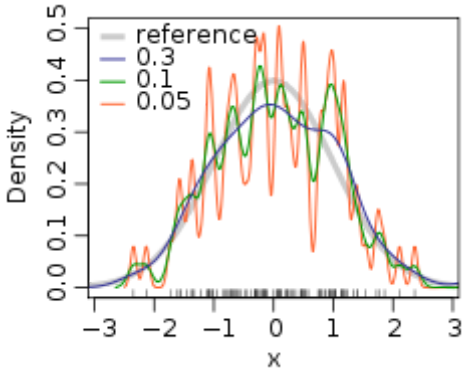
Kernel density estimation

From Wikipedia, the free encyclopedia

In statistics, **kernel density estimation (KDE)** is a non-parametric way to estimate the probability density function of a random variable. Kernel density estimation is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample. In some fields such as signal processing and econometrics it is also termed the *Parzen–Rosenblatt window* method, after Emanuel Parzen and Murray Rosenblatt, who are usually credited with independently creating it in its current form.[1][2]



Kernel density estimation of 100 normally distributed random numbers using different smoothing bandwidths.

Contents

Definition

Let $(x_1, x_2, \ldots, x_n)$ be an independent and identically distributed sample drawn from some distribution with an unknown density f. We are interested in estimating the shape of this function f. Its *kernel density estimator* is

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right),$$

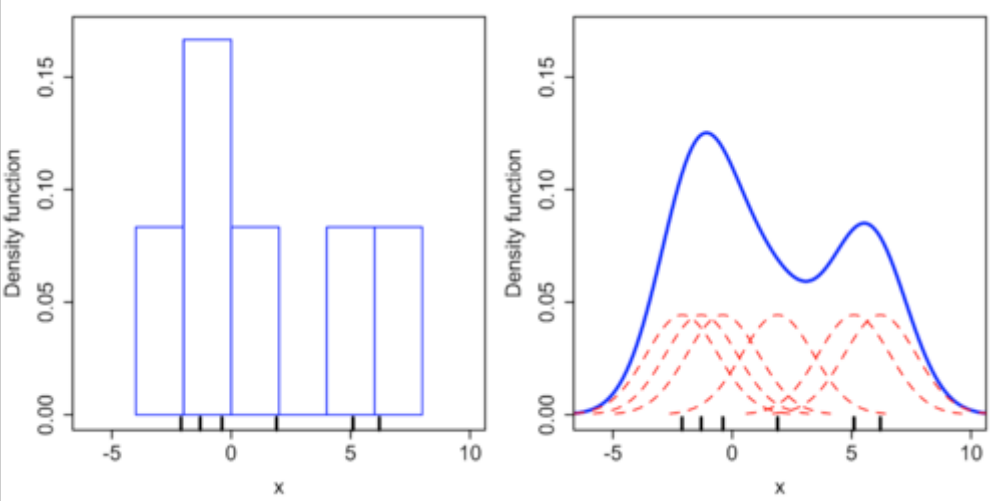
where $K(\bullet)$ is the kernel — a non-negative function that integrates to one and has mean zero — and $h > 0$ is a smoothing parameter called the *bandwidth*. A kernel with subscript h is called the *scaled kernel* and defined as $K_h(x) = 1/h\ K(x/h)$. Intuitively one wants to choose h as small as the data allow, however there is always a trade-off between the bias of the estimator and its variance; more on the choice of bandwidth below.

A range of kernel functions are commonly used: uniform, triangular, biweight, triweight, Epanechnikov, normal, and others. The Epanechnikov kernel is optimal in a mean square error sense,[3] though the loss of efficiency is small for the kernels listed previously,[4] and due to its convenient mathematical properties, the normal kernel is often used, which means $K(x) = \phi(x)$, where ϕ is the standard normal density function.

The construction of a kernel density estimate finds interpretations in fields outside of density estimation.[5] For example, in thermodynamics, this is equivalent to the amount of heat generated when heat kernels (the fundamental solution to the heat equation) are placed at each data point locations x_i. Similar methods are used to construct discrete Laplace operators on point clouds for manifold learning.

