snp17.*y*	Robust confidence intervals for median (and other percentile) slopes

Author: Roger Newson, Imperial College London, UK. Email: r.newson@imperial.ac.uk Date: 30 May 2012.

Abstract: A program is presented for calculating robust confidence intervals for generalized Theil–Sen median (and other percentile) slopes (and per–unit ratios) of a variable Y with respect to a variable X. The confidence intervals are robust to the possibility that the conditional population distributions of Y, given different values of X, differ in ways other than location, such as having unequal variances. The program uses the program `somersd`, and is part of the `somersd` package.

Keywords: robust, confidence interval, median, percentile, slope, difference, ratio, Kendall's tau, Somers' D, Theil–Sen, Hodges–Lehmann.

1 Syntax

`censlope` *yvarname xvarname* [*weight*] [`if` *exp*] [`in` *range*] [`,` `centile(`*numlist*`)` `eform` `ystargenerate(`*newvarlist*`)` `estaddr` *somersd_options iteration_options*]

where *yvarname* and *xvarname* are variable names, *somersd_options* are any of the options used by `somersd`, and *iteration_options* are any of the options described under **Iteration options**.

`fweights`, `iweights` and `pweights` are allowed; see help for `weight`. They are interpreted as for `somersd`.

`bootstrap`, `by`, `jackknife`, and `statsby` are allowed; see help for `prefix`.

1.1 Description

`censlope` calculates confidence intervals for generalized Theil–Sen median slopes, and other percentile slopes, of a Y–variable specified by *yvarname* with respect to an X–variable specified by *xvarname*. These confidence intervals are robust to the possibility that the population distributions of the Y–variable, conditional on different values of the X–variable, are different in ways other than location. This might happen if, for example, the conditional distributions had different variances. For positive–valued Y–variables, `censlope` can be used to calculate confidence intervals for median per–unit ratios, or other percentile per–unit ratios, associated with a unit increment in the X–variable. if the X–variable is binary with values 0 and 1, then the generalized Theil–Sen percentile slopes are the generalized Hodges–Lehmann percentile differences between the group of observations whose X–value is 1 and the group of observations whose X–value is 0. `censlope` is part of the `somersd` package, and requires the `somersd` program in order to work. It executes the `somersd` command

`somersd` *xvarname yvarname* [*weight*] [`if` *exp*] [`in` *range*] [, *somersd_options*]

and then estimates the percentile slopes. The estimates and confidence limits for the percentile slopes are evaluated using an iterative numerical method, which the user may change from the default, using the *iteration_options*.

1.2 Ordinary options

`centile(`*numlist*`)` specifies a list of percentile slopes to be reported, and defaults to `centile(50)` (median only) if not specified. Specifying `centile(25 50 75)` will produce the 25th, 50th and 75th percentile differences.

`eform` specifies that exponentiated percentile slopes are to be given. This option is used if *yvarname* specifies the log of a positive–valued variable. In this case, confidence intervals are calculated for percentile ratios or per–unit ratios between values of the original positive variable, instead of for percentile differences or per–unit differences.

`ystargenerate(`*newvarlist*`)` specifies a list of variables to be generated, corresponding to the percentile slopes, containing the differences $Y^*(\beta) = Y - X\beta$, where β is the percentile slope. The variable names in the *newvarlist* are matched to the list of percentiles specified by the `centile()` option, sorted in ascending order of percent. If the two lists have different lengths, then `censlope` generates a number *nmin* of new variables equal to the minimum length of the two lists, matching the first *nmin* percentiles with the first *nmin* new variable names. Usually, there is only one percentile slope (the median slope), and one new `ystargenerate()` variable, whose median can be used as the intercept when drawing a straight line through the data points on a scatter plot.

`estaddr` specifies that the results saved in `r()` will also be saved in `e()` (see **Saved results**). This makes it easier to use `censlope` with `parmby`, in order to create an output dataset (or resultsset) with one observation per by–group and data on confidence intervals for Somers' D and median slopes. `parmby` is part of the package `parmest`, downloadable from SSC.

1.3 Iteration options

options	description
fromabs(#)	initial estimate for absolute magnitude of slopes
brackets(#)	maximum number of rows for the bracket matrix
technique(*algorithm_spec*)	iterative numerical solution technique
iterate(#)	perform maximum of # iterations; default is iterate(16000)
tolerance(#)	tolerance for the percentile slopes
log	display an iteration log of the brackets during bracket convergence
nolimits	do not calculate confidence limits

where *algorithm_spec* is

algorithm [# [*algorithm* [#]] ...]

and *algorithm* is { bisect | regula | ridders }

The censlope command calculates estimates and confidence limits for a median or other percentile slope β by solving numerically a scalar equation in β, using an iterative method. The options controlling the exact iterative method will probably not be used very often, because censlope is intended to have sensible defaults. However, users who wish to change the default method may do so, using a set of options similar to the maximization options used by Stata maximum likelihood estimation commands (see [R] **maximize**). These options are as follows:

fromabs(#) specifies an initial estimate of the typical absolute magnitude of a percentile slope. If fromabs() is not specified, then it defaults to the aspect ratio $(ymax - ymin)/(xmax - xmin)$ (where $xmax$ and $xmin$ are the maximum and minimum X–values, and $ymax$ and $ymin$ are the maximum and minimum Y–values) if that ratio is defined and nonzero, and to 1 otherwise. This magnitude is used in the construction of the bracket matrix. Candidate bracket β–values will have values of zero or of $\pm fromabs \times 2^K$, where K is a nonnegative integer. The bracket matrix is a matrix with 2 columns and 3 or more rows, each row containing a candidate β–value in column 1 and the corresponding ζ^*–value in column 2. It is used to find an initial pair of β–values for input into the iterative numerical solution method, which attempts to find a solution in β between the two initial β–values. The bracket matrix is initialized to have β–values $-fromabs$, 0 and $+fromabs$, and ζ^*–values corresponding to these β–values. If a target ζ–value is outside the range of the ζ^*–values of the bracket matrix, then the bracket matrix is extended by adding new rows before the first row by successively doubling the β–value in the first row, or by adding new rows after the last row by successively doubling the β–value in the last row, until there is a ζ^*–value in the second column on either size of the target ζ–value. For an explanation of this terminology, see **Methods and formulas** below.

brackets(#) specifies a maximum number of rows for the bracket matrix. The minimum is brackets(3). The default is brackets(1000).

technique(*algorithm_spec*) specifies an iterative solution method for finding a solution in β to the equation to be solved. The following algorithms are currently implemented in censlope.

technique(bisect) specifies an adapted version of the bisection method for step functions.

technique(regula) specifies an adapted version of the regula falsi (or false position) method for step functions.

technique(ridders) specifies an adapted version of the method of Ridders (1979) for step functions.

The default is technique(ridders 5 bisect *iterate*), where *iterate* is the value of the iterate() option. The bisection method is guaranteed to converge in a number of iterations similar to the binary logarithm of the tolerance() option. The regula falsi and Ridders methods are usually faster if the ζ^*–function is very nearly continuous, but may sometimes be slower if the ζ^*–function is a very discrete step function. All methods are modified versions, for step functions, of the methods of the same names described in Press *et al.* (1992).

