acs.R: An R Package for Neighborhood-Level Data from the U.S. Census

Ezra Haber Glenn, AICP
Lecturer in Community Development
Massachusetts Institute of Technology
Department of Urban Studies and Planning

Draft, presented July 6, 2011

Abstract

Over the past decade, the U.S. Census Bureau has implemented the American Community Survey as a replacement for its traditional decennial "long-form" survey. This year—for the first time ever—ACS data was made available at the census tract and block group level for the entire nation, representing geographies small enough to be useful to local planners; in the future these estimates will be updated on a yearly basis, providing much more current data than was ever available in the past. Although the ACS represents a bold strategy with great promise for planners working at the neighborhood scale, it will require them to become comfortable with statistical techniques and concerns that they have traditionally been able to avoid.

To help with this challenge the author has been working with local-level planners to determine the most common problems associated with using ACS data, and has implemented these functions as a package in the R statistical programming language. The effort is still in a "beta" stage, with much work to be done, but the basic framework is in place. The package defines a new "acs" class object (containing estimates, standard errors, and metadata for tables from the ACS), with methods to deal appropriately with common tasks (e.g., combining subgroups or geographies, mathematical operations on estimates, tests of significance, plots of confidence intervals, etc.).

Keywords: census, demographics, R, ACS, sampling

The Dawn of the ACS; the Nature of Estimates

Every ten years, the U.S. Census Bureau undertakes a complete count of the country's population, or at least attempts to do so; that's what a census is. The information they gather is very limited: this is known as the Census "short form," which consists of only six questions on sex, age, race, and household composition. This paper has nothing to do with that.

Department of Urban Studies and Planning, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Room 7-337; eglenn@mit.edu; 617.253.2024.

Starting in 1940, along with this complete enumeration of the population, the Census Bureau began gathering demographic data on a wide variety of additional topics—everything from income and ethnicity to education and commuting patterns; in 1960 this effort evolved into the "long form" survey, administered to a smaller sample of the population (approximately one in six) and reported in summary files.[1] From that point forward census data was presented in two distinct formats: actual numbers derived from complete counts for some data (the "SF-1" and "SF-2" 100% counts), and estimates derived from samples for everything else (the "SF-3" tables). For most of this time, however, even the estimates were generally treated as counts by both planners and the general public, and outside of the demographic community not much attention was paid to standard errors and confidence intervals.

Starting as a pilot in 2000, and implemented in earnest by mid-decade, the American Community Survey (ACS) has now replaced the Census long-form survey, and provides almost identical data, but in a very different form. The idea behind the ACS—known as "rolling samples" (Alexander, 2001)—is simple: rather than gather a one-in-six sample every ten years, with no updates in between, why not gather much smaller samples every month on an ongoing basis, and aggregate the results over time to provide samples of similar quality? The benefits include more timely data as well as more care in data collection (and therefore a presumed reduction in non-sampling errors); the downside is that the data no longer represent a single point in time, and the estimates reported are derived from much smaller samples (with much larger errors) than the decennial long-form. One commentator describes this situation elegantly as "Warmer (More Current) but Fuzzier (Less Precise)" than the long-form data (MacDonald, 2006); another compares the old long-form to a once-in-a-decade "snapshot" and the ACS to a ongoing "video," noting that a video allows the viewer to look at individual "freeze-frames," although they may be lower resolution or too blurry—especially when the subject is moving quickly (Alexander, 2002).

To their credit, the Census Bureau has been diligent in calling attention to the changed nature of the numbers they distribute, and now religiously reports margins of error along with all ACS data. Groups such as the National Research

[1] These were originally known as "summary tape files."