Kernel density estimates are closely related to histograms, but can be endowed with properties such as smoothness or continuity by using a suitable kernel. To see this, we compare the construction of histogram and kernel density estimators, using these 6 data points: $x_1 = -2.1$, $x_2 = -1.3$, $x_3 = -0.4$, $x_4 = 1.9$, $x_5 = 5.1$, $x_6 = 6.2$. For the histogram, first the horizontal axis is divided into sub-intervals or bins which cover the range of the data. In this case, we have 6 bins each of width 2. Whenever a data point falls inside this interval, we place a box of height 1/12. If more than one data point falls inside the same bin, we stack the boxes on top of each other.

For the kernel density estimate, we place a normal kernel with variance 2.25 (indicated by the red dashed lines) on each of the data points x_i. The kernels are summed to make the kernel density estimate (solid blue curve). The smoothness of the kernel density estimate is evident compared to the discreteness of the histogram, as kernel density estimates converge faster to the true underlying density for continuous random variables.[6]



Comparison of the histogram (left) and kernel density estimate (right) constructed using the same data. The 6 individual kernels are the red dashed curves, the kernel density estimate the blue curves. The data points are the rug plot on the horizontal axis.

Bandwidth selection

The bandwidth of the kernel is a free parameter which exhibits a strong influence on the resulting estimate. To illustrate its effect, we take a simulated random sample from the standard normal distribution (plotted at the blue spikes in the rug plot on the horizontal axis). The grey curve is the true density (a normal density with mean 0 and variance 1). In comparison, the red curve is *undersmoothed* since it contains too many spurious data artifacts arising from using a bandwidth $h = 0.05$, which is too small. The green curve is *oversmoothed* since using the bandwidth $h = 2$ obscures much of the underlying structure. The black curve with a bandwidth of $h = 0.337$ is considered to be optimally smoothed since its density estimate is close to the true density.

The most common optimality criterion used to select this parameter is the expected L_2 risk function, also termed the mean integrated squared error:

$$\operatorname{MISE}(h) = \operatorname{E}\left[\int (\hat{f}_h(x) - f(x))^2\,dx\right].$$

Under weak assumptions on f and K,[1][2] MISE (h) = AMISE(h) + $o(1/(nh) + h^4)$ where o is the little o notation. The AMISE is the Asymptotic MISE which consists of the two leading terms

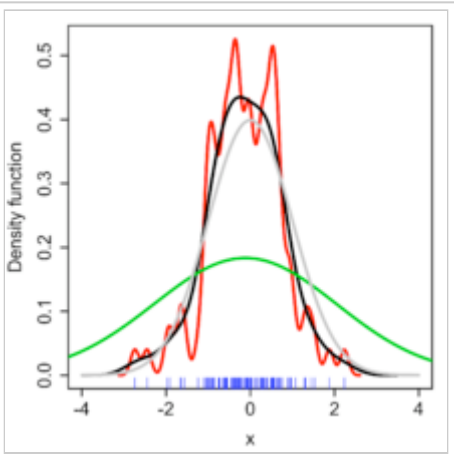
$$\operatorname{AMISE}(h) = \frac{R(K)}{nh} + \frac{1}{4} m_2(K)^2 h^4 R(f'')$$

where $R(g) = \int g(x)^2\,dx$ for a function g, $m_2(K) = \int x^2 K(x)\,dx$ and f'' is the second derivative of f. The minimum of this AMISE is the solution to this differential equation

$$\frac{\partial}{\partial h} \operatorname{AMISE}(h) = -\frac{R(K)}{nh^2} + m_2(K)^2 h^3 R(f'') = 0$$

or

$$h_{\operatorname{AMISE}} = \frac{R(K)^{1/5}}{m_2(K)^{2/5} R(f'')^{1/5} n^{1/5}}.$$



Kernel density estimate (KDE) with different bandwidths of a random sample of 100 points from a standard normal distribution. Grey: true density (standard normal). Red: KDE with h=0.05. Black: KDE with h=0.337. Green: KDE with h=2.