You can switch between algorithms by specifying more than one in the technique() option. By default, censlope will use an algorithm for five iterations before switching to the next algorithm. To specify a different number of iterations, include the number after the technique in the option. For example, specifying technique(ridders 10 bisect 1000) requests that censlope perform 10 iterations using the Ridders algorithm, perform 1000 iterations using the bisection algorithm, and then switch back to Ridders for 10 iterations, and so on. The process continues until convergence, or until the maximum number of iterations is reached.

iterate(#) specifies the maximum number of iterations. When the number of iterations equals iterate(), the

iterative solution program stops and records failure to converge. If convergence is declared before this threshold is reached, it will stop when convergence is declared. The default value of `iterate(#)` is the current value of `set maxiter`, which is `iterate(16000)` by default.

`tolerance(#)` specifies the tolerance for the percentile differences. When the relative difference between the current β–brackets is less than or equal to `tolerance()`, the `tolerance()` convergence criterion is satisfied. `tolerance(1e-6)` is the default.

`log` specifies that an iteration log showing the progress of the numerical solution method is to be displayed. Note that, if an iteration log is displayed, then there will be 4 separate iteration sequences per percentile, estimating the left estimate, the right estimate, the lower confidence limit, and the upper confidence limit, respectively. For this reason, the default is not to produce an iteration log. However, if `censlope` is expected to be slow (as in the case of very large datasets), then an iteration log can be specified to reassure the user that progress is being made.

`nolimits` specifies that lower and upper confidence limits will not be calculated. This will save computational time if the user plans to calculate confidence limits using a resampling prefix command, such as `bootstrap` or `jackknife`.

1.4 Saved results

`censlope` saves the following results in `r()`:

Scalars

`r(level)`	confidence level
`r(fromabs)`	value of the `fromabs()` option
`r(tolerance)`	value of the `tolerance()` option

Macros

`r(yvar)`	name of the *Y*-variable
`r(xvar)`	name of the *X*-variable
`r(eform)`	`eform` if specified
`r(centiles)`	list of percents for the percentiles
`r(technique)`	list of techniques from the `technique()` option
`r(tech_steps)`	list of step numbers for the techniques

Matrices

`r(cimat)`	confidence intervals for percentile differences or ratios
`r(rcmat)`	return codes for entries of `r(cimat)`
`r(bracketmat)`	bracket matrix
`r(techstepmat)`	column vector of step numbers for the techniques

The matrix `r(cimat)` has one row per percentile, and columns containing the percents, percentile estimates, lower confidence limits and upper confidence limits, labelled `Percent`, `Pctl_Slope`, `Minimum` and `Maximum` if `eform` is not specified, or `Percent`, `Pctl_Ratio`, `Minimum` and `Maximum` if `eform` is specified. The matrix `r(rcmat)` has the same numbers of rows and columns as `r(cimat)`, with the same labels, and the first column contains the percents, but the other entries contain return codes for the estimation of the corresponding entries of `r(cimat)`. These return codes are equal to 0 if the β–value was estimated successfully (or not requested by the user), 1 if the corresponding ζ^*–value could not be calculated, 2 if the corresponding ζ^*–value could not be bracketed, 3 if the β–brackets failed to converge, and 4 if the β–value could not be calculated from the converged β–brackets. The matrix `r(bracketmat)` is the final version of the bracket matrix described in the help for the `fromabs()` and `brackets()` options of `censlope`, and has one row per β–bracket, and two columns, labelled `Beta` and `Zetastar`, containing the β–brackets and the corresponding ζ^*–values. The matrix `r(techstepmat)` is a column vector, with one row for each of the techniques listed in the `technique()` option, with a row label equal to the name of the technique and a value equal to the number of steps for that technique. The `fromabs()`, `brackets()`, `tolerance()` and `technique()` options are described under **Iteration options** above.

`censlope` also saves in `e()` a full set of estimation results for the `somersd` command

`somersd` *xvarname yvarname* [*weight*] [`if` *exp*] [`in` *range*] [, *somersd_options*]

as described in **Description** above. If `estaddr` is specified, then this set of estimation results is expanded by adding a set of `e()` results with the same names and contents as the `r()` results. This allows the user to pass a `censlope` command to `parmby`, producing an output dataset (or resultsset) with one observation per by–group and data on confidence intervals for Somers' D and for the median slope.

2 Methods and formulas

The Theil–Sen median slope was introduced by Theil (1950) and developed further by Sen (1968). If the X–variable is binary, then the Theil–Sen slope is the Hodges–Lehmann median difference (Hodges and Lehmann, 1963). The methods used by `censlope` are a generalization of the methods of Theil and Sen. They include, as a special case, the methods used by `cendif` (Newson, 2000b), which calculates confidence intervals for generalized Hodges–Lehmann median differences, and is also part of the `somersd` package.

Percentile slopes are defined in terms of the parameters Somers' D (Somers, 1962) and Kendall's τ_a (Kendall and Gibbons, 1990). A discussion of the connections between these parameters appears in Newson (2002). For the purposes of `censlope`, we will define Somers D and Kendall's τ_a in the very general sense used in the manual `somersd.pdf`, distributed with the `somersd` package. Given two random variables U and V, we denote by $\tau(U,V)$ the Kendall's τ_a of U and V, and denote by $D(U|V)$ the Somers' D of U with respect to V. Briefly, if two (U,V)–pairs (U_i,V_i) and (U_j,V_j) are sampled from some population of such pairs using some sampling scheme, then $\tau(U,V)$ is the difference between the probability that the two (U,V)–pairs are concordant (meaning that the higher of the two U–values is paired with the higher of the two V–values) and the probability that the two (U,V)–pairs are discordant (meaning that the higher of the two U–values is paired with the *lower* of the two V–values). We define $D(U|V)$ as the difference between the corresponding *conditional* probabilities, given that the two V–values are strictly ordered (meaning that one V–value is known to be higher than the other V–value). Note that both $\tau(U,V)$ and $D(U|V)$ are differences between probabilities, and therefore both may have values ranging from -1 (for a "perfect negative association") to +1 (for a "perfect positive association"), but $\tau(U,V)$ is always symmetric in U and V, whereas $D(U|V)$ is not. We will use the notation $\theta(U,V)$ to stand for the value of either $\tau(U,V)$ or $D(U|V)$ in the population, and denote the corresponding sample value as $\hat{\theta}(U,V)$. The `somersd` package allows us to choose between Somers D and Kendall's τ_a using the `taua` option, and also provides other options, to specify a version of either parameter corresponding to a specific sampling scheme.

For an outcome variable Y, a predictor variable X and a proportion q such that $0 \leq q \leq 1$, a $100q$th percentile slope of Y with respect to X is defined as a value β satisfying

$$\theta(Y-\beta X, X) \quad = \quad 1-2q. \tag{1}$$

If $q = 0.5$, then $1-2q = 0$, and a solution in β to (1) is known as a Theil–Sen median slope (Theil, 1950; Sen, 1968). Note that there is not always a unique solution to (1) in β. If the joint population distribution of Y and X is discrete (as are all population distributions sampled by applied statisticians in the real world), then $\theta(Y-\beta X, X)$ will be a monotonically non–increasing step function of β, and there may be no exact solution, or an interval of exact solutions. However, the confidence intervals derived here will contain any solution with the specified confidence level, *if* a solution exists.

