`kwstat`: Kernel-weighted sample statistics

Florian Wendelspiess Chávez Juárez*

July 22, 2014

Version 1.0

Abstract

This manual describes the user written Stata® command `kwstat` and provides several examples. `kwstat` stands for **k**ernel **w**eighted **stat**istics and is an ad-hoc method to visualize the behavior a variable y in function of another variable x. It is based on kernel weighted sample statistics such as the mean (Nadaraya-Watson estimator) but also standard deviation, skewness, kurtosis, deciles, etc. The method is not statistically founded and should only be used for illustration and visual data analysis.

Keywords: Stata, non-parametric estimation, kernel weighted estimation, visual data analysis, sample statistics

JEL-Classification: C14, C19

Contents

*University of Geneva. florian@chavezjuarez.com

1 Introduction and methods

The use of non-parametric methods has increased over the last years. One reason for this increase is the availability of larger datasets with a larger number of observations. A large number of observations is required for many non-parametric estimators. One of the most popular non-parametric estimators is the Nadaraya-Watson estimator of the conditional mean (Nadaraya, 1964; Watson, 1964). The Nadaraya-Watson estimator for one explanatory variable is implemented in Stata® with the command `lpoly`. The Nadaraya-Watson estimator allows us to quickly obtain an overview of the relationship between a variable y and another variable x without assuming any functional form.

The intuition is relatively easy. Assume we want to estimate the average of y when $x = x_0$. This is we want to estimate $E[y|x = x_0]$. We could simply take all the observations where $x = x_0$ and then take the mean of y. This approach is possible if we have many observations for each level of x_0. However, if x is continuous we might have very little observations for each level of x. Therefore the kernel weighted approach allows us to consider some neighboring values of x_0. The kernel function gives them a weight which dependent on the distance to x_0. The further away an observation is, the less weight it becomes. Then we simply take the kernel weighted average of y and repeat this of many different level of x_0.

The idea of `kwstat` is to do exactly the same, but instead of focusing only on the mean of y, it proposes to compute also other statistics. For instance, if we would like to estimate the standard deviation of y in function of x we simply compute the kernel weight around x_0 and compute the standard deviation using these weights.

IMPORTANT! Note that this extension to statistics other than the mean is not rooted in any discussion in the statistical literature. Therefore is should be considered to be an ad-hoc approach to the visualization of conditional statistics.

1.1 What does `kwstat` exactly do?

`kwstat` does a very similar job as `lpoly` but instead of computing the mean it computes other statistics. Practically the following steps are performed:

1. Several values of the x-axis are selected. By default the range is divided into 100 equally distributed distances. The user can also chose to select each possible value of the x-variable (this can be very computationally heavy for large datasets and continuous variables).

2. For each selected point x_0 the kernel weight is computed of each neighboring observation is computed.

3. Different sample statistics are computed at each x_0 using the kernel weights. Technically this computation is performed using the Stata® command `tabstat` with `aweights`.

2 The kwstat command

2.1 Syntax

The syntax of kwstat is as follows:

kwstat *yvar xvar* [*if*] [*in*] [, bw(*real*) lpolybw kernel(*str*) stats(*str*) at grid(*int*) save prefix(*str*) nograph graphtype(*str*) graphoptions(*str*)]

where *yvar* is the outcome variable (e.g. wages) and *xvar* is the explanatory (or x-axis) variable. Note that only one explanatory variable is allowed. The command kwstat allows limiting the sample using the *if* or *in* statement. In contrast, the current version does not support sample weights. Let us now have a look at the options.

2.2 Options

kwstat has a series of options allowing the researcher to adapt the analysis to his or her needs.

bw(*real*) allows you to define the bandwidth of the kernel used in the computation. If nothing is specified, kwstat uses $h = 1.06\sigma_x n^{-\frac{1}{5}}$, which is an approximation of the optimal bandwidth for the Gaussian kernel when estimating the Nadaraya-Watson estimator. Hence, it is not necessarily the optimal bandwidth for the estimation of other statistics and/or other kernels. I strongly advise users to try several bandwidths and to compare them visually. See also section 2.4, where I discuss the issue of the bandwidth with some more details.

