.72%
Median Area	98.73		90% - 95%	96	8.93%
			95% - 100%	688	64.00%
			Total	1075	100.00%

ZIP Code Mean Center Distance from MSSA Edge				
	All Other Match Codes (1, 2, 3)	Less than 70% Area Match		
		Frontier	Rural	Urban
Count	1,232	13	345	76
Max	53,419	1,836	4,836	7,307
Min	4	15	1	28
Std	4,213	2,956	1,003	1,877
Mean	3,861	2,888	813	2,279
Median	2,349	1,792	493	2,082

Further analysis on a sample to test the validity of the cross walk was performed. A dataset of approximately 35,000 practicing physicians in California were geocoded with street level data. Only those with a match score of 100 (e.g. definitively matched to a known address) were kept and matched to be inside a single MSSA. We used this dataset as the control and then used our crosswalk to assemble to crosswalk predicted MSSA. The results are presented in Table 7. Those rows labeled ‘Matched’ identify those records where our crosswalk delivered the same MSSA as the street level geocode. Those rows with labeled ‘Didn’t Match’ identify those records where our crosswalk delivered a different MSSA than the street level geocode. The predominant error is found in urban Code 4 (ZIP Code weighted population center) at 34.75% of all Code 4’s. The total error rate for this method is 25%.

Table 7 – Control vs. Crosswalk comparison

		Urban		Rural		Frontier		Totals		
		Count	% of Total	Count	% of Total	Count	% of Total	Of Match	Of Code	% of Total
Code 1	Matched	49	92.45%	0	0.00%	0	0.00%	49	92.45%	53
	Didn't Match	4	7.55%	0	0.00%	0	0.00%	4	7.55%	
Code 2	Matched	510	80.19%	120	18.87%	6	0.94%	636	100.00%	636
	Didn't Match	0	0.00%	0	0.00%	0	0.00%	0	0.00%	
Code 3	Matched	13,901	71.87%	2,053	10.61%	68	0.35%	16,022	82.84%	19,342
	Didn't Match	3,232	16.71%	88	0.45%	0	0.00%	3,320	17.16%	
Code 4	Matched	8,903	56.90%	1,056	6.75%	0	0.00%	9,959	63.65%	15,646
	Didn't Match	5,437	34.75%	250	1.60%	0	0.00%	5,687	36.35%	
Totals	Matched	23,363	65.48%	3,229	9.05%	74	0.21%	26,666	74.74%	35,677
	Didn't Match	8,673	24.31%	338	0.95%	0	0.00%	9,011	25.26%	

Discussion

Analysis of Results

The results presented above are representative of a total number of ZIP Codes and not percentages of population. Nonetheless, these results demonstrate an effective crosswalk process. Of the ZIP Codes in California, 42.96% of all ZIPs are 100% within a single MSSA and therefore can positively be matched to a single MSSA. Moreover, an additional 37.70% of the total ZIP Codes have greater than 70% of their area inside a single MSSA. Indeed of these 37.70%, nearly 70% have greater than 90% of their area contained within a single MSSA (Median area within a single MSSA was 97.64% of the total area in any given ZIP

Code boundary). A high degree of certainty can be placed on these matched values. Given the high number of ZIP Codes having this quality combined with the high confidence in the match, we can conclude that this portion of our crosswalk has merit.

Only 19.34% of ZIP Codes are not entirely or mostly (< 70%) contained with a Single MSSA. These are matched to a single MSSA based on the location of the weighted mean population (as show in Figure 8). These remaining 19.34% are further analyzed by measuring how far the weighted mean center of the zip code is to the nearest MSSA edge. Effectively this demonstrates the closeness of being placed in another MSSA. The comparison for those with over 70% area inside one MSSA from those below 70% is split (and of these separated out by Rural/Urban/Frontier). Results from this last analysis demonstrate the areas of greatest concern using our method. A predominant number of these matches of concern are in rural areas (nearly 5 to 1). Moreover, these rural distances show the lowest distance from the mean center to the edge, possibly indicating a probable error when assigning a ZIP Code to a single MSSA. These results point to the continuing need for a more effective method of cross walking these last ZIP Code values to single MSSA's. However, one does need to keep in mind that these mostly rural ZIP Codes are a very small percentage of the total California population, and therefore are not likely to affect seriously the significance of most studies using our crosswalk method.