If $\theta(\,\cdot\,,\,\cdot\,)$ stands for Somers' D rather than Kendall's τ_a, then the value of $\theta(Y-\beta X, X)$ depends only on the conditional distribution of pairs of bivariate observations (X_1,Y_1) and (X_2,Y_2) satisfying $X_1 < X_2$. For such pairs of observations, the pairwise slope $(Y_2-Y_1)/(X_2-X_1)$ is always defined. If neither X nor Y is subject to left– or right–censorship, then the equality (1) becomes

$$\begin{aligned} 1-2q &= D(\,Y-\beta X\,|\,X\,) \\ &= \Pr(\,Y_1-\beta X_1 < Y_2-\beta X_2\,) \quad - \quad \Pr(\,Y_1-\beta X_1 > Y_2-\beta X_2\,) \\ &= \Pr[\,(Y_2-Y_1)/(X_2-X_1) > \beta\,] \quad - \quad \Pr[\,(Y_2-Y_1)/(X_2-X_1) < \beta\,]. \end{aligned} \tag{2}$$

Therefore, a 0.5th percentile (or median) slope has the expected property that a pairwise slope is equally likely to be above or below it. If in addition the distributions of X and Y are limited to finite sets of discrete values, then the distribution of pairwise slopes will be bounded, and a 0th percentile slope will be any number below all possible pairwise slopes, and a 100th percentile slope will be any number above all possible pairwise slopes.

We aim to include a value β in a confidence interval for a $100q$th percentile slope if and only if the sample $\hat{\theta}(Y-\beta X, X)$ is compatible with a *population* $\theta(Y-\beta X, X)$ equal to $1-2q$. The methods of Newson (2000a) and Newson (2006b), used by the program `somersd` and updated in the manual `somersd.pdf`, typically use a monotonically–increasing transformation $\zeta(\,\cdot\,)$, which *may* be Normalizing and/or variance–stabilizing when applied to $\hat{\theta}(Y-\beta X, X)$. We define

$$\zeta^*(\beta) \quad = \quad \zeta[\,\hat{\theta}(Y-\beta X, X)\,]. \tag{3}$$

Note that $\zeta^*(\beta)$ is a randomly variable function of β, with a population standard error $\mathrm{SE}[\,\zeta^*(\beta)\,]$, estimated consistently by a corresponding *sample* standard error $\widehat{\mathrm{SE}}[\,\zeta^*(\beta)\,]$, whose formula is one of those described in

somersd.pdf. We will assume that, if $\theta(Y - X\beta, X) = 1 - 2q$, then the pivotal quantity

$$[\zeta^*(\beta) - \zeta(1-2q)] / \mathrm{SE}[\zeta^*(\beta)] \tag{4}$$

has a standard Normal distribution. In general, the sample $\zeta^*(\beta)$ is a monotonically non–increasing step function of β, bounded above by $\zeta(-1)$ and below by $\zeta(1)$, either of which bounds may be infinite, depending on the choice of transformation $\zeta(\cdot)$.

Figure 1 illustrates an example of a function $\zeta^*(\beta)$ from the auto data. Here, the observations are car models, the Y–variable is trunk (trunk space in cubic feet), the X–variable is foreign (a binary variable indicating non–US origin), the transformation is the hyperbolic arctangent or Fisher's z (as recoomended by Edwardes (1995)), and a slope β is a difference (expressed in cubic feet) between cars made by non–US and US companies. The function $\zeta^*(\beta)$ is plotted against the differences β over the range of differences for which the absolute value of $\zeta^*(\beta)$ is finite. (As there are no differences between non–US and US cars above 9 cubic feet or below -18 cubic feet, the value of $\zeta^*(\beta)$ is $-\infty$ for $\beta > 9$ and $+\infty$ for $\beta < -18$.) This plot was made using the program cendif, which is restricted to binary X–variables, and calculates the full set of differences in the Y–variable between observations in the two groups. The square data points give values of $\zeta^*(\beta)$ for differences β actually observed in the auto data, and the line gives values of $\zeta^*(\beta)$ for values of β between these observed values. Note that the sample $\zeta^*(\beta)$ is a monotonically non–increasing step function of β, which is discontinuous at the observed differences and constant within the open intervals between consecutive observed differences. This implies that a unique exact solution for (1) does not usually exist, as there is usually either no exact solution or an interval of exact solutions between two consecutive observed differences. In a finite sample, this will be true for observed slopes in general, whether or not the X–variable is binary.

If we knew the value of $\mathrm{SE}[\hat{\zeta}^*(\beta)]$, then a $100(1-\alpha)\%$ confidence interval for a $100q$th percentile difference might be the interval of values β for which

$$\zeta(1-2q) - z_\alpha \,\mathrm{SE}[\hat{\zeta}^*(\beta)] \quad \leq \quad \zeta^*(\beta) \quad \leq \quad \zeta(1-2q) + z_\alpha \,\mathrm{SE}[\hat{\zeta}^*(\beta)], \tag{5}$$

where z_α is the $100(1-\frac{1}{2}\alpha)$th percentile of the standard Normal distribution. To construct such a confidence interval, we proceed as follows. Given a value ζ in the range of $\zeta(\cdot)$, we define

$$B_\mathrm{L}(\zeta) = \sup\{\beta : \zeta^*(\beta) > \zeta\}, \quad B_\mathrm{R}(\zeta) = \inf\{\beta : \zeta^*(\beta) < \zeta\},$$

$$B_\mathrm{C}(\zeta) = \begin{cases} \text{Undefined}, & \text{if } B_\mathrm{L}(\zeta) = -\infty \text{ and } B_\mathrm{R}(\zeta) = \infty, \\ B_\mathrm{L}(\zeta), & \text{if } B_\mathrm{L}(\zeta) > -\infty \text{ and } B_\mathrm{R}(\zeta) = \infty, \\ B_\mathrm{R}(\zeta), & \text{if } B_\mathrm{R}(\zeta) < +\infty \text{ and } B_\mathrm{L}(\zeta) = -\infty, \\ [B_\mathrm{L}(\zeta) + B_\mathrm{R}(\zeta)]/2, & \text{otherwise.} \end{cases} \tag{6}$$

(By convention, the supremum (or infimum) of a set unbounded to the right (or left) are defined as $+\infty$ (or $-\infty$), respectively, and the supremum and infimum for an empty set are $-\infty$ and $+\infty$, respectively.) Clearly, $B_\mathrm{L}(\zeta) \leq B_\mathrm{C}(\zeta) \leq B_\mathrm{R}(\zeta)$, and the values of $B_\mathrm{L}(\zeta)$ and $B_\mathrm{R}(\zeta)$ (if finite) can be either the same observed slope, or two successive observed slopes. The confidence interval for a $100q$th percentile slope is centered on the sample $100q$th percentile slope, defined as

$$\hat{\xi}_q = B_\mathrm{C}[\zeta(1-2q)]. \tag{7}$$

The lower and upper confidence limits for a qth percentile slope are, respectively,

$$\hat{\xi}_q^{(\min)} = B_\mathrm{L}\{\zeta(1-2q) - z_\alpha \,\widehat{\mathrm{SE}}[\zeta^*(\hat{\xi}_q)]\}, \quad \hat{\xi}_q^{(\max)} = B_\mathrm{R}\{\zeta(1-2q) + z_\alpha \,\widehat{\mathrm{SE}}[\zeta^*(\hat{\xi}_q)]\}. \tag{8}$$

If tdist is specified, then censlope uses the t-distribution with $\nu = N - 1$ degrees of freedom if there are N unclustered observations, or $\nu = N_\mathrm{clust} - 1$ degrees of freedom if there are N_clust clusters, instead of the normal distribution, and therefore $t_{\nu,\alpha}$ replaces z_α in (8). Note that the upper and lower confidence limits may occasionally be infinite, in the case of extreme percentiles and/or very small sample numbers. censlope codes these infinite limits as plus or minus the Stata creturn value c(maxdouble), which is the system maximim double precision value (see on–line help for creturn).