Neither the AMISE nor the h_{AMISE} formulas are able to be used directly since they involve the unknown density function f or its second derivative f'', so a variety of automatic, data-based methods have been developed for selecting the bandwidth. Many review studies have been carried out to compare their efficacies,[7][8][9][10][11][12][13] with the general consensus that the plug-in selectors[5] [14] and cross validation selectors[15][16][17] are the most useful over a wide range of data sets.

Substituting any bandwidth h which has the same asymptotic order $n^{-1/5}$ as h_{AMISE} into the AMISE gives that AMISE(h) = $O(n^{-4/5})$, where O is the big o notation. It can be shown that, under weak assumptions, there cannot exist a non-parametric estimator that converges at a faster rate than the kernel estimator.[18] Note that the $n^{-4/5}$ rate is slower than the typical n^{-1} convergence rate of parametric methods.

If the bandwidth is not held fixed, but is varied depending upon the location of either the estimate (balloon estimator) or the samples (pointwise estimator), this produces a particularly powerful method termed adaptive or variable bandwidth kernel density estimation.

Bandwidth selection for kernel density estimation of heavy-tailed distributions is said to be relatively difficult.[19]

A rule-of-thumb bandwidth estimator

If Gaussian basis functions are used to approximate univariate data, and the underlying density being estimated is Gaussian, the optimal choice for h (that is, the bandwidth that minimises the mean integrated squared error) is[20]

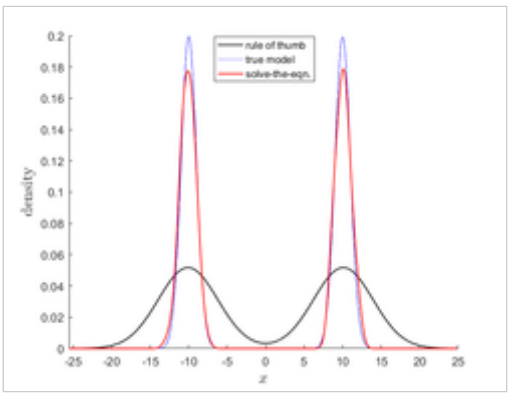
$$h = \left(\frac{4\hat{\sigma}^5}{3n} \right)^{\frac{1}{5}} \approx 1.06 \hat{\sigma} n^{-1/5},$$

where $\hat{\sigma}$ is the standard deviation of the samples. This approximation is termed the *normal distribution approximation*, Gaussian approximation, or *Silverman's (1986) rule of thumb*. While this rule of thumb is easy to compute, it should be used with caution as it can yield widely inaccurate estimates when the density is not close to being normal. For example, consider estimating the bimodal Gaussian mixture:

$$\frac{1}{2\sqrt{2\pi}}\exp(-(x-10)^2/2)+\frac{1}{2\sqrt{2\pi}}\exp(-(x+10)^2/2)$$

from a sample of 200 points. The figure on the right below shows the true density and two kernel density estimates --- one using the rule-of-thumb bandwidth, and the other using a solve-the-equation bandwidth.[5][14] The estimate based on the rule-of-thumb bandwidth is significantly oversmoothed. The Matlab script for this example uses kde.m (http://www.mathworks.com/matlabcentral/fileexchange/14034) and is given below.

```
randn('seed',1) % use for reproducibility
data=[randn(100,1)-10;randn(100,1)+10]; % normal mixture with two hum
n=length(data); % number of samples
h=std(data)*(4/3/n)^(1/5); % Silverman's rule of thumb
phi=@(x)(exp(-.5*x.^2)/sqrt(2*pi)); % normal pdf
ksden=@(x)mean(phi((x-data)/h)/h); % kernel density
fplot(ksden,[-25,25],'k') % plot kernel density with rule of thumb
hold on
fplot(@(x)(phi(x-10)/2+phi(x+10)/2),[-25,25],'b') % plot the true den
kde(data); % plot kde with solve-the-equation bandwidth
```



Comparison between rule of thumb and solve-the-equation bandwidth.