Council have also stressed the need to increase attention to the nature of the ACS (Citro & Kalton, 2007), and in recent years the Census Bureau has increased their training and outreach efforts, including the publication of an excellent series of "Compass" reports to guide data users (U.S. Census Bureau, 2009) and additional guidance on their "American FactFinder" website. Unfortunately, the inclusion of all these extra numbers still leaves planners somewhat at a loss as to how to proceed: when the errors were not reported we felt we could ignore them and treat the estimates as counts; now we have all these extra columns in everything we download, without the tools or the perspective to know how to deal with them. To resolve this uncomfortable situation and move to a more productive and honest use of ACS data, we need to take a short detour into the peculiar sort of thing that is an estimate.

Challenges of Estimates in General

The peculiar sort of thing that is an estimate. Contrary to popular belief, estimates are strange creatures, quite unlike ordinary numbers. As an example: if I count the number of days between now and when a draft of this paper is due to the conference organizers, I may discover that I have exactly eight days left to write it: that's an easy number to deal with, whether or not I like the reality it represents. If, on the other hand, I *estimate* that I still have another three days of coding to work through before I can write up the last section, then I am dealing with something different: how confident am I that three days will be enough? Could the coding take as many as *five* days? More? Is there any chance it could be done in fewer? (Ha!)

Add to this the complexity of combining *multiple* estimates—for example, if I suspect that "roughly half" of the code I am developing will need to be checked by a demographer friend, and I also need to complete grading for my class during this same period, which will probably require "around two days of work"—and you begin to appreciate the strange and bizarre ways we need to bend our minds to deal with estimates.

When faced with these issues, people typically do one of two things. The most obvious, of course, is to simply treat estimates like real numbers and ignore the fact that they are really something different. A more epistemologically-honest approach is to think of estimates as "fuzzy numbers," which jibes well with the latest philosophical leanings. Unfortunately, the first of these is simply wrong, and the second is mathematically unproductive. Instead, I prefer to think of estimates as "two-dimensional numbers"—they represent complex little *probability distributions* that spring to life to describe our state of knowledge (or our relative lack thereof). When the estimates are the result of random sampling—as is the case with surveys such as the ACS—these distributions are well understood, and can be easily and efficiently described with just two (or, for samples of small n, three) parameters.

In fact, although the "dimensional" metaphor here may be new, the underlying concept is exactly how statisticians typically treat estimates: we think of *distributions* of maximum likelihood, and describe them in terms of both a center (often confusingly called "the" estimate) and a spread (typically the standard error or margin of error); the former helps us locate the distribution somewhere on the number line and the latter defines the curve around that point. An added advantage of this technique is that it provides a hidden translation (or perhaps a *projection*) from two-dimensions down to a more comfortable one: instead of needing to constantly think about the entire distribution around the point, we are able to use a shorthand, envisioning each estimate as a single point surrounded by the safe embracing brackets of a given confidence interval.

So far so good, until (as noted above), it comes time to *combine* estimates in some way. For this, the underlying mathematics requires that we forego the convenient metaphor of flattened projections and remember that these numbers really do have two-dimensions; to add, subtract, or otherwise manipulate them we must do so up in that 2-D space—quite literally—by squaring the standard errors and working with *variances*. (Of course, once we are done with whatever we wanted to do, we get back down onto the safe flat number line with a dimension-clearing square root.)

Dealing with estimates in ACS data. All that we have said about estimates in general, of course, applies to the ACS in particular. The ACS provides an unprecedented amount of data of particular value for planners working at the local level, but brings with it certain limitations and added complexities. As a result, when working with these estimates, planners find that a number of otherwise straightforward tasks become quite daunting, especially when one realizes that these problems—and those in the following section on multi-year estimates—can all occur in the same basic operation (see Section on the facing page).