lpolybw is a second option to modify the optimal bandwidth with respect to the default value. By activating this option the optimal bandwidth proposed by the command lpoly is used. Note, however, that this is only possible for the kernels implemented in lpoly (e.g. it's not possible for the logit kernel) and that these optimized bandwidths were designed for the kernel regression (estimation of the mean). They are therefore not necessarily optimal for the estimation of other statistics. I strongly advise users to try several bandwidths and to compare them visually. See also section 2.4, where I discuss the issue of the bandwidth with some more details.

kernel(*str*) allows you to change the type of kernel you would like to use. By default the *epanechnikov* kernel is used. The following options are available:

Argument	Description	Formula
normal	Normal kernel	$K(z) = \Phi(z)$
gaussian	Gaussian kernel (see normal kernel)	
triangle	Triangle kernel	$K(z) = \begin{cases} 1-\lvert z\rvert & \text{if } \lvert z\rvert \leq 1 \\ 0 & \text{otherwise} \end{cases}$
beta	Beta kernel	$K(z) = \begin{cases} 0.75(1-z)(1+z) & \text{if } \lvert z\rvert \leq 1 \\ 0 & \text{otherwise} \end{cases}$
logit	Logit kernel	$K(z) = \frac{\frac{exp(z)}{1+exp(z)}}{1-\frac{exp(z)}{1+exp(z)}}$
uniform	Uniform kernel	$K(z) = \begin{cases} 0.5 & \text{if } \lvert z\rvert \leq 1 \\ 0 & \text{otherwise} \end{cases}$
cosine	Cosine kernel	$K(z) = \begin{cases} 1+cos(2\pi z) & \text{if } \lvert z\rvert \leq 0.5 \\ 0 & \text{otherwise} \end{cases}$
parzen	Parzen kernel	$K(z) = \begin{cases} \frac{4}{3}-8z^2+8\lvert z\rvert^3 & \text{if } \lvert z\rvert < \frac{1}{2} \\ \frac{8}{3}(1-\lvert z\rvert)^3 & \text{if } \frac{1}{2} < \lvert z\rvert \leq 1 \\ 0 & \text{otherwise} \end{cases}$
default	Epanechnikov	$K(z) = \begin{cases} \frac{3}{4}\frac{(1-0.5z^2)}{\sqrt{5}} & \text{if } \lvert z\rvert \leq \sqrt{5} \\ 0 & \text{otherwise} \end{cases}$

`stats(str)` lets you specify the statistics you would like to compute. The default value is the mean, providing equivalent results to `lpoly`. All statistics supported by the command `tabstat` are supported: `mean, sd, min, max, range, kurtosis, skewness, semean, p10, p95` etc. (e.g. see `help tabstat` for the full list). Several statistics can be selected together, for instance `stats(mean sd p10)` returns the mean, the standard deviation and the first decile of y in function of x.

`at` by default the statistics are evaluated on an equally spaced grid at 100 points. Using the option `at` you can change this and let `kwstat` compute the statistics for each value of the variable x. Note, however, that his can become computationally heavy when many different values are present. You can also change the number of points on the equally spaced grid using the option `grid(int)`

`grid(int)` allows you to change the number of points for which the statistics are computed. By default, 100 points on the equally spaced grid between the minimum and the maximum of *xvar* are used.

`save` allows you to store the variables produces by `kwstat` . This can be helpful for posterior use (e.g. to create your own graph). You can also use the option `prefix` to change the name of the variables. By *default* the variables are names `_kwstats_stat` where *stat* refers to the statistic chosen in the option `stats`.

`prefix(str)` allows you to change the prefix of the generated variable from the default value of `_kwstat_` to the prefix of your preference.

nograph suppresses the output of the graph. Note that this makes only sense if you also use the option save to keep the computed variables for posterior use. Otherwise the command would not produce any output.

graphtype(*str*) lets you change the graph type. By default a *line* graph is provided. You can change it to *scatter* or *connected*.

graphoptions(*str*) allows you to provide a string containing options for the twoway graph. You can for instance change the legend or the label of the axis. See help twoway for more details.

2.3 Returned results

kwstat is an r-class command and returns a single scalar containing the bandwidth and a macro with the chosen kernel:

Scalars	
r(bw)	Bandwidth
Macro	
r(kernel)	Kernel

Moreover, by using the option save the user can output the estimated values of the statistics for posterior use.