Relation to the characteristic function density estimator

Given the sample $(x_1, x_2, \ldots, x_n)$, it is natural to estimate the characteristic function $\varphi(t) = \mathrm{E}[e^{itX}]$ as

$$\hat{\varphi}(t) = \frac{1}{n}\sum_{j=1}^{n} e^{itx_j}$$

Knowing the characteristic function, it is possible to find the corresponding probability density function through the Fourier transform formula. One difficulty with applying this inversion formula is that it leads to a diverging integral, since the estimate $\hat{\varphi}(t)$ is unreliable for large t's. To circumvent this problem, the estimator $\hat{\varphi}(t)$ is multiplied by a damping function $\psi_h(t) = \psi(ht)$, which is equal to 1 at the origin and then falls to 0 at infinity. The "bandwidth parameter" h controls how fast we try to dampen the function $\hat{\varphi}(t)$. In particular when h is small, then $\psi_h(t)$ will be approximately one for a large range of t's, which means that $\hat{\varphi}(t)$ remains practically unaltered in the most important region of t's.

The most common choice for function ψ is either the uniform function $\psi(t) = \mathbf{1}\{-1 \le t \le 1\}$, which effectively means truncating the interval of integration in the inversion formula to $[-1/h, 1/h]$, or the gaussian function $\psi(t) = e^{-\pi t^2}$. Once the function ψ has been chosen, the inversion formula may be applied, and the density estimator will be

$$\begin{aligned}\hat{f}(x) &= \frac{1}{2\pi}\int_{-\infty}^{+\infty}\hat{\varphi}(t)\psi_h(t)e^{-itx}\,dt = \frac{1}{2\pi}\int_{-\infty}^{+\infty}\frac{1}{n}\sum_{j=1}^{n}e^{it(x_j-x)}\psi(ht)\,dt \\ &= \frac{1}{nh}\sum_{j=1}^{n}\frac{1}{2\pi}\int_{-\infty}^{+\infty}e^{-i(ht)\frac{x-x_j}{h}}\psi(ht)\,d(ht) = \frac{1}{nh}\sum_{j=1}^{n}K\Big(\frac{x-x_j}{h}\Big),\end{aligned}$$

where K is the Fourier transform of the damping function ψ. Thus the kernel density estimator coincides with the characteristic function density estimator.

Statistical implementation

A non-exhaustive list of software implementations of kernel density estimators includes:

- In Analytica release 4.4, the *Smoothing* option for PDF results uses KDE, and from expressions it is available via the built-in `Pdf` function.
- In C/C++, FIGTree (http://www.umiacs.umd.edu/~morariu/figtree/) is a library that can be used to compute kernel density estimates using normal kernels. MATLAB interface available.
- In C++, libagf (http://libagf.sf.net) is a library for variable kernel density estimation.
- In CrimeStat, kernel density estimation is implemented using five different kernel functions – normal, uniform, quartic, negative exponential, and triangular. Both single- and dual-kernel density estimate routines are available. Kernel density estimation is also used in interpolating a Head Bang routine, in estimating a two-dimensional Journey-to-crime density function, and in estimating a three-dimensional Bayesian Journey-to-crime estimate.
- In ELKI, kernel density functions can be found in the package `de.lmu.ifi.dbs.elki.math.statistics.kernelfunctions`
- In ESRI products, kernel density mapping is managed out of the Spatial Analyst toolbox and uses the Quartic(biweight) kernel.
- In Excel, the Royal Society of Chemistry has created an add-in to run kernel density estimation based on their Analytical Methods Committee Technical Brief 4 (http://www.rsc.org/Membership/Networking/InterestGroups/Analytical/AMC/Software/kerneldensities.asp).
- In gnuplot, kernel density estimation is implemented by the `smooth kdensity` option, the datafile can contain a weight and bandwidth for each point, or the bandwidth can be set automatically[21] according to "Silverman's rule of thumb" (see above).
- In Haskell, kernel density is implemented in the statistics (http://hackage.haskell.org/package/statistics) package.
- In Java, the Weka (machine learning) package provides weka.estimators.KernelEstimator (http://weka.sourceforge.net/doc.stable/weka/estimators/KernelEstimator.html), among others.
- In JavaScript, the visualization package D3.js offers a KDE package in its science.stats package.
- In JMP, The Distribution platform can be used to create univariate kernel density estimates, and the Fit Y by X platform can be used to create bivariate kernel density estimates.
- In Julia, kernel density estimation is implemented in the KernelDensity.jl (https://github.com/JuliaStats/KernelDensity.jl) package.
- In MATLAB, kernel density estimation is implemented through the `ksdensity` function (Statistics Toolbox). This function does not provide an automatic data-driven bandwidth but uses a rule of thumb, which is optimal only when the target density is normal. A free MATLAB software package which implements an automatic bandwidth selection method[5] is available from the MATLAB Central File Exchange for
  - 1-dimensional data (http://www.mathworks.com/matlabcentral/fileexchange/14034-kernel-density-estimator)
  - 2-dimensional data (http://www.mathworks.com/matlabcentral/fileexchange/17204-kernel-density-estimation)
  - n-dimensional data (http://www.mathworks.com/matlabcentral/fileexchange/58312-kernel-density-estimator-for-high-dimensions)
    A free MATLAB toolbox with implementation of kernel regression, kernel density estimation, kernel estimation of hazard function and many others is available on these pages (http://www.math.muni.cz/english/science-and-research/developed-software/232-matlab-toolbox.html) (this toolbox is a part of the book [22]).