We ran a sample error analysis using physician data. The result of the error analysis shows us that 75% of the time, our crosswalk is valid while 25% of the time our crosswalk identified the incorrect MSSA. More than half (5,437 of 9,011) of these incorrect matches are from Urban Code 4 process types. This analysis indicates that the mean weighted point protocol might not be the best method for a crosswalk, but delivers a correct match approximately 56% of the time in urban areas. Additional work that could be done in this area, is a business process structure such that for those Code 4 ZIP's a single MSSA is identified from all potential MSSA's intersecting based on a weighted sample from the percent of population inside each MSSA. More analysis needs to be performed to establish these probability scores.

Implications

The California Office of Statewide Health Planning and Development (OSHPD) administers multiple healthcare data collection programs as well as healthcare workforce and community development programs. The Medical Service Study Area units have been used by the Office for three decades, specifically to identify areas of unmet healthcare need and allocate public health funds to those areas. The implementation of geographic information systems (GIS) technologies has substantially enhanced the application of MSSA boundaries to the broader business needs of the Office and to other state health agencies. For example, the geocoded addresses of where primary care physicians practice are being aggregated by MSSA and combined with various census demographic variables to calculate ratios and rates of primary care access in local communities. The California Healthcare Workforce Policy Commission uses these improved assessments to better target limited state funds into those communities that reflect the greatest need. Other health agencies have also embraced the MSSA for research and analysis efforts. The California Cancer Registry has summarized geocoded cancer cases to the MSSA level and prepared analyses of both colorectal and breast cancer diagnoses for local communities. The California Dialogue on Cancer intends to use these results in targeted outreach efforts to increase cancer screening in those communities reporting the highest incidence of late stage diagnosis.

It is clear that the MSSA geographic framework continues to benefit California health programs with an assessment tool and analytic capacity, improving business processes that impact citizens across the state. However, analysis has been hampered since the smallest geographic identifiers available for most health care data sets are ZIP Codes, and other relevant population data are not available for these areas.[9] The transformation methods presented in this paper speak directly to this issue, offering the capacity to port ZIP Code-level data sets to census-based MSSA units that deliver much more robust and meaningful population-based research, analysis, and program administration. Issues of too few patient records per ZIP Code as well as shifting boundaries and populations within ZIP Codes have posed problems throughout the history of the program. The OSHPD plans to implement ZIP Code-to-MSSA cross walks, as the paper describes, to remedy these issues and deliver consistently relevant patient data sets and map products through application of the MSSA geographic framework.

Other key health-related data sets being targeted for transformation to MSSA include vital statistics (birth and death records) and Medi-Cal information, each available publicly at the ZIP Code level.[10] These health indicators can be ported into the MSSA framework and then combined with any number of census demographic variables as denominators in calculating rates, ratios, or basic population comparisons. Likewise, using current-year demographic estimates and future-year projections, health planners and policymakers will have access to rich sets of information and tools that provide a much clearer understanding of the healthcare landscape and empower decision making processes in California.

Next Steps

We propose to further the cross walk structure to more effectively manage a geographic framework for health data in California. We will refine our process to

[9] WWAMI (Washington, Wyoming, Alaska, Montana, & Idaho) Rural Health Research Center, RUCA March 2005, <http://www.fammed.washington.edu/wwamirhrc/rucas/methods.html>.
[10] Vital Statistics of California <http://www.dhs.ca.gov/hisp/chs/OHIR/tables/> and Medical Care Statistics <http://www.dhs.ca.gov/admin/ffdmb/mcss/default.htm>

maximize the confidence in our approach. Finally we will place these crosswalk tables into the Enterprise GIS architecture[11] so that when external users query OSHPD for health data, that data will be delivered on the MSSA geography rather than the ZIP Code geography.