Figure 1 illustrates these formulas in the case of the Y–variable trunk and the X–variable foreign in the auto data. The median difference in trunk capacity $\hat{\xi}_{0.5}$, and its lower and upper 95% confidence limits, are shown as reference lines on the horizontal axis. The estimated median difference in trunk space between non–US and US cars is -3 cubic feet, with 95% confidence limits from -5 to -1 cubic feet. The reference lines on the vertical axis are the optimum, minimum and maximum values of $\zeta^*(\beta)$ required for β to be in the confidence interval.

Note that `censlope` inherits *all* the options of `somersd`, so $\theta(X, Y - \beta X)$ in (1) can stand for any of the generalized versions of Somers' D and Kendall's τ_a described in `somersd.pdf`. We can therefore estimate generalized percentile slopes or differences, defined in terms of generalized Somers' D or Kendall's τ_a parameters. For instance, we can use the `wstrata()` option to estimate median slopes and differences restricted to comparisons within strata defined by a confounding variable, or we might use the option `funtype(wcluster)` to estimate within–cluster median differences and slopes.

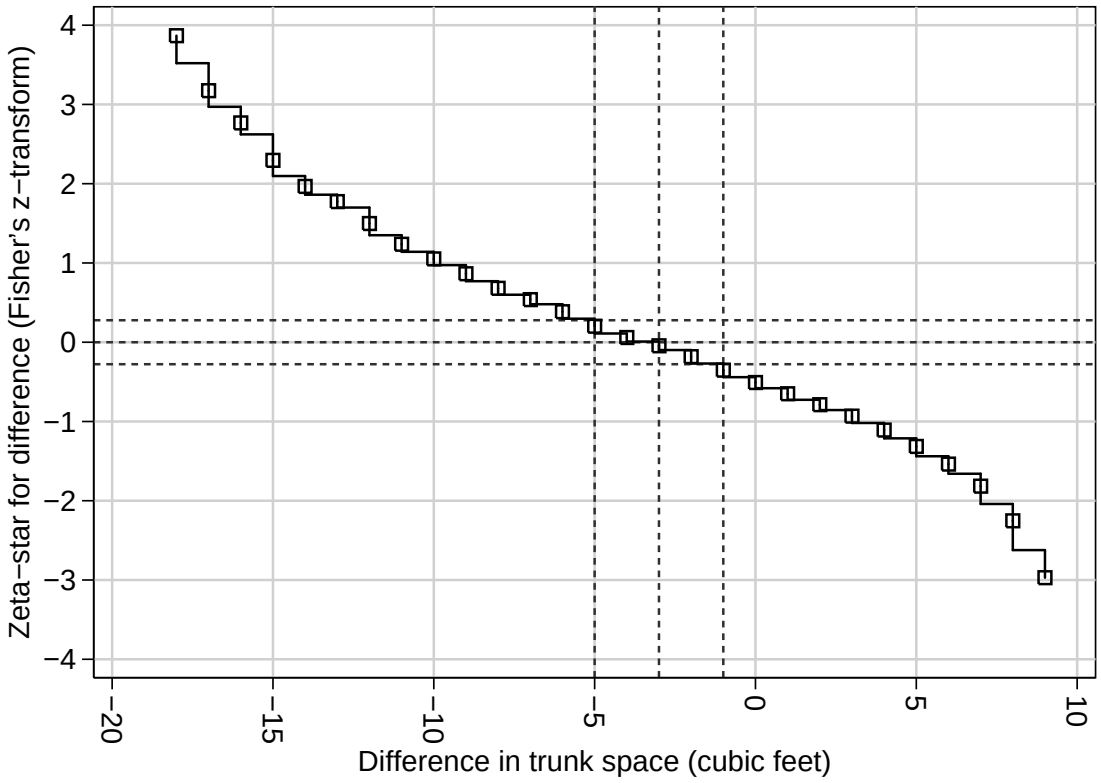

Figure 1. $\zeta^*(\beta)$ plotted against the difference β in trunk space between non–US and US cars

2.1 Numerical evaluation of $B_{\mathrm{L}}(\zeta)$ and $B_{\mathrm{R}}(\zeta)$

We can see, by (6), (7) and (8), that the key to calculating confidence intervals for percentile slopes is calculating $B_{\mathrm{L}}(\zeta)$ and $B_{\mathrm{R}}(\zeta)$ for a given ζ. Traditionally, this has been done by calculating every possible pairwise slope $(Y_i - Y_j)/(X_i - X_j)$ for each pair of observations in the sample to make a dataset of all pairwise slopes, and by using this dataset to find the median (and other percentile) slopes. This requires an amount of computational time, and data storage space, proportional to N^2, where N is the number of observations. For this reason, confidence intervals for median slopes have traditionally only been calculated for small samples. as have confidence intervals for other rank statistics, such as Somers' D and Kendall's τ_a, which are also commonly calculated by comparing every pair of (X, Y)–pairs. See Sprent and Smeeton (2007) for some worked examples using traditional methods.

It is not necessary to compare each pair of (X, Y)–pairs. `somersd` uses the algorithm of Newson (2006a), which calculates Somers' D, Kendall's τ_a and their jackknife variances in a time asymptotically proportional to $N \log N$, using a search tree to avoid having to compare every pair of (X, Y)–pairs. We can therefore use `somersd` to calculate $\zeta^*(\beta)$ for any β in a time proportional to $N \log N$. `censlope` uses versions of some of the iterative numerical methods of Chapter 9 of Press *et al.* (1992), modified for step functions, to evaluate $B_{\mathrm{L}}(\zeta)$ and $B_{\mathrm{R}}(\zeta)$, for a given ζ. This is done by defining the object function $\omega(\beta) = \zeta^*(\beta) - \zeta$ and attempting to find a solution in β to the equation

$$0 \quad = \quad \omega(\beta) \quad = \quad \zeta^*(\beta) - \zeta, \tag{9}$$

using `somersd` to calculate $\omega(\beta)$. This requires a computational time of order $N_{\mathrm{eval}} N \log N$, where N_{eval} is the number of evaluations of the object function in the iteration sequence. For very large datasets ($N > 1000$), this will typically take less time than a quadratic algorithm that compares all pairs of (X, Y)–pairs. However, in small datasets, such

as the `auto` data, `cendif` typically takes *much* less time to calculate a Hodges–Lehmann median difference, using its quadratic algorithm, than `censlope` takes using one of its iterative algorithms to do the same. This is not surprising. The performance study of Newson (2006a) seems to imply that, if there are less than 100 observations, then the execution time of `somersd` is dominated by "constant" terms not dependent on sample size, whether `somersd` is using a quadratic algorithm or a search tree algorithm. Therefore, we would expect the computational time for an iteration sequence, involving N_{eval} calls to `somersd`, to have a component proportional to N_{eval}, which will dominate execution time if the sample size is small and the number of iterations is large.