- In Mathematica, numeric kernel density estimation is implemented by the function `SmoothKernelDistribution` here (http://reference.wolfram.com/mathematica/ref/SmoothKernelDistribution.html) and symbolic estimation is implemented using the function `KernelMixtureDistribution` here (http://reference.wolfram.com/mathematica/ref/KernelMixtureDistribution.html) both of which provide data-driven bandwidths.

- In Minitab, the Royal Society of Chemistry has created a macro to run kernel density estimation based on their Analytical Methods Committee Technical Brief 4 (http://www.rsc.org/Membership/Networking/InterestGroups/Analytical/AMC/Software/kerneldensities.asp).
- In the NAG Library, kernel density estimation is implemented via the `g10ba` routine (available in both the Fortran[23] and the C[24] versions of the Library).
- In Nuklei (http://nuklei.sourceforge.net/), C++ kernel density methods focus on data from the Special Euclidean group $\boldsymbol{SE(3)}$.
- In Octave, kernel density estimation is implemented by the `kernel_density` option (econometrics package).
- In Origin, 2D kernel density plot can be made from its user interface, and two functions, Ksdensity for 1D and Ks2density for 2D can be used from its LabTalk (http://wiki.originlab.com/~originla/ltwiki/index.php?title=Category:LabTalk_Programming), Python, or C code.
- In Perl, an implementation can be found in the Statistics-KernelEstimation module (http://search.cpan.org/~janert/Statistics-KernelEstimation-0.05)
- In Python, many implementations exist: pyqt_fit.kde Module (http://pythonhosted.org/PyQt-Fit/mod_kde.html) in the PyQt-Fit package (https://pypi.python.org/packages/source/P/PyQt-Fit/PyQt-Fit-1.3.4.tar.gz), SciPy (`scipy.stats.gaussian_kde`), Statsmodels (`KDEUnivariate` and `KDEMultivariate`), and Scikit-learn (`KernelDensity`) (see comparison[25]).
- In R, it is implemented through the `density`, the `bkde` function in the KernSmooth library (https://cran.r-project.org/web/packages/KernSmooth/index.html) and the pareto density estimation in the `ParetoDensityEstimation` function AdaptGauss library (https://cran.r-project.org/web/packages/AdaptGauss/index.html) (the first two included in the base distribution), the `kde` function in the ks library (https://cran.r-project.org/web/packages/ks/index.html), the `dkden` and `dbckden` functions in the evmix library (https://cran.r-project.org/web/packages/evmix/index.html) (latter for boundary corrected kernel density estimation for bounded support), the `npudens` function in the np library (https://cran.r-project.org/web/packages/np/index.html) (numeric and categorical data), the `sm.density` function in the sm library (https://cran.r-project.org/web/packages/sm/index.html). For an implementation of the `kde.R` function, which does not require installing any packages or libraries, see kde.R (http://web.maths.unsw.edu.au/~zdravkobotev/kde.R).
- In SAS, `proc kde` can be used to estimate univariate and bivariate kernel densities.
- In Stata, it is implemented through `kdensity`;[26] for example `histogram x, kdensity`. Alternatively a free Stata module KDENS is available from here (https://ideas.repec.org/c/boc/bocode/s456410.html) allowing a user to estimate 1D or 2D density functions.
- In Apache Spark, you can use the `KernelDensity()` class (see official documentation for more details [1] (http://spark.apache.org/docs/latest/mllib-statistics.html#kernel-density-estimation))