Authors

Michael Byrne
Research Geographer
mbyrne@oshpd.ca.gov
Office of Statewide Health Planning and Development

Scott Christman
Senior Information Systems Specialist
Scott.christman@oshpd.ca.gov
Office of Statewide Health Planning and Development

Appendix

CreateMeanCenter.py

```
#Mike Byrne
#Feb 15, 2006
#CreateMeanCenter.py
#Creates a new feature class which is the mean center inside a singlge feature of
#the Input Feature Class, based on a weighted field value from another feature class.
#(Example: Mean Center of Census Blocks (population) within individual ZipCodes)
# Import system modules
import win32com.client, sys, string, math
# Create the Geoprocessor object
gp = win32com.client.Dispatch("esriGeoprocessing.GpDispatch.1")

#argument variables
inFL = sys.argv[1]          #the Layer you want the centers (eg zipcode")
myField = sys.argv[2]       #field in the inFL which determines unique groupings of polyongs
qryDef = sys.argv[3]        #definition on the inFC, to limit loop; have found a memory lapse at about 250 records
weightFL = sys.argv[4]      #the Layer containing which will determine the centers within individual inFL's
weightField = sys.argv[5]   #field in the weightFL on which the center function will be performed
outWS = sys.argv[6]         #output workspace (best if not the same as in the input)
outFCName = sys.argv[7]     #output FeatureClass
outFC = outWS + "\\" + outFCName
mbFC = outWS + "\\mbtempfc" #temporary file

try:
   if gp.Exists(outFC):
      gp.Delete_management(outFC)
   gp.CreateFeatureClass_management(outWS, outFCName, "POINT", inFL, "", "", gp.Describe(inFL).SpatialReference )
   gp.AddField_management(outFC, "ProcessType", "TEXT", "", "", "5", "", "NULLABLE", "NON_REQUIRED", "")
   gp.AddField_management(outFC, "NumFeat", "LONG", "10", "", "", "", "NULLABLE", "NON_REQUIRED", "")
   if qryDef == "#" :
      sCur = gp.SearchCursor(inFL)  #set up a loop for each polygon feature in the inFL
   else:
      sCur = gp.SearchCursor(inFL, qryDef)
   sRow = sCur.Next()
```

[11] See http://gis.esri.com/library/userconf/health05/docs/pap1056.pdf

```
    cnt = 1
    while sRow <> None:                    #for each polygon feature in the inFL.
        proType = "MC"
        myFeature = sRow.GetValue(myField)  #it is a good idea to dissolve (on the inFL) prior to running this, so you
        flLyr = "fl_" + str(cnt)            #know there is only one 'type' per feature
        wLyr = "w_" + str(cnt)
        if gp.Exists(mbFC):  #delete temporary output files
            gp.Delete(mbFC)
        gp.AddMessage("working on feature " + myFeature + " (" + str(cnt) + ")") #print "working on feature " + myFeature + "
(" + str(cnt) + ")"
        gp.MakeFeatureLayer_management(inFL, flLyr, "[" + myField + "] = '" + myFeature + "'" )
        gp.MakeFeatureLayer_management(weightFL, wLyr, "[" + weightField + "] > 0")
        #gp.AddMessage("  flLyr has " + str(gp.GetCount_management(flLyr)) + " features")
        #gp.AddMessage("  wLyr has " + str(gp.GetCount_management(wLyr)) + " features")
        gp.SelectLayerByLocation_management(wLyr, "HAVE_THEIR_CENTER_IN", flLyr, "", "NEW_SELECTION")
        #gp.AddMessage("  Checking count " + str(gp.GetCount_management(wLyr)))
        if gp.GetCount_management(wLyr) > 0:  #if Get_Count greater than 0, mean center on features
            gp.MeanCenter_stats(wLyr, mbFC, weightField, "", "")
            #check to see if result is inside original poly; if not, centralfeature, changeflag
            mbLyr = "mbLyr_" + str(cnt)
            gp.MakeFeatureLayer_management(mbFC, mbLyr)
            gp.SelectLayerByLocation_management(mbLyr, "HAVE_THEIR_CENTER_IN", flLyr, "", "NEW_SELECTION")
            if gp.GetCount_management(mbLyr) < 1: #then resulting Mean Center falls outside polygon
                gp.AddMessage("    Using Central Feature to process feature... ")
                gp.Delete(mbFC)
                gp.CentralFeature_stats(wLyr, mbFC, "Euclidean Distance", weightField)
                proType = "CF"
            del mbLyr
        else :  #if Get_Count = 0, then nothing to do a central feature on, only possibility
            gp.AddMessage("    finding mean center self ...")
            gp.MeanCenter_stats(flLyr, mbFC)  # is Mean Center on Self; implies hand checking type = MCS, Count = 0
            proType = "MCS"
        #gp.AddMessage("  Adding Fields")
        gp.AddField_management(mbFC, myField, "TEXT", "5", "", "", "", "NULLABLE", "NON_REQUIRED", "")
        gp.AddField_management(mbFC, "ProcessType", "TEXT", "", "", "5", "", "NULLABLE", "NON_REQUIRED", "")
        gp.AddField_management(mbFC, "NumFeat", "LONG", "10", "", "", "", "NULLABLE", "NON_REQUIRED", "")
        #gp.AddMessage("    Updating Fields")
        uCur = gp.UpdateCursor(mbFC)
        uRow = uCur.Next()
        while uRow <> None:
            uRow.Setvalue(myField, myFeature)
            uRow.Setvalue("ProcessType",proType)
            uRow.Setvalue("NumFeat",gp.GetCount_management(wLyr))
            uCur.UpdateRow(uRow)
            uRow = uCur.Next()
        #gp.AddMessage("  Appending ...")
        gp.Append_management(mbFC, outFC, "TEST")
        del uRow, uCur, myFeature, flLyr, wLyr, proType
        cnt = cnt + 1
        #gp.AddMessage("Made it to the Bottom of the loop")
        sRow = sCur.Next() #next Feature

    if gp.Exists(mbFC):   #delete temporary output files
        gp.Delete(mbFC)
    del cnt, sRow, sCur, myField, proType
    del mbFC, outFC, weightField, qryDef
    del inFL, weightFL, outWS, outFCName
    del gp
except:
    gp.AddMessage("MB error is " + gp.GetMessages())
```