The algorithms used by `censlope` use versions of standard bracket convergence methods for finding roots, modified for step functions. To solve an equation of the form (9), we would normally start with two β–values β_0 and β_1, whose corresponding respective ω–values ω_0 and ω_1 bracket zero, meaning that $\omega_0\,\omega_1 < 0$ (because the two ω–values have opposite signs). If $\omega(\,\cdot\,)$ is continuous, then, by the imtermediate–value theorem, there will be a solution to (9) between β_0 and β_1, and this solution will be unique if $\omega(\,\cdot\,)$ is strictly monotonic. However, in this case, $\omega(\,\cdot\,)$ is not continuous, but a nonincreasing step function similar to Figure 1. Therefore, instead of expecting to find a unique solution to (9), we try to find a supremum (or infimum) of the set of β–values with positive (or negative) values of the object function. In this case, the two ω–values are said to bracket zero if and only if

$$\text{sign}(\omega_1) \neq 0 \quad \text{and} \quad \text{sign}(\omega_1) \neq \text{sign}(\omega_0). \tag{10}$$

In other words, ω_1 is a strict bracket, which must not be zero, whereas ω_0 is a partial bracket, which may either be zero or have the opposite sign to ω_1. During each iteration, we compute a new β–value β_{new}, between β_0 and β_1, with a corresponding ω–value $\omega_{\text{new}} = \omega(\beta_{\text{new}})$. In the next iteration, the pair $(\beta_{\text{new}}, \omega_{\text{new}})$ will replace (β_1, ω_1) if $\text{sign}(\omega_{\text{new}}) = \text{sign}(\omega_1)$, and will replace (β_0, ω_0) otherwise. Iterations proceed until β_0 and β_1 have a relative difference no more than the value of the `tolerance()` option. When this has happened, we can use either of the β–values to estimate $B_{\text{L}}(\zeta)$ or $B_{\text{R}}(\zeta)$ (depending on whether we initialized $\beta_1 < \beta_0$ or $\beta_0 < \beta_1$).

The numerical methods specified by the `technique()` option differ in the method used to calculate β_{new}. The technique `bisect` does this using the simple bisection formula $\beta_{\text{new}} = (\beta_0 + \beta_1)/2$. The technique `regula` uses simple bisection if $\omega_0 = 0$, and uses the regula falsi (or false position) method otherwise. The technique `ridders` uses simple bisection if $\omega_0 = 0$, and uses the method of Ridders (1979) otherwise. The simple bisection method is guaranteed to converge slowly, whereas the modified regula falsi and Ridders methods will be faster if the object function $\omega(\,\cdot\,)$ is nearly continuous, but may be a lot slower if $\omega(\,\cdot\,)$ is very discrete. The user may specify a combination of methods, such as starting with the regula falsi or Ridders method for earlier iterations (when the object function is nearly continuous over a long interval), and moving to the bisection method later (when the object function is highly discrete over a short interval).

For each percentage $100q$, `censlope` attempts to evaluate $B_{\text{L}}[\zeta(1-2q)]$ and $B_{\text{R}}[\zeta(1-2q)]$ in order to evaluate the percentile estimate $\hat{\xi}_q$, and then (if this evaluation is successful) evaluates the two confidence limits. This implies 4 sequences of iterations, to evaluate the "left estimate", the "right estimate" and the two confidence limits, respectively. Typically, using the default tolerance of `1e-6`, and the "slow but sure" bisection method, this implies 4 sets of around 20 iterations. Together with the initialization of the brackets, this implies a large number (80–100) of calls to `somersd`. However, that number is usually fewer than 100 evaluations per percentile, implying less work than (say) bootstrapping Somers' D, which would typically involve at least 1000 evaluations. On the other hand, if the sample size is large, then this method would probably be unthinkable for practical statisticians without the algorithm of Newson (2006a).

2.2 Comparisons with existing methods

Sen (1968) developed a confidence interval formula for $\hat{\xi}_q$ in the special case where $q = 0.5$, $\theta(Y, X) = \tau(Y, X)$ and $\zeta(\theta) = \theta$, using methods similar to the present ones. In this special case, (1) becomes simply $\tau(Y - \beta X, X) = 0$. The main difference from the present method was in the method used for calculating the distribution of $\zeta^*(\beta)$. Sen assumed that the variables X and $Y - \beta X$ were not only "Kendall-uncorrelated", but also statistically independent. For small sample sizes ($N \leq 10$), the confidence interval was calculated using tables of the exact distribution of the sample Kendall's τ_a, based on that assumption. For larger sample sizes, the population standard error $\text{SE}[\,\zeta^*(\beta)\,]$ was calculated from the marginal sample distribution of X, using the same assumption. (See Kendall and Gibbons (1990) for tables of the exact distribution for small sample sizes, and also for a demonstration that the Central Limit Theorem works *very* well at sample sizes as small as 8 for the sample Kendall's τ_a under the null hypothesis of independence.) The assumption of independence between the predictor variable X and the "residuals" $Y - \beta X$ implies that the conditional population distributions of Y, given each value of X, are different only in location, and may not differ in the conditional variance, or indeed in any other conditional moment about the mean. The original

Sen method therefore does not use the assumption of Normality, but does use the assumption of homoskedasticity, which typically causes more problems when it is wrong.

Lehmann (1963) derived a confidence interval for the Hodges–Lehmann median difference, which is the Theil–Sen slope for binary X–variables, based on the same assumption of independence. This method was popularized by Conover (1999), Campbell and Gardner (1988) and Altman *et al.* (2000), and is available in unofficial Stata, using Duolao Wang's `npshift` routine (Wang, 1999) or Patrick Royston's `cid` routine, downloadable from SSC (Royston, 1998). The method is essentially a special case of the Sen (1968) method, and is presumably subject to the same cautions.

The method used by `censlope` and `cendif`, by contrast, can estimate percentile differences other than the median difference. Even in the case of a median difference, the predictor variable X and the "residuals" $Y - \beta X$ are only assumed to be "Kendall-uncorrelated", and not necessarily independent. The population standard error $\mathrm{SE}[\,\zeta^*(\beta)\,]$ is estimated using the sample standard error $\widehat{\mathrm{SE}}[\,\zeta^*(\beta)\,]$, which is calculated using an infinitesimal jackknife method described in `somersd.pdf`. This method is robust to heteroskedasticity, *possibly* at the price of being less robust to extremely small sample sizes than the traditional methods. Therefore, the method of `censlope` can be compared to the original Sen method as Huber confidence intervals can be compared to maximum likelihood or quasi–likelihood confidence intervals, and the method of `cendif` can be compared to the Lehmann method as the unequal–variance t–test can be compared to the equal–variance t–test. Lehmann's method, like the equal-variance t-test, assumes that you can use data from the larger of two samples to estimate the population variability of the smaller sample.