2.4 Optimal bandwidth

It is crucial to correctly choose the bandwidth when using kernel estimates. For the case of kernel regression (see lpoly) a well-established literature provides ways to compute the optimal bandwidth. In contrast, the procedures proposed by kwstat are ad-hoc methods which are not based on an established statistical literature. Therefore, no optimal bandwidth computation is available. The primary goal of the routine is to provide visual data analysis, which can be done without having an optimized bandwidth. For this reason, I suggest to start with either the default bandwidth (optimized for the Nadaraya-Watson estimator and the normal kernel) or the bandwidth proposed by lpoly. Following this first estimate, the user should use different bandwidths to see how sensitive the estimated curve is to this value.

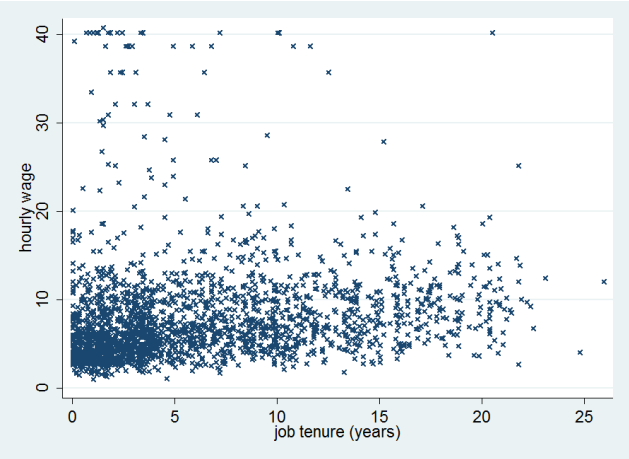
3 Examples and illustrations

In this section I provide some examples with both simulated and real data. The idea is to provide an idea of what kwstat can do and to clearly highlight the limits. The do-files used to create all examples in this section can be found in the appendix A of this manual.

3.1 Relationship between wages and tenure or age

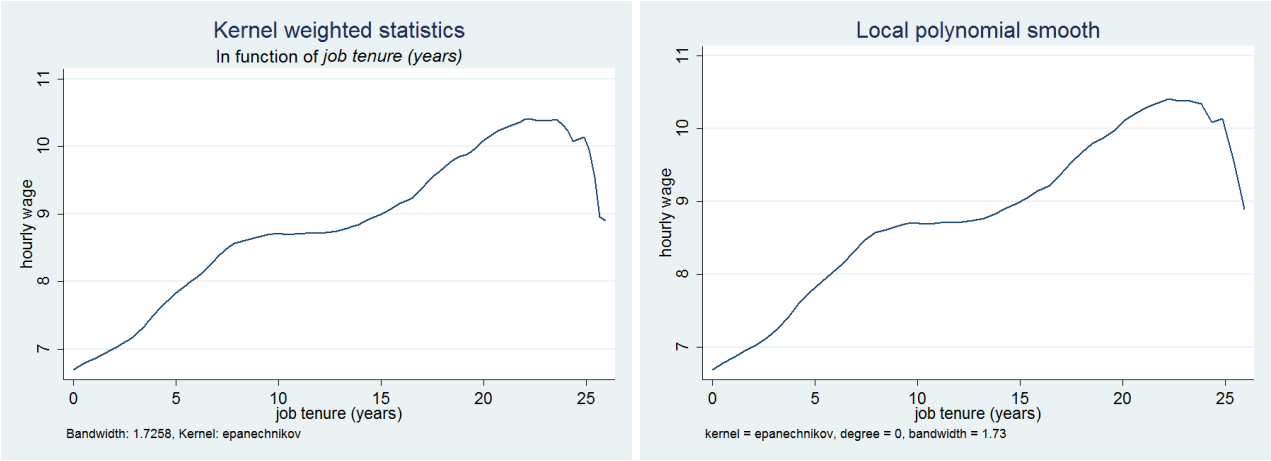
In the first example I use the dataset `nlsw88` provided by Stata® . You can simply load it by typing `sysuse nlsw88, clear`. First, let us have a look at the data itself. Figure 1 displays the scatter plot of wages in function of tenure.