MSSA_zip.py

Formatted: English (U.S.)

Formatted: English (U.S.)

```
#Mike Byrne
#Feb. 15, 2006
#MSSA_zip.py
##Creates a crosswalk of Zipcodes to MSSA's.  The result is a new table with
##an field for MSSA's and a field for Zipcodes.  There is a many-to-one
```

```
##relationship for Zips to MSSA's.

# Import system modules
import win32com.client, sys, math, string, os
# Create the Geoprocessor object
gp = win32com.client.Dispatch("esriGeoprocessing.GpDispatch.1")
# Load required toolboxes...
#gp.AddToolbox("D:/Program Files/ArcGIS/ArcToolbox/Toolboxes/Data Management Tools.tbx")
#gp.AddToolbox("D:/Program Files/ArcGIS/ArcToolbox/Toolboxes/Analysis Tools.tbx")

#local input dataset variables
#inWS = "D:\\data\\zip\\ziptest.mdb"
inMSSA = sys.argv[1]
#inMSSA = inWS + "\\MSSA106" #"MSSA2000"
MSSAField = sys.argv[2]
#MSSAField = "MSSA_ID"
inZip = sys.argv[3]
#inZip = inWS + "\\Zip"
zipField = sys.argv[4]
#zipField = "Zip_Code"
inZipPoints = sys.argv[5]
#inZipPoints = inWS + "\\Zipcode_Points"
inZipWP = sys.argv[6]
#inZipWP = inWS + "\\ZipMeanCenter"
outWS = sys.argv[7]
#outWS = "D:\\data\\zip\\zipoutput.mdb" #zipoutput.mdb
year = sys.argv[8]
#year = "2004"

#create output and initialize fields
outTab = "Zip_" + year + "_Xwalk"
outTable = outWS + "\\Zip_" + year + "_Xwalk"

if gp.Exists(outTable):
    gp.Delete_management(outTable)
gp.CreateTable_management(outWS, outTab)
#test with creating inside current personal geodatabase; e.g. is there a record lock write error with an insert cursor
#then test if exists, delete and create
field1 = MSSAField
field2 = "Zip" + year
field3 = "Zip" + year + "Code"
field4 = "Zip" + year + "Per_Area"
field5 = "mbXWalkFlag"
field6 = "mbXWalkArea"
gp.AddField_management(outTable, field1, "TEXT", "", "", 10)
gp.AddField_management(outTable, field2, "TEXT", "", "", 5)
gp.AddField_management(outTable, field3, "SHORT")
gp.AddField_management(outTable, field4, "Double")
##create some temporary fields, which we will delete later.  Make sure the don't
##exist prior to creating them.  if they do, delete them
myFields = gp.ListFields(inZip, "mbXWalk*")
myField = myFields.Next()
while myField:
    gp.DeleteField_management(inZip, field.Name)
    myField = myFields.Next()
del myField, myFields
gp.AddField_management(inZip, field5, "Short")
gp.AddField_management(inZip, field6, "Double")
gp.CalculateField(inZip, field5, "0")
gp.CalculateField(inZip, field6, "[Shape_Area]")

#process step #1 - Enclosed zip codes
calcValue = 1
cnt = 1
s1Cur = gp.SearchCursor(inMSSA)
s1Row = s1Cur.Next()
while s1Row <> None:
    MSSA_ID = s1Row.GetValue(MSSAField)
    gp.AddMessage("Process step #1 - working on MSSA_ID " + MSSA_ID) #print "Process step #1 - working on
MSSA_ID " + MSSA_ID
```

```