In order to *combine* estimates—for example, to aggregate Census tracts into the neighborhoods or to merge subcategories of variables ("Children under age 5", "Children 5-9 yrs.", "Children 10-12 yrs.", etc.) into larger, more meaningful groups—planners must add a series of estimates and *also* calculate the standard error for the sum of these estimates (approximated by the square root of the sum of the squared standard errors for each estimate): $SE_{\hat{A}+\hat{B}} \approx \sqrt{SE_{\hat{A}}{}^2 + SE_{\hat{B}}{}^2}$. The same is true for subtraction: $SE_{\hat{A}-\hat{B}} \approx \sqrt{SE_{\hat{A}}{}^2 + SE_{\hat{B}}{}^2}$, an important fact when calculating t-statistics to compare differences across geography or change over time. A different set of rules applies for multiplying and dividing standard errors, with added complications related to how the two estimates are related (one formula for dividing when the numerator is a *subset* of the denominator, as is true for calculating proportions, and a different formula when it is not, for ratios and averages). As a result, even simple arithmetic become complex when dealing with estimates derived from ACS samples.

Challenges of Multi-Year Estimates in Particular

In addition to these problems involved in using sample estimates and standard errors, the "rolling" nature of the ACS forces local planners to consider a number of additional issues related to the process of deriving estimates from multi-year samples.

Adjusting for inflation. Although it is collected every month on an ongoing basis, ACS data is only reported once a year, in updates to the 1-, 3-, and 5-year products. Internally, figures are adjusted to address seasonal variation before being combined, and then all dollar-value figures are adjusted to represent real dollars *in the latest year of the survey.* Thus, when comparing dollar-value data from the 2006–2008 survey with data from the 2007–2009 survey, users must keep in mind that the they are comparing apples to oranges (or at least 2008-priced apples to 2009-priced apples), and the adjustment is not always as intuitive as one might assume: although the big difference between these two surveys would seem to be that one contains data from 2006 and the other contains data from 2009—they both contain the same data from 2007 and 2008—this is not entirely true, since the latter survey has updated *all* the data to be in "2009 dollars." When making comparisons, then, planners must note the end years for both surveys and convert one to the other.

Overlapping errors. Another problem when comparing ACS data across time periods stems from a different aspect of this overlap: looking again at these two three-year surveys (2006–2008 vs. 2007–2009), we may be confronted with a situation in which the data being compared is identical in all ways except for the year (i.e., we are looking at the exact same variables from the exact same geographies). In such a case, the fact that that the data from 2007 and 2008 is present in both sets means that we might be *underestimating* the difference between the two if we don't account for this fact: the Census Bureau recommends that the standard error of a difference-of-sample-means be multiplied by $\sqrt{(1 - C)}$, where C represents the percentage of overlapping years in the two samples (U.S. Census Bureau, 2009, p. A-20); in this case, the standard error would thus be corrected by being multiplied by $\sqrt{(1 - \frac{2}{3})} = .577$, almost *doubling* the t-statistic of any observed difference.

At the same time, if we are comparing, say, one location or indicator in the first time period with a different location or indicator in the second, this would not be the case, and an adjustment would be inappropriate.

Additional Issues in Using ACS Data

In addition to those points described about, there are a few other peculiarities in dealing with ACS data, mostly related to "hiccups" with the implementation of the sampling program in the first few years.

Group quarters. Prior to 2006, the ACS did not include group quarters in its sampling procedures.[2] As a result, comparisons between periods that span this time period may under- or over-estimate certain populations. For example, if a particular neighborhood has a large student dormitory, planners may see a large increase in the number of college-age residents—or residents without cars, etc.—when comparing data from 2005 and 2006 (or, say, when comparing data from the 2005–2007 ACS and the 2006–2008 ACS). Unfortunately, there is no simple way to address this problem, other than to be mindful of it.

What do we mean by 90%?. Because the ACS reports "90% margins of error" and not standard errors in raw form, data users must manually convert these figures when they desire confidence intervals of different levels. Luckily, this is not a difficult operation: all it requires is that one divide the given margin of error by the appropriate z-statistic (traditionally 1.645, representing 90% of the area under a standard normal curve), yielding a standard error, which can be then multiplied by a *different* z-statistic to create a new margin of error.

Unfortunately, in the interest of simplicity, the "90%" margins of error reported in the early years of the ACS program were actually computed using a z-statistic of 1.65, not 1.645. Although this is not a huge problem, it is recommended that users remember to divide by this different factor when recasting margins of error from 2005 or earlier (U.S. Census Bureau, 2009).