Figure 1: Scatter plot of wages by tenure



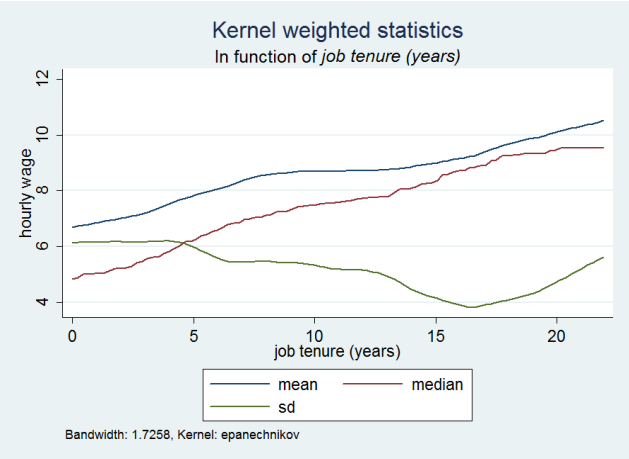
Admittedly it is difficult to see much from this graph due to the large number of observations and the large dispersion. Thus, a natural way to have a closer look at the data is to plot the Nadaraya-Watson estimator of the mean. Figure 2 displays this estimator for both commands `kwstat` (left) and `lpoly` (right). We can see that both produce basically the same figure. The small differences are due

Figure 2: Nadaraya-Watson estimator using `kwstat` and `lpoly`



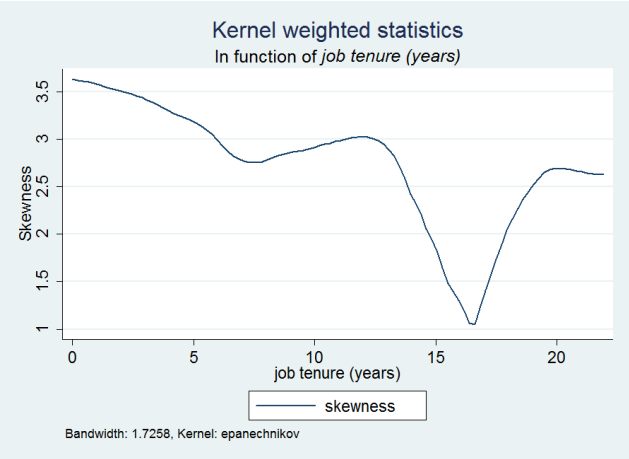
to a different number of points for which the mean was estimated. Now, it could also be interesting to look at more statistics than just the mean. `kwstat` allows you to produce a similar graph for other statistics. For instance, we might want to look at the first decile, the standard deviation and the median. Figure 3 displays these three statistics in function of tenure along with the Nadaraya-Watson estimator seen before.

Figure 3: Different statistics of wage in function of tenure



From this graph we get a more complete view of the wages in function of tenure. For instance, we can see that the median is always below the mean for all values of tenure but that the gap between the two in getting smaller. This might also suggest that the dispersion becomes smaller when tenure increases. This suggestion is confirmed by the standard deviation which becomes smaller when increasing tenure. When looking at the skewness of the wage distribution in function of tenure we can observe an interesting behavior reported in figure 4. We can see that the skewness is always positive but decreasing

Figure 4: Skewness of wages in function of tenure



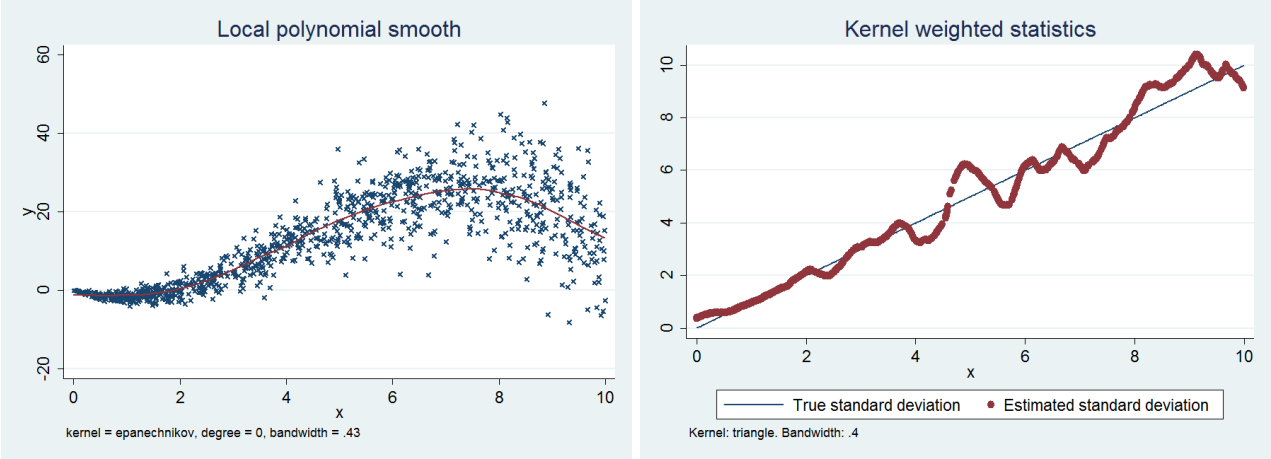
with tenure, except for tenure above 17 years where the skewness suddenly increases again.
It is important to notice that `kwstat` is intended for an exploratory use by the researcher. The goal is not to use these graphs directly in research output, because the method lack of statistical foundation.