    MSSALyr = "MSSA" + str(cnt)
    qry = "[MSSA_ID] = '" + MSSA_ID + "'"
    gp.MakeFeatureLayer_management(inMSSA, MSSALyr, qry)
    ZipLyr = "zip" + str(cnt)
    zqry = "[ENC_ZIP] <> ''"
    gp.MakeFeatureLayer_management(inZipPoints, ZipLyr, zqry)
    gp.SelectLayerByLocation_management(ZipLyr, "COMPLETELY_WITHIN", MSSALyr , "", "NEW_SELECTION")
    ss1Cur = gp.SearchCursor(ZipLyr)
    ss1Row = ss1Cur.Next()
    while ss1Row <> None:
        gp.AddMessage("    Inserting zipcode value " + ss1Row.GetValue(zipField)) # "    Inserting zipcode value " +
ss1Row.GetValue(zipField)
        i1Cur = gp.InsertCursor(outTable)
        i1Row = i1Cur.NewRow()
        i1Row.SetValue(MSSAField,MSSA_ID)
        i1Row.SetValue(field2,ss1Row.GetValue(zipField))
        i1Row.SetValue(field3,calcValue)
        i1Cur.InsertRow(i1Row)
        del i1Row, i1Cur
        ss1Row = ss1Cur.Next()
    del MSSALyr, ZipLyr, zqry, qry, ss1Row, ss1Cur, MSSA_ID
    cnt = cnt + 1
    s1Row = s1Cur.Next()
del s1Row, s1Cur

#process step #2 - Zip codes entirely within single MSSA's
calcValue = 2
s2Cur = gp.SearchCursor(inMSSA)
s2Row = s2Cur.Next()
while s2Row <> None:
    MSSA_ID = s2Row.GetValue(MSSAField)
    gp.AddMessage("Process step #2 - working on MSSA_ID " + MSSA_ID) # "Process step #2 - working on MSSA_ID " +
MSSA_ID
    MSSALyr = "MSSA" + str(cnt)
    qry = "[MSSA_ID] = '" + MSSA_ID + "'"
    gp.MakeFeatureLayer_management(inMSSA, MSSALyr, qry)
    ZipLyr = "zip" + str(cnt)
    gp.MakeFeatureLayer_management(inZip, ZipLyr)
    gp.SelectLayerByLocation_management(ZipLyr, "COMPLETELY_WITHIN", MSSALyr , "", "NEW_SELECTION")
    u2Cur = gp.UpdateCursor(ZipLyr)
    u2Row = u2Cur.Next()
    while u2Row <> None:
        gp.AddMessage("    Inserting zipcode value " + u2Row.GetValue(zipField)) #print "    Inserting zipcode value " +
u2Row.GetValue(zipField)
        i2Cur = gp.InsertCursor(outTable)
        i2Row = i2Cur.NewRow()
        i2Row.SetValue(MSSAField,MSSA_ID)
        i2Row.SetValue(field2,u2Row.GetValue(zipField))
        i2Row.SetValue(field3,calcValue)
        i2Cur.InsertRow(i2Row)
        #update the flag field on the ZipLyr
        u2Row.SetValue(field5,1)
        u2Cur.UpdateRow(u2Row)
        del i2Row, i2Cur
        u2Row = u2Cur.Next()
    del MSSALyr, ZipLyr, qry, u2Row, u2Cur, MSSA_ID
    cnt = cnt + 1
    s2Row = s2Cur.Next()
del s2Row, s2Cur

#process step #3 - Zip codes with most of their area inside a single MSSA
calcValue = 3
s3Cur = gp.SearchCursor(inMSSA)
s3Row = s3Cur.Next()
while s3Row <> None:
    MSSA_ID = s3Row.GetValue(MSSAField)
    gp.AddMessage("Process step #3 - working on MSSA_ID " + MSSA_ID) #print "Process step #3 - working on
MSSA_ID " + MSSA_ID
    MSSALyr = "MSSA" + str(cnt)
    qry = "[MSSA_ID] = '" + MSSA_ID + "'"