The issue of heteroskedasticity, as it affects the t–test, is discussed in Moser, Stevens and Watts (1989) and in Moser and Stevens (1992), who explored the issue, using exact analytical formulas to compare the equal–variance t–test with the Satterthwaite unequal–variance t–test. Their conclusion (as I understand it) appears to be that we should view the equal–variance t–test as a special method for use only when we "know" that the sub–population variances are equal, rather than to follow the more "traditional" practice of viewing the unequal–variance t–test as a special method for use only when we "know" that the sub–population variances are unequal. I have carried out some unpublished simulations, comparing `cendif` to the Lehmann method, and to the two t-tests. These simulations, some of which are briefly described in `cendif.pdf` and in Newson (2002), seem to point to a similar recommendation regarding the two types of rank–based methods for median differences. However, more work is probably required on this issue.

A possible alternative confidence interval method for median slopes, also robust to unequal variability, is the bootstrap, recommended by Wilcox (1998). It is possible to use `censlope` together with the Stata `bootstrap` command prefix, and an example of this use appears in the online help for `censlope`. Note that, when `censlope` is used with `bootstrap`, `jackknife` or any other resampling prefix, the user should use the iteration option `nolimits`, which approximately halves the required computation time by not calculating confidence limits iteratively for the individual bootstrap subsamples.

3 Examples

3.1 Example 1. Weight per inch in the `auto` data

In the `auto` data, we can use `censlope` to estimate the median slope of `weight` (in US pounds) with respect to `length` (in US inches) as follows:

```
. censlope weight length
Outcome variable: weight
Somers' D with variable: length
Transformation: Untransformed
Valid observations: 74
Symmetric 95% CI
------------------------------------------------------------------------------
             |             Jackknife
      length |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |   .8286359   .0275321    30.10   0.000     .7746739    .8825978
------------------------------------------------------------------------------
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50   32.745114   30.588225   35.106387
```

The untransformed Somers' D of weight with respect to length is 0.83, with a confidence interval from 0.77 to 0.88, indicating that, in the population from which these cars were sampled, a longer car is 77% to 88% more likely

to be heavier than a shorter car than to be lighter than a shorter car. Each additional inch of length typically adds 30.59 to 35.11 pounds of weight.

If we use the z–transform for Somers' D, then the results are as follows:

```
. censlope weight length, transf(z)
Outcome variable: weight
Somers' D with variable: length
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
      length |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |   1.183767   .0878602    13.47   0.000     1.011564     1.35597
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
                 Somers_D     Minimum     Maximum
      weight   .82863585   .76640832   .87545512
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50   32.745093   30.588221   35.081996
```

This time, Somers' D is 0.77 to 0.88, implying (again) that longer cars are 77% to 88% more likely to be heavier than shorter cars than to be lighter than shorter cars. The typical increase in weight per additional inch of length is 30.59 to 35.08 pounds per inch, which is very similar to the previous confidence interval.

Transformations such as Fisher's z are more likely to be important in estimating percentile slopes other than the median. We can ask for the 25th and 75th percentiles as well, using the `centile()` option:

```
. censlope weight length, transf(z) centile(25(25)75)
Outcome variable: weight
Somers' D with variable: length
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
      length |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |   1.183767   .0878602    13.47   0.000     1.011564     1.35597
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
                 Somers_D     Minimum     Maximum
      weight   .82863585   .76640832   .87545512
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        25   24.102562   19.999994   27.058827
        50   32.745093   30.588221   35.081996
        75   41.818174    38.63634   46.136372
```

We see that the 25th percentile slope is 20.00 to 27.06 pounds per inch, and that the 75th percentile slope is 38.64 to 46.14 pounds per inch.

3.2 Example 2. Weights of non–US and US cars

If we compare the weights of non–US cars with the weights of US cars using `censlope` to calculate a Hodges–Lehmann median difference, then the results are as follows:

```
. censlope weight foreign, transf(z)
Outcome variable: weight
Somers' D with variable: foreign
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
     foreign |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |  -.9749561   .1908547    -5.11   0.000    -1.349024   -.6008878
```

```
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
               Somers_D     Minimum     Maximum
      weight  -.75087413  -.87382282  -.53768098
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50  -1095.0002  -1330.0003  -749.99945
```

We see (by the confidence interval for Somers' D) that non–US cars are 54% to 87% less likely to be heavier than US cars than to be lighter than US cars. The sample median difference is -1095 pounds, and we can be 95% confident that the *population* median difference is between -1330 and -750 pounds. Again, we can estimate other percentile differences than the median:

```
. censlope weight foreign, transf(z) centile(0(25)100)
Outcome variable: weight
Somers' D with variable: foreign
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
     foreign |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |  -.9749561   .1908547    -5.11   0.000    -1.349024   -.6008878
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
               Somers_D     Minimum     Maximum
      weight  -.75087413  -.87382282  -.53768098
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
         0       -3080  -8.99e+307       -3080
        25  -1555.0001  -1790.0001  -1319.9997
        50  -1095.0002  -1330.0003  -749.99945
        75  -485.00001  -810.00056  -99.999931
       100   1619.9993   1619.9999   8.99e+307
```

The 25th percentile difference is between -1790 pounds and -1320 pounds. The 75th percentile difference is -810 to -100 pounds, showing that the weights of US and non–US cars do not overlap a great deal. The 0th percentile difference of -3080 pounds is the most negative difference (between the lightest non–US car and the heaviest US car), and the 100th percentile difference of 1620 pounds is the most positive difference (between the heaviest non–US car and the lightest US car). The lower confidence limit for the 0th percentile difference is `c(mindouble)`, denoting "minus infinity", whereas the higher confidence limit for the 100th percentile is `c(maxdouble)`, denoting "plus infinity" (see help for `creturn`). In fact, the 95% confidence limits for the 0th and 100th percentiles are conservative, as we can be 100% confident that the *population* minimum difference is no higher than the corresponding *sample* minimum difference, and we can be 100% confident that the *population* maximum difference is no lower than the corresponding *sample* maximum difference.

Alternatively, we can estimate Hodges–Lehmann percentile ratios instead of Hodges–Lehmann percentile differences, by logging the weights and using the `eform` option:

```
. gene logweight=log(weight)
. censlope logweight foreign, transf(z) centile(0(25)100) eform
Outcome variable: logweight
Somers' D with variable: foreign
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
     foreign |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
   logweight |  -.9749561   .1908547    -5.11   0.000    -1.349024   -.6008878
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
               Somers_D     Minimum     Maximum
   logweight  -.75087413  -.87382282  -.53768098
95% CI(s) for percentile ratio(s)
   Percent  Pctl_Ratio     Minimum     Maximum
         0   .36363652           0   .36363652
```

```
  25   .57309349   .53268748   .62190848
  50   .67538394   .61424299   .76325174
  75    .8378454   .73890344   .96698107
 100   1.8999989   1.8999984   8.99e+307
```

We see that a randomly–chosen non–US car typically weighs 61% to 76% as much as a randomly–chosen US car, and the 75th percentile ratio of 74% to 97% indicates that there is little overlap between the two groups of cars. The confidence interval for Somers' D shows (again) that a randomly–chosen non–US car is 87% to 54% more likely to be lighter than a randomly–chosen US car than to be heavier than a randomly–chosen US car.