See also

Wikimedia Commons has media related to ***Kernel density estimation***.

- Kernel (statistics)
- Kernel smoothing
- Kernel regression
- Density estimation (with presentation of other examples)
- Mean-shift
- Scale space The triplets {(x, h, KDE with bandwidth h evaluated at x: all x, $h > 0$} form a scale space representation of the data.
- Multivariate kernel density estimation
- Variable kernel density estimation

References

1. Rosenblatt, M. (1956). "Remarks on Some Nonparametric Estimates of a Density Function". *The Annals of Mathematical Statistics*. **27** (3): 832. doi:10.1214/aoms/1177728190.
2. Parzen, E. (1962). "On Estimation of a Probability Density Function and Mode". *The Annals of Mathematical Statistics*. **33** (3): 1065. doi:10.1214/aoms/1177704472. JSTOR 2237880.

3. Epanechnikov, V.A. (1969). "Non-parametric estimation of a multivariate probability density". *Theory of Probability and its Applications*. **14**: 153–158. doi:10.1137/1114019.
4. Wand, M.P; Jones, M.C. (1995). *Kernel Smoothing*. London: Chapman & Hall/CRC. ISBN 0-412-55270-1.
5. Botev, Z.I.; Grotowski, J.F.; Kroese, D.P. (2010). "Kernel density estimation via diffusion". *Annals of Statistics*. **38** (5): 2916–2957. doi:10.1214/10-AOS799.
6. Scott, D. (1979). "On optimal and data-based histograms". *Biometrika*. **66** (3): 605–610. doi:10.1093/biomet/66.3.605.
7. Park, B.U.; Marron, J.S. (1990). "Comparison of data-driven bandwidth selectors". *Journal of the American Statistical Association*. **85** (409): 66–72. doi:10.1080/01621459.1990.10475307. JSTOR 2289526.
8. Park, B.U.; Turlach, B.A. (1992). "Practical performance of several data driven bandwidth selectors (with discussion)". *Computational Statistics*. **7**: 251–270.
9. Cao, R.; Cuevas, A.; Manteiga, W. G. (1994). "A comparative study of several smoothing methods in density estimation". *Computational Statistics and Data Analysis*. **17** (2): 153–176. doi:10.1016/0167-9473(92)00066-Z.
10. Jones, M.C.; Marron, J.S.; Sheather, S. J. (1996). "A brief survey of bandwidth selection for density estimation". *Journal of the American Statistical Association*. **91** (433): 401–407. doi:10.2307/2291420. JSTOR 2291420.
11. Sheather, S.J. (1992). "The performance of six popular bandwidth selection methods on some real data sets (with discussion)". *Computational Statistics*. **7**: 225–250, 271–281.
12. Agarwal, N.; Aluru, N.R. (2010). "A data-driven stochastic collocation approach for uncertainty quantification in MEMS". *International Journal for Numerical Methods in Engineering*. **83** (5): 575–597.
13. Xu, X.; Yan, Z.; Xu, S. (2015). "Estimating wind speed probability distribution by diffusion-based kernel density method". *Electric Power Systems Research*. **121**: 28–37.
14. Sheather, S.J.; Jones, M.C. (1991). "A reliable data-based bandwidth selection method for kernel density estimation". *Journal of the Royal Statistical Society, Series B*. **53** (3): 683–690. JSTOR 2345597.
15. Rudemo, M. (1982). "Empirical choice of histograms and kernel density estimators". *Scandinavian Journal of Statistics*. **9** (2): 65–78. JSTOR 4615859.
16. Bowman, A.W. (1984). "An alternative method of cross-validation for the smoothing of density estimates". *Biometrika*. **71** (2): 353–360. doi:10.1093/biomet/71.2.353.
17. Hall, P.; Marron, J.S.; Park, B.U. (1992). "Smoothed cross-validation". *Probability Theory and Related Fields*. **92**: 1–20. doi:10.1007/BF01205233.
18. Wahba, G. (1975). "Optimal convergence properties of variable knot, kernel, and orthogonal series methods for density estimation". *Annals of Statistics*. **3** (1): 15–29. doi:10.1214/aos/1176342997.
19. Buch-Larsen, TINE (2005). "Kernel density estimation for heavy-tailed distributions using the Champernowne transformation". *Statistics*. **39** (6): 503–518. doi:10.1080/02331880500439782.
20. Silverman, B.W. (1998). *Density Estimation for Statistics and Data Analysis*. London: Chapman & Hall/CRC. p. 48. ISBN 0-412-24620-1.
21. Janert, Philipp K (2009). *Gnuplot in action : understanding data with graphs*. Connecticut, USA: Manning Publications. ISBN 978-1-933988-39-9. See section 13.2.2 entitled *Kernel density estimates*.
22. Horová, I.; Koláček, J.; Zelinka, J. (2012). *Kernel Smoothing in MATLAB: Theory and Practice of Kernel Smoothing*. Singapore: World Scientific Publishing. ISBN 978-981-4405-48-5.
23. The Numerical Algorithms Group. "NAG Library Routine Document: nagf_smooth_kerndens_gauss (g10baf)" (PDF). *NAG Library Manual, Mark 23*. Retrieved 2012-02-16.
24. The Numerical Algorithms Group. "NAG Library Routine Document: nag_kernel_density_estim (g10bac)" (PDF). *NAG Library Manual, Mark 9*. Retrieved 2012-02-16.
25. Vanderplas, Jake (2013-12-01). "Kernel Density Estimation in Python". Retrieved 2014-03-12.
26. https://www.stata.com/manuals13/rkdensity.pdf

External links

- Introduction to kernel density estimation (http://www.mvstat.net/tduong/research/seminars/seminar-2001-05) A short tutorial which motivates kernel density estimators as an improvement over histograms.
- Kernel Bandwidth Optimization (http://2000.jukuin.keio.ac.jp/shimazaki/res/kernel.html) A free online tool that instantly generates an optimized kernel density estimate of your data.
- Free Online Software (Calculator) (http://www.wessa.net/rwasp_density.wasp) computes the Kernel Density Estimation for any data series according to the following Kernels: Gaussian, Epanechnikov, Rectangular, Triangular, Biweight, Cosine, and Optcosine.
- Kernel Density Estimation Applet (http://pcarvalho.com/things/kerneldensityestimation/index.html) An online interactive example of kernel density estimation. Requires .NET 3.0 or later.

Categories: Estimation of densities | Nonparametric statistics | Machine learning