The problem of medians, averages, percentages, and other non-count units. Another issue that often arises when dealing with ACS data is how to aggregate non-count data, especially when medians, averages, or percentages are reported. (Technically speaking, this is a problem related to all summary data, not just ACS estimates, but it springs up in the same place as dealing with standard errors, when planners attempt to combine ACS data from different geographies or to add columns.) The ACS reports both an estimate and a 90% margin of error for all different types of data, but different types must be dealt with differently. When data is in the form of averages, percentages, or proportions—all the results of some prior process of division—the math can become rather tricky, and one really needs to build up the new estimates from the underlying counts; when working with medians, this technically requires second-order statistical estimations of the shapes of the distribution around the estimates medians.

Putting it All Together: a brief example

As a brief example of the complexity involved with these sort of manipulations, consider the following:

[2] "Group quarters" are defined as "a place where people live or stay, in a group living arrangement, that is owned or managed by an entity or organization providing housing and/or services for the residents.... Group quarters include such places as college residence halls, residential treatment centers, skilled nursing facilities, group homes, military barracks, correctional facilities, and workers dormitories."

A planner working in the city of Lawrence, MA, is assembling data on two different neighborhoods, known as the "North Common" district and the "Arlington" district. In order to improve the delivery of translation services for low-income senior citizens in the city, the planner would like to know which of these two neighborhoods has a higher percentage of residents who are age 65 or over and speak English "not well" or "not at all".

Luckily, the ACS has data on this, available at the census tract level from the 2005–2009 dataset on Table B16004 ("Age By Language Spoken At Home By Ability To Speak English For The Population 5 Years And Over"). For starters, however, the planner will need to combine a few columns—the numerator she wants is the sum of those elderly residents who speak English "not well" and "not at all", and the ACS actually breaks each of these down into four different linguistic sub-categories ("Speak Spanish", "Speak other Indo-European languages", "Speak Asian and Pacific Island languages", and "Speak other languages"). So for each tract she must combine values from $2 \times 4 = 8$ columns—each of which must be treated as a "two-dimensional number" and dealt with accordingly: given the number of tracts, that's $8 \times 3 = 24$ error calculations for each of the two districts.

Once that is done, the next step is to aggregate the data (the combined numerators and also the group totals to be used as denominators) for the three tracts in each district, which again involves working with both estimates and standard errors and the associated rules for combining them: this will require $4 \times 3 = 12$ more error terms. The actual conversion from a numerator (the number of elderly residents in these limited-English categories) and a denominator (the total number of residents in the district) into a proportion involves yet another trick of "two-dimensional" math for each district, yielding—after two more steps—a new estimate with a new standard error.[3] And then finally, the actual test for significance between these two district-level percentages represents one last calculation—a difference of means—to combine these kinds of numbers.

In all, even this simple task required $(24 \times 2) + (12 \times 2) + 2 + 1 = 75$ individual calculations on our estimate-type data, each of which is far more involved than what would be required to deal with non-estimate numbers. (Note that to compare these numbers with the same data from two years earlier to look for significant change would involve the same level of effort all over, with the added complications mentioned in Section on the preceding page.) And while none of this work is particularly difficult—nothing harder than squares and square roots—it can get quite tedious, and the chance of error really increases with the number of steps: in short, this would seem to be an ideal task for a computer rather than a human.

Census Data and the R Project

Next we turn to the use of the R programming language to create just such a tool. In recent years, the R statistical package has emerged as the leading open-source alternative to applications such as SPSS, Stata, and SAS. In some fields—notably biological modeling and econometrics—R is becoming more widely used than commercial competitors, due in large part to the open source development model which allows researchers to collaboratively design custom-built packages for niche applications. Unfortunately, one area of application that has not been as widely explored—despite the potential for fruitful development—is the use of R for demographic analysis for urban planning.[4]

Based on a collaborative development model, the `acs.R` package is the result of work with local planners, students, and other potential data-users. Through conversations with planning practitioners, observation at conferences and field trainings, and research on both Census resources and local planning efforts that make use of ACS data we have identified a short-list of features for inclusion in the package, including functions to help download, explore, summarize, manipulate, analyze, and present ACS data at the neighborhood scale.