    gp.MakeFeatureLayer_management(inMSSA, MSSALyr, qry)
    ZipLyr = "zip" + str(cnt)
    zqry = "[" + field5 + "] = 0"
    gp.MakeFeatureLayer_management(inZip, ZipLyr, zqry)
    outCl = outWS + "\\mbtmpClip"
    if gp.Exists(outCl):
        gp.Delete_management(outCl)
    gp.Clip_analysis(ZipLyr, MSSALyr, outCl)
    ss3Cur = gp.SearchCursor(outCl)
    ss3Row = ss3Cur.Next()
    while ss3Row <> None:
        gp.AddMessage("  evaluating zip_code " + ss3Row.GetValue(zipField)) #print "  evaluating zip_code " +
ss3Row.GetValue(zipField)
        #evaluate the previous area (in mbArea field) vs. the current area.  if current > 70% previous, then go
        if ss3Row.Shape.Area > (ss3Row.GetValue(field6) * .7):
            gp.AddMessage("    Inserting zipcode value " + ss3Row.GetValue(zipField)) #print "    Inserting zipcode value " +
ss3Row.GetValue(zipField)
            i3Cur = gp.InsertCursor(outTable)
            i3Row = i3Cur.NewRow()
            i3Row.SetValue(MSSAField,MSSA_ID)
            i3Row.SetValue(field2,ss3Row.GetValue(zipField))
            i3Row.SetValue(field3,calcValue)
            i3Row.SetValue(field4, (ss3Row.Shape.Area / ss3Row.GetValue(field6) * 100) )
            i3Cur.InsertRow(i3Row)
            u3Cur = gp.UpdateCursor(inZip, "[Zip_Code] = '" + ss3Row.GetValue(zipField) + "'")
            u3Row = u3Cur.Next()
            while u3Row <> None:
                u3Row.SetValue(field5,1)
                u3Cur.UpdateRow(u3Row)
                u3Row = u3Cur.Next()
            del u3Row, u3Cur, i3Row, i3Cur
        ss3Row = ss3Cur.Next()
    del zqry, qry, MSSALyr, ZipLyr, ss3Row, ss3Cur, MSSA_ID
    cnt = cnt + 1
    s3Row = s3Cur.Next()
if gp.Exists(outCl):
    gp.Delete_management(outCl)
del s3Row, s3Cur, outCl

#process step #4 - Weighted Zipcode centers inside MSSA's
calcValue = 4
s4Cur = gp.SearchCursor(inMSSA)
s4Row = s4Cur.Next()
while s4Row <> None:
    MSSA_ID = s4Row.GetValue(MSSAField)
    gp.AddMessage("Process step #4 - working on MSSA_ID " + MSSA_ID) #print "Process step #4 - working on
MSSA_ID " + MSSA_ID
    MSSALyr = "MSSA" + str(cnt)
    qry = "[MSSA_ID] = '" + MSSA_ID + "'"
    gp.MakeFeatureLayer_management(inMSSA, MSSALyr, qry)
    ZipLyr = "zip" + str(cnt)
    zqry = "[" + field5 + "] = 0"
    gp.MakeFeatureLayer_management(inZip, ZipLyr, zqry)
    ZipWPLyr = "zipWeigthePointLayer" + str(cnt)
    gp.MakeFeatureLayer_management(inZipWP, ZipWPLyr)  #create layer of population weighted points
    gp.SelectLayerByLocation_management(ZipWPLyr, "COMPLETELY_WITHIN", ZipLyr,"", "NEW_SELECTION")
    gp.SelectLayerByLocation_management(ZipWPLyr, "COMPLETELY_WITHIN", MSSALyr,"", "SUBSET_SELECTION")
    ss4Cur = gp.SearchCursor(ZipWPLyr)
    ss4Row = ss4Cur.Next()