3.3 Example 3. Unadjusted and weight–adjusted fuel efficiencies the `auto` data

`censlope` does not only estimate unadjusted effects. It can also estimate confounder–adjusted effects, which we might not expect to be able to do using rank methods.

In the `auto` data, we might compare fuel efficiencies (in miles per gallon) between non–US and US cars. We can make this comparison either crudely or stratifying by quintile of weight. We use `xtile` to define the quintiles of weight, and `censlope` to measure median differences in fuel efficiency, as follows:

```
. xtile weightgp=weight, nquantiles(5)
. tab weightgp foreign
           5 |
 quantiles |       Car type
 of weight |  Domestic    Foreign |     Total
-----------+----------------------+----------
         1 |         4         11 |        15
         2 |         8          7 |        15
         3 |        12          3 |        15
         4 |        14          1 |        15
         5 |        14          0 |        14
-----------+----------------------+----------
     Total |        52         22 |        74
. censlope mpg foreign, transf(z)
Outcome variable: mpg
Somers' D with variable: foreign
Transformation: Fisher's z
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
     foreign |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         mpg |   .4937249   .1708551     2.89   0.004     .1588551    .8285947
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
                Somers_D     Minimum     Maximum
         mpg   .45716783   .15753219   .67972072
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50    4.999998   1.9999991   7.0000031
. censlope mpg foreign, transf(z) wstrata(weightgp)
Outcome variable: mpg
Somers' D with variable: foreign
Transformation: Fisher's z
Within strata defined by: weightgp
Valid observations: 74
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |               Jackknife
     foreign |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         mpg |  -.3768859   .2170165    -1.74   0.082    -.8022305    .0484587
------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
                Somers_D     Minimum     Maximum
         mpg        -.36  -.66528187   .04842076
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50  -2.0000003  -4.0000008   7.654e-07
```

We divide the cars into weight quintiles using `xtile` to create a new variable `weightgp` denoting weight quintile,

and note that all quintiles except the highest contain both US and non–US cars. When we estimate the unadjusted median difference in fuel efficiency between non–US and US cars, we find that, if we choose a car of each type at random, then the non–US car typically travels 5 more miles per gallon than the US car, with confidence limits from 2 more miles per gallon to 7 more miles per gallon. *However*, if we use the option `wstrata(weightgp)` to restrict the analysis to comparisons between non–US and US cars in the same weight quintile, then we find that non–US cars typically travel 0 to 4 *fewer* miles per gallon than US cars in the same weight quintile.

In this analysis, there is only one confounding variable, namely `weight`. More often, in the real world, there are multiple confounders. However, we can use the method of propensity scores to "collapse" multiple confounders to a single confounder, known as the propensity score, and then use this propensity score to create propensity groups for use in a stratified analysis. (See Rosenbaum and Rubin (1983) and Imai and van Dyk (2004) for more about propensity scores.)

3.3 Example 4. Flow rates in a mountain stream over time

The following example is from Chapter 11 of Sprent and Smeeton (2007). The data points are 7 measurements of rate of flow (expressed in cubic metres per second) in a mountain stream, made at various times (expressed in hours from the start of the thaw). The data are plotted in Figure 2.

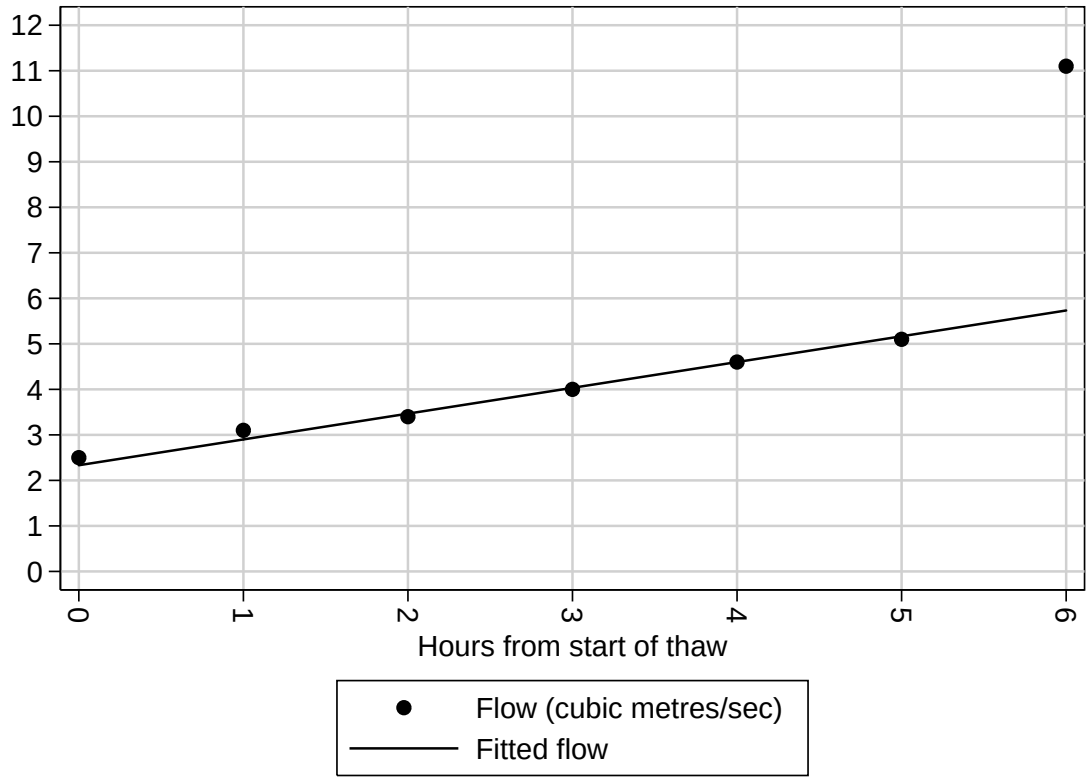

Figure 2. Flow rates of a mountain stream over time (from Sprent and Smeeton, 2007)

We can use `censlope` to calculate a confidence interval for the Theil–Sen median slope of flow with respect to time, expressed in cubic metres per second per hour:

```
. censlope flow hours, tdist transf(z) ystar(resid)
Outcome variable: flow
Somers' D with variable: hours
Transformation: Fisher's z
Valid observations: 7
Degrees of freedom: 6
Symmetric 95% CI for transformed Somers' D
------------------------------------------------------------------------------
             |             Jackknife
       hours |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
```

```
        flow |   17.61636          .        .       .            .           .
-------------------------------------------------------------------------------
Asymmetric 95% CI for untransformed Somers' D
                  Somers_D     Minimum     Maximum
        flow             1           1           1
95% CI(s) for percentile slope(s)
   Percent  Pctl_Slope     Minimum     Maximum
        50   .56666653   .44999988   2.3666683
```

We note that the Somers' D of flow rate with respect to time is 1 (with a zero standard error), indicating that later measurements always recorded higher flow rates than earlier measurements. The sample Theil–Sen median difference was 0.567 netres per second per hour, with 95% confidence limits for the *population* median difference from 0.450 to 2.367 metres per second per hour. These limits are similar to (but not the same as) the confidence limits of 0.500 and 1.925 calculated by Sprent and Smeeton, using the original exact formula from Sen (1968). There is therefore only a small discrepancy, in this case, between the `censlope` method (which is more robust to heteroskedasticity) and the original Sen method (which is probably more robust to very small sample numbers).