In passing, it should be noted that most local planning offices are still a long way from using R for statistical work, whether Census-based or not, and the learning curve is probably too steep to expect much change simply as a result of one new package. Nonetheless, one goal in developing `acs.R` is that over time, if the R project provides more packages designed for common tasks associated with neighborhood planning, eventually more planners at the margin (or perhaps in larger offices with dedicated data staff) may be willing to make the commitment to learn these tools (and possibly even help develop new ones).

Implementation of `acs.R`

As with most R packages, at the core of `acs.R` is the definition of a new `acs` class to hold data from the ACS and a set of associated functions for the proper handling of these objects. Consistent with the guidance of the R Core Development Team (Chambers, 1998, 2006), `acs.R` implements "S4" methods, representing a more structured approach closer to a "true" object-oriented programming than the earlier "S3" methods. Importantly, although some have argued that a strength of R is the ability to use the S3 style as a "quick-and-dirty" object-oriented approach for simple tasks (Hankin, 2007), in this case a more rigorous approach helps insist that ACS estimates (and indeed, any survey estimates) represent, as we have noted, "a different sort of creature from normal numbers," taking a special form (as dictated by the `acs` class) which can only be manipulated or compared using

[3] Note, also, that these steps must be done in the correct order: a novice might first compute the tract-level proportions, and then try to sum them, in violation of the points made in section on the previous page.

[4] There are a few R packages to bridge the gap between GIS and statistical analysis (Bivand, Pebesma, & Gómez-Rubio, 2008) and one contribution to help with the downloading and management of spatial data and associated datasets from the 2000 Census (Almquist, 2010), but to my knowledge none work with ACS data or address the types of issues raised above.

special techniques (as controlled by the appropriate methods).

The acs *class*

Class definition. All acs objects contain the following nine slots:

end.year a single integer value indicating the last year included in the dataset (e.g., 2009 for data from the 2005–2009 ACS)

span a single integer value representing the number of years the dataset spans (e.g., 3 for data from the 2007–2009 ACS)

geography a dataframe containing four variables extracted from the data's geographic header: the geographic place names, the summary level, and two numeric identifiers

acs.colnames a vector of character strings giving the variable names for each column

modified a single logical flag to indicated whether the object has been modified since construction

acs.units a vector containing categorical labels (factors, in R-speak) designating the type of units in each column (e.g., count or percentage or dollars)

currency.year a single integer value indicating the year that all currency values have been adjusted to (by default the same as end.year, but able to be modified by the user for comparisons)

estimate a matrix holding the reported ACS estimates

standard.error a matrix holding the calculated values of the standard errors for each estimate, derived from the reported 90% confidence intervals

The contents of any slot can be accessed via functions named for these slots: estimate(my.data) returns a matrix of the estimates, and currency.year(my.data) returns the year the dollar-values have been adjusted to.

Creation of objects. As with any S4 class, acs objects can be created by hand using the new(class="acs", ...) function, but most will be created instead using "convenience" functions or via the manipulation, combination, or subsetting of previously-created acs objects. For starters, the package provides a basic read.acs() function, which reads downloaded data from an American FactFinder query and parses the results to create a new acs object. At present, this function is fairly basic, but eventual improvements will include the ability to unzip saved results as well as to download data directly from the Census via the R session.

An additional "helper" function, acs.change.colnames(), allows users to change the column names associated with an acs object, either through a single command or via an interactive session, which addresses a major problem identified with dealing with downloaded Census data: default variable names are either far too descriptive (e.g., "Universe: POPULATION 5 YEARS AND OVER: 65 years and over; Speak Asian and Pacific Island languages; Speak English not at all (Estimate)"), or far too brief and cryptic ("B16004_62_EST").