    while ss4Row <> None:
        gp.AddMessage("    insertting zip_code " + ss4Row.GetValue(zipField)) #print "    insertting zip_code " +
ss4Row.GetValue(zipField)
        i4Cur = gp.InsertCursor(outTable)
        i4Row = i4Cur.NewRow()
        i4Row.SetValue(MSSAField,MSSA_ID)
        i4Row.SetValue(field2,ss4Row.GetValue(zipField))
        i4Row.SetValue(field3,calcValue)
        i4Cur.InsertRow(i4Row)
        u4Cur = gp.UpdateCursor(inZip, "[Zip_Code] = '" + ss4Row.GetValue(zipField) + "'")
        u4Row = u4Cur.Next()
        while u4Row <> None:
            u4Row.SetValue(field5,1)
            u4Cur.UpdateRow(u4Row)
            u4Row = u4Cur.Next()
        del i4Row, i4Cur, u4Row, u4Cur
        ss4Row = ss4Cur.Next()
    del zqry, qry, MSSALyr, ZipLyr, ZipWPLyr, ss4Row, ss4Cur, MSSA_ID
    cnt = cnt + 1
    s4Row = s4Cur.Next()
del s4Row, s4Cur

#process step #5 - Intersect method, for any not meeting the above; crecent mean centers and 0 pop centers do this
calcValue = 5
s5Cur = gp.SearchCursor(inMSSA)
s5Row = s5Cur.Next()
while s5Row <> None:
    MSSA_ID = s5Row.GetValue(MSSAField)
    gp.AddMessage("Process step #5 - working on MSSA_ID " + MSSA_ID) #print "Process step #5 - working on
MSSA_ID " + MSSA_ID
    MSSALyr = "MSSA" + str(cnt)
    qry = "[MSSA_ID] = '" + MSSA_ID + "'"
    gp.MakeFeatureLayer_management(inMSSA, MSSALyr, qry)
    ZipLyr = "zip" + str(cnt)
    zqry = "[" + field5 + "] = 0"
    gp.MakeFeatureLayer_management(inZip, ZipLyr, zqry)
    gp.SelectLayerByLocation_management(ZipLyr, "INTERSECT", MSSALyr,"", "NEW_SELECTION")
    u5Cur = gp.UpdateCursor(ZipLyr)
    u5Row = u5Cur.Next()
    while u5Row <> None:
        gp.AddMessage("    insertting zip_code " + u5Row.GetValue(zipField)) #print "    insertting zip_code " +
u5Row.GetValue(zipField)
        i5Cur = gp.InsertCursor(outTable)
        i5Row = i5Cur.NewRow()
        i5Row.SetValue(MSSAField,MSSA_ID)
        i5Row.SetValue(field2,u5Row.GetValue(zipField))
        i5Row.SetValue(field3,calcValue)
        i5Cur.InsertRow(i5Row)
        u5Row.SetValue(field5,1)
        u5Cur.UpdateRow(u5Row)
        del i5Row, i5Cur
        u5Row = u5Cur.Next()
    del zqry, qry, MSSALyr, ZipLyr, u5Row, u5Cur, MSSA_ID
    cnt = cnt + 1
    s5Row = s5Cur.Next()
del s5Row, s5Cur

gp.AddMessage("Finished excecuting, cleaning variables ...") #print "Finished excecuting, cleaning variables ..."
del outTable, outTab, calcValue
fields = gp.ListFields(inZip, "mbXWalk*")
field = fields.Next()
while field:
    #print field.Name
    gp.DeleteField_management(inZip, field.Name)
    field = fields.Next()
del field, fields
del field6, field5, field4, field3, field2, field1, MSSAField, zipField
del year, inZipWP, inZipPoints, inZip, inMSSA
del outWS
#del gp
```