We might want to use the Theil–Sen slope to draw a line through the data points, as in Figure 2. To do this, we need an intercept as well as a slope. A candidate for such an intercept, used by Sprent and Smeeton, is the median of the "residuals" $Y_j - \beta X_j$. We calculate this intercept, and use it to draw a line, as follows:

```
. egen intercept = median(resid)
. gene flowhat = flow - resid + intercept
. lab var flowhat "Fitted flow"
. twoway scatter flow hours || line flowhat hours, xlab(0(1)6) ylab(0(1)12)
```

First, we use `egen` to calculate the median of the variable `resid`, generated by the `ystargenerate()` option, which stores the "residuals". This median is stored in a new variable `intercept`. Then, we generate the fitted values of the flow rate in a new variable `flowhat`. These fitted values are plotted as a line against the time variable `hours`, and the observed flow values are superimposed to create the graph of Figure 2.

Note that the line drawn using the Theil–Sen slope passes close to the first 6 data points, and is not influenced greatly by the "outlier" at 6 hours after the start of the thaw. By contrast, the conventional least–squares regression slope has intercept 1.507 cubic metres per second and slope 1.107 metres per second per hour, because of the influence of the "outlier". This lack of sensitivity to tiny minorities of "outliers" is often seen as an advantage of the Theil–Sen slope.

4 Acknowledgements

I would like to thank my former colleague Nigel Smeeton of King's College London, UK for drawing my attention to the Theil–Sen median slope, and for very kindly giving me a copy of his very useful textbook (co–authored with Peter Sprent) on rank methods (Sprent and Smeeton, 2007). I would also like to thank all at StataCorp for providing the Mata programming language, without which the present version of `somersd`, including `censlope`, would not have been easy to write.

5 Historical note

This document is part of a post-publication update, arising originally from two articles, which appeared in the Stata Technical Bulletin (STB) as Newson (2000a) and Newson (2000b). These introduced the modules `somersd` and `cendif`, respectively, both of which are parts of the `somersd` package. Post-publication updates of those STB articles are now distributed with the `somersd` package as `somersd.pdf` and `cendif.pdf`, which are manuals for the `somersd` and `cendif` modules. This document `censlope.pdf`, written in the same format as the other two manuals, was added later, after `censlope` was written in May 2006. The definitive reference to be cited for the methods of `censlope` is Newson (2006c).

6 References

Altman, D. G., D. Machin, T. N Bryant, and M. J Gardner. 2000. *Statistics with Confidence*. London, UK: British Medical Journal.

Campbell, M. J. and M. J. Gardner. 1988. Calculating confidence intervals for some non–parametric analyses. *British Medical Journal* 296: 1454–1456.

Conover, W. J. 1999. *Practical Nonparametric Statistics*. 3rd ed. New York, NY: John Wiley & Sons.

Edwardes, M. D. deB. 1995. A confidence interval for $\Pr(X < Y)$ - $\Pr(X > Y)$ estimated from simple cluster samples. *Biometrics* 51: 571–578.

Hodges, J. L. and E. L. Lehmann. 1963. Estimates of location based on rank tests. *The Annals of Mathematical Statistics* 34: 598–611.

Imai, K. and D. A. van Dyk. Causal inference with general treatment regimes: generalizing the propensity score. 2004. *Journal of the American Statistical Association* 99(467): 854–866.

Kendall, M. G. and J. D. Gibbons. 1990. *Rank Correlation Methods. 5th edition.* New York, NY: Oxford University Press.

Lehmann, E. L. 1963. Nonparametric confidence intervals for a shift parameter. *The Annals of Mathematical Statistics* 34(4): 1507–1512.

Moser, B. K., G. R. Stevens and C. L. Watts. 1989. The two–sample t–test versus Satterthwaite's approximate F–test. *Communications in Statistics – Theory and Methods* 18(11): 3963–3975.

Moser, B. K. and G. R. Stevens. 1992. Homogeneity of variance in the two–sample means test. *The American Statistician* 46(1): 19–21.

Newson, R. 2000a. snp15: `somersd` – Confidence intervals for nonparametric statistics and their differences. *Stata Technical Bulletin* 55: 47–55. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 312–322.

Newson, R. 2000b. snp16: Robust confidence intervals for median and other percentile differences between groups. *Stata Technical Bulletin* 58: 30–35. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 324–331.

Newson, R. 2002. Parameters behind "nonparametric" statistics: Kendall's tau, Somers' D and median differences. *Stata Journal* 2(1): 45–64. A pre-publication draft can be downloaded from Roger Newson's website at `http://www.imperial.ac.uk/nhli/r.newson/` as of 30 May 2012, using the `net` command in Stata.

Newson, R. 2006a. Efficient calculation of jackknife confidence intervals for rank statistics. *Journal of Statistical Software* 15(1): 1–10. Downloadable from `http://www.jstatsoft.org/` as of 30 May 2012.

Newson, R. 2006b. Confidence intervals for rank statistics: Somers' D and extensions. *Stata Journal* 6(3): 309–334. A pre-publication draft can be downloaded from Roger Newson's website at `http://www.imperial.ac.uk/nhli/r.newson/` as of 30 May 2012, using the `net` command in Stata.

Newson, R. 2006c. Confidence intervals for rank statistics: Percentile slopes, differences, and ratios. *Stata Journal* 6(4): 497–520. A pre-publication draft can be downloaded from Roger Newson's website at `http://www.imperial.ac.uk/nhli/r.newson/` as of 30 May 2012, using the `net` command in Stata.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery. 1992. *Numerical recipes in C: the art of scientific computing. 2nd edition.* Cambridge, UK: Cambridge University Press.

Ridders, C. J. F. 1979. A new algorithm for computing a single root of a real continuous function. *IEEE Transactions on Circuits and Systems*, vol. CAS–26(11): 979–980.

Rosenbaum P. R. and D. B. Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70(1): 41–55.

Royston, P. 1998. CID: Stata module to calculate confidence intervals for means or differences. On the Ideas list at `http://ideas.repec.org/c/boc/bocode/s338001.html` as of 30 May 2012.

Sen, P. K. 1968. Estimates of the regression coefficient based on Kendall's tau. *Journal of the American Statistical Association* 63(324): 1379-1389.

Somers, R. H. 1962. A new asymmetric measure of association for ordinal variables. *American Sociological Review* 27: 799-811.

Sprent, P. and N. C. Smeeton. 2007. *Applied nonparametric statistical methods. 4th edition.* Boca Raton, FA: Chapman & Hall/CRC.

Theil, H. 1950. A rank invariant method of linear and polynomial regression analysis, I, II, III. *Proceedings of the Koninklijke Nederlandse Akademie Wetenschappen, Series A – Mathematical Sciences* 53: 386–392, 521–525, 1397–1412.

Wang, D. 1999. sg123: Hodges–Lehmann estimation of a shift in location between two populations. *Stata Technical Bulletin* 52: 52–53. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 255–257.

Wilcox, R. R. 1998. A note on the Theil-Sen regression estimator when the regressor is random and the error term is heteroscedastic. *Biometrical Journal* 40(3): 261–268.