Subsetting, replacement, and data management. As with dataframes and other matrix-like data structures, often a user may be interested in only a subset of an acs object—a few rows or columns, for example. Consistent with the standard R convention, the acs.R package defines methods for subsetting acs objects by row or column using square brackets: if my.data is an acs-class object, then my.data[1:10,6] (for example) would return a new acs object containing data from the first ten rows of the sixth column. Importantly, subsetting returns a *complete* acs object, with estimates and standard errors, as well as all the metadata slots carried over from the original object.

Similarly, data within an acs object may be modified via the standard R syntax for replacement, with a slight twist: since an acs object must contain *both* estimates *and* standard errors, the replacement must provide a *list* of two vectors (or matrices, for more complex replacements). The first (or the one named "estimate") is used to replace the estimate data, and the second (or the one named "error" or "standard.error") is used to replace the standard error data. Thus, my.data[7,2] <- list(estimate=200,error=12) would replace the estimate in row 7, column 2 of my.data with the value of 200 and the corresponding standard error would be changed to 12; my.data[3,6:8] <- list(matrix(c(99,101,76), nrow=1), matrix(c(12,9,44), nrow=1)) would replace the estimates and errors in my.data with these new values, while keeping the other metadata the same.[5]

The package also provides new methods for cbind() and rbind() to deal appropriately with binding together multiple acs objects,[6] checking first to see whether the objects are compatible (same year, same span, same currency units, same column or row headers, etc.), and then creates a new object by pasting them together.

New methods

Once a user has created some acs objects to hold estimate-type data, these objects may be combined, manipulated, and analyzed with a number of new methods provided by the package. In general, methods that combine or modify acs objects—including those for subsetting (but not replacement)—follow R's functional style: rather than actually modifying the object, they return a *new* object which can be saved or used, but the original object remains the same (unless reassigned: e.g., my.data<-my.data + your.data).

[5] One exception: the modified flag for the object will be changed to TRUE to represent that the data has been altered.

[6] Currently only two.

Displaying ACS data. At present, the `acs.R` package provides new `acs` methods to R's generic functions for displaying data. By default, evaluating the name of an `acs` object (which is functionally the same as a call to `show()`) will print some standard header information concerning the year(s) of the survey, followed by a matrix of the estimates with 90% confidence margins. Calls to `summary()` currently provide standard summary data on both estimates and standard errors, although this is clearly less than ideal, as it strips the connection between estimates and errors; we are working with the target audience to determine whether some other summary statistics might be more helpful.

The package also provides a basic `plot()` method for `acs` objects, by default plotting the estimates as points in series by either rows or columns, with confidence bands representing 90% margins of error. If `plot()` has been passed an `acs` object of length one (a single cell), a probability density curve is plotted instead of a confidence interval. The function adds additional options to `plot(...)`, allowing the user to specify alternate confidence intervals, as well as colors, line types, and other graphic elements for these plots.

Math. To simplify many of the procedures described in Section without requiring the user to keep track of estimates and standard errors, the package redefines basic arithmetic operations and other common functions to deal appropriately with `acs` objects. Building on the example described in Section on page 3, if `my.data` represents an `acs` object with tract-level survey counts to be summed in the first three rows, `my.data[1,1] + my.data[2,1] + my.data[3,1]` (or even `sum(my.data[1:3,1])` would provide a new estimate and standard error for their total. Similarly, `my.data[,8]/my.data[,1]` could create a variable to hold a new proportion (or ratio) derived from ACS counts, dividing all estimates in the column 8 (say, the estimated number of senior citizens with limited English ability, by tract) by those in column 1 (say, the estimated total number of people this age living in the tract). All mathematical operators can also combine `acs` object and ordinary numbers, so it is possible to add (or subtract, multiply, divide) an `acs` object and a constant—for example, to add $1,000 to all the estimates in a column or to turn proportions into percentages by multiplying by 100.