3.2 Simulated data

Let us now have a look at simulated data, where y is a polynomial function of degree 3 of x and an error term. The error term has a zero mean and an increasing variance at the lower and the upper

tail of x. Figure 5 displays on the left side the scatter plot and the Nadaraya-Watson estimator (using `lpoly`) and on the right side the output of `kwstat` .

Figure 5: Nadaraya-Watson estimator and output of `kwstat`



From the Nadaraya-Watson estimator we can clearly see the polynomial shape of the mean. We also see that the variance of the disturbance term increases with x. We can now use the `kwstat` to focus on this variance. The standard deviation estimated with `kwstat` is displayed on the right graph of figure 5. Given that these graphs are based on simulated data, I can also display the true standard deviation of the error term.

4 Installation and updates

The most convenient way to install `kwstat` is to use the command `scc install`. Simply type:

```
ssc install kwstat
```

Alternatively you can also download the package from the author's website and put the files in the correct folder. Typically this folder is `C:/ado/plus/b/`

4.1 Update

The simplest way to update `kwstat` is to run

```
ssc install kwstat, replace
```

You can also check if new updates are available by clicking on the version at the top right of the help file. The link will open a website with the information on updates. You can also check if this manual is the newest version by visiting:

http://www.econ.chavezjuarez.com/vcheck.php?i=kwstat_manual&v=1.0

References

Nadaraya, E.A., "On Estimating Regression," *Theory of Probability & Its Applications*, 1964, *9* (1), pp.141–142.

Watson, Geoffrey S., "Smooth Regression Analysis," *Sankhyā: The Indian Journal of Statistics, Series A*, 1964, *26* (4), pp. 359–372.

A Do-files used in this manual

A.1 Do file of section 3.1

```
clear all
sysuse nlsw88 // Load the sample data

// Figure 1
     scatter wage tenure, msymbol(x)

// Figure 2 left
     kwstat wage tenure, lpoly

// Figure 2 right
     lpoly wage tenure, noscatter

// Figure 3 (note that I limit to tenure<22)
     kwstat wage tenure if tenure<22, lpoly stats(mean median sd)

// Figure 4 (note that I limit to tenure<22)
     kwstat wage tenure if tenure<22, lpoly stats(skewness) graphoptions( ytitle("Skewness") legend(on))
```

A.2 Do file of section 3.2

```
clear all
set seed 1234 // fix the random seed to reproduce the same graph
set obs 1000 // generate 10K observations
gen x = uniform()*10 // define x ~ U[0,10]

drawnorm e // define e ~ N[0,1]
replace e = e*x // modify the error term
gen e_sd=1*x // true std. dev. of e

gen y = -4*x +2.5*x^2-0.2*x^3 + e  // generate y as a polynomial function of degree 3 of x and e

// show the Nadaraya-Watson estimator (over the scatter plot): figure 5 left
     lpoly y x, msymbol(x)

// show the standard deviation y in function of x
     kwstat y x,  at stats(sd) graphtype(scatter) save prefix(est_)  bw(0.4)

// combine the estimated with the true value (Figure 5 right))
twoway (line e_sd x)(scatter est_sd x), ///
      legend(order(1 "True standard deviation" 2 "Estimated standard deviation")) ///
      title("Kernel weighted statistics") note("Kernel: epanechnikov. Bandwidth: `r(bw)'")
```

B Versions

Version	Description
1.0	First release of `kwstat`