The new methods provided for these math functions provide limited error-checking at present (for example, ensuring that all currency has been adjusted for the same year when it is added), and some warnings (for example, noting when the results of a division operation assume that the numerator is a subset of the denominator), but we still need to do more along these lines.

Statistical analysis. With nothing more than basic mathematical operators, users could certainly recreate most standard statistical functions, but one of the goals behind `acs.R` is to simplify much of this routine work. At present, the package provide a new method for the existing R `confint()` function, which accepts an `acs` object as input and returns the upper and lower bounds of a confidence interval of a specified level (90% by default, but specified through the `conf.level` option). The package also provides a new `t.test()` function to compare differences in estimates between locations or across time, which deal correctly with all of the complications of multi-year estimates raised above.

Other bells and whistles

The package currently includes one additional function, `acs.currency.convert()`, to adjust the `currency.year` for a given `acs` object using historical consumer price index data (see Section). In the future we hope to add additional functions, including a richer set of summary and plot options, methods for R's `merge()` and `apply()` functions, and the ability to interact directly and manage queries with the Census "DataFerret" and "FactFinder 2" sites.

Finally, we would like to stress the notion of this effort as a *collaborative* development model; if you have suggestions, concerns, or would just like to test out the code as we develop it, please email `eglenn@mit.edu`.

Acknowledgements

Planners working in the U.S. all owe a tremendous debt of gratitude to our truly excellent Census Bureau, and this seems as good a place as any to recognize this work. In particular, I have benefited from the excellent guidance the Census has issued on the transition to the ACS: the methodology coded into the `acs.R` package draws heavily on these works, especially the *Compass* series cited in the references (U.S. Census Bureau, 2009).

I would also like to thank my colleagues in the Department of Urban Studies and Planning at MIT, including Professor Joe Ferreira who encouraged me to attend the CUPUM conference, and Department Head Amy Glasmeier, who helped get me here. Both of these individuals, as well as Duncan Kincaid in our computing center, have been supportive of my efforts to introduce programming methods in general—and R in particular—into our Master in City Planning program. Finally, I should thank the graduate students in my "Quantitative Reasoning and Statistical Methods" classes over the past three years, who have been willing to experiment with R and have provided excellent feedback on the challenges of working with ACS at the local level.

Traditionally one should also thank one's wife and kids for being supportive, but they are getting a family vacation to Banff out of this, so I think they are all set. (But thanks anyways, of course.)

References

Alexander, C. H. (2001). Still rolling: Leslie Kish's "Rolling Samples" and the American Community Survey. In *Proceedings of Statistics Canada Symposium.*

Alexander, C. H. (2002). *A discussion of the quality of estimates from the American Community Survey for small population groups* (Tech. Rep.). Washington, DC: U.S. Census Bureau.

Almquist, Z. W. (2010). US Census spatial and demographic data in R: The UScensus2000 suite of packages. *Journal of Statistical Software*, *37*(6), 1–31.

Bivand, R. S., Pebesma, E. J., & Gómez-Rubio, V. (2008). *Applied spatial data analysis with R*. New York, NY: Springer-Verlag.

Chambers, J. M. (1998). *Programming with data: A guide to the S language*. New York, NY: Springer-Verlag.

Chambers, J. M. (2006). *How S4 methods work* (Tech. Rep.). R Core Development Team.

Citro, C. F., & Kalton, G. (Eds.). (2007). *Using the American Community Survey: Benefits and challenges*. Washington, DC: National Research Council.

Hankin, R. K. S. (2007). *A step-by-step guide to writing a simple package that uses S4 methods: a "hello world" example* (Tech. Rep.). R Core Development Team.

MacDonald, H. (2006). The American Community Survey: Warmer (more current), but fuzzier (less precise) than the decennial census. *Journal of the American Planning Association*, *72*(4), 491-504.

U.S. Census Bureau. (2009). *A compass for understanding and using American Community Survey data: What state and local governments need to know* (Tech. Rep.). Washington, DC: U.S. Census Bureau.