Neural Network Interest Group

Título/*Title*:
Optimal Parzen Window Estimation
and PDF Modelling

Autor(es)/*Author(s)*:
J.P. Marques de Sá

Relatório Técnico/*Technical Report* No. 1 /2009

FEUP/INEB, Rua Dr. Roberto Frias, 4200-465, Porto, Portugal

Título/*Title*:

Optimal Parzen Window Estimation
and PDF Modelling

Autor(es)/*Author(s)*:

J.P. Marques de Sá

Relatório Técnico/*Technical Report* No. 1 /2009

Publicado por/*Published by*: NNIG. http://paginas.fe.up.pt/~nnig/

Optimal Parzen Window Estimation

J.P. Marques de Sá
INEB, June 2009

Contents

1 Determination of optimal *h* and IMSE for a given *n*

Based on the formulas of Tapia, RA, Thompson, JR (1978) *Nonparametric Probability Density Estimation*. The John Hopkins University Press.

We are considering:

$$\hat{f}_n(x) = \frac{1}{nh_n}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h_n}\right)$$

The formulas of optimal *IMSE* and *h* (optimal in the *IMSE* sense) are (p. 59)

$$IMSE \cong \frac{1}{nh_n}\int_{-\infty}^{\infty} K^2(y)dy + h_n^{2r} k_r^2 \int_{-\infty}^{\infty} \left|f^{(r)}(x)\right|^2 dx$$

$$h_n = n^{-1/(2r+1)}\alpha(K)\beta(f)$$

$$\text{with } \alpha(K) = \left[\frac{\int K^2(y)dy}{2rk_r^2}\right]^{1/(2r+1)}, \ \beta(f) = \left[\int \left|f^{(r)}(x)\right|^2 dx\right]^{-1/(2r+1)}$$

From now on we only consider the Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

This kernel is also a pdf and is such that its *characteristic exponent*, *r*, is 2, where *r* is the largest positive number such that the *characteristic coefficient*

$$k_r = \lim_{u \to 0}\left[\frac{1 - k(u)}{|u|^r}\right]$$

is nonzero and finite and $k(u)$ is the characteristic function of the kernel pdf:

$$k(u) = \int_{-\infty}^{\infty} e^{iux} K(x)dx,$$

a Fourier transform of *K*. In a loose sense *r* controls the kernel decay.
Now, for the Gaussian kernel we have:

$$k(u) = \int_{-\infty}^{\infty} e^{iux} K(x)dx = e^{-u^2/2} \quad \Rightarrow \quad k_2 = \lim_{u \to 0}\frac{1 - e^{-u^2/2}}{u^2} = \lim_{u \to 0}\frac{ue^{-u^2/2}}{2u} = \frac{1}{2}$$

Moreover, notice that for k_3 is infinite. Therefore, the Gaussian kernel (along with other ones) has $r = 2$, $k_2 = ½$.

We now proceed to simplify the above expressions, denoting: h, k, α, β instead of h_n, k_2, $\alpha(K)$, $\beta(f)$; $I_K = \int K^2$; $I_2 = \int \left| f^{(2)} \right|^2$. We have:

$$h = \left(\frac{I_K}{n} \right)^{0.2} I_2^{-0.2} = \left(\frac{I_K}{I_2} \right)^{0.2} n^{-0.2} \quad (\alpha = I_K^{0.2};\ \beta = I_2^{-0.2});$$

And since $nh^5 I_2 = I_K$, we have:

$$IMSE = \frac{I_K}{nh} + \frac{h^4 I_2}{4} = \frac{5 I_K}{4nh} = \left(\frac{5 I_K}{4h} \right) \frac{1}{n}$$

We now analyze a few examples.

1.1 The formulas for the Gaussian kernel

For Gaussian kernel we have:

$$I_K = \int K^2 = \frac{1}{2\pi} \int e^{-y^2} dy .$$

With $y = z/\sqrt{2}$, $\int e^{-y^2} dy = \frac{1}{\sqrt{2}} \int e^{-z^2/2} dz = \frac{\sqrt{2\pi}}{\sqrt{2}} = \sqrt{\pi}$.

Thus, $I_K = \int K^2 = \frac{\sqrt{\pi}}{2\pi} = \frac{1}{2\sqrt{\pi}} = 0.282095$

$$\alpha = (0.282095)^{0.2} = 0.7764$$

This is the value presented by Tapia, RA, Thompson, JR (1978) in page 60.
Moreover:

$$h = \left(\frac{I_K}{n} \right)^{0.2} I_2^{-0.2} = \left(\frac{0.282095}{I_2} \right)^{0.2} n^{-0.2}$$

$$IMSE = \frac{5 I_K}{4nh} = 0.454178\, I_2^{0.2} n^{-0.8}$$

Therefore in the following computations, where Gaussian kernel is assumed, we only have to compute I_2 and obtain:

$$h(n) = \left(\frac{0.282095}{I_2} \right)^{0.2} n^{-0.2} = k_h n^{-0.2}$$

$$IMSE(n) = 0.454178\, I_2^{0.2} n^{-0.8} = k_{IMSE} n^{-0.8}$$

1.2 The n data points are from N(0,1)

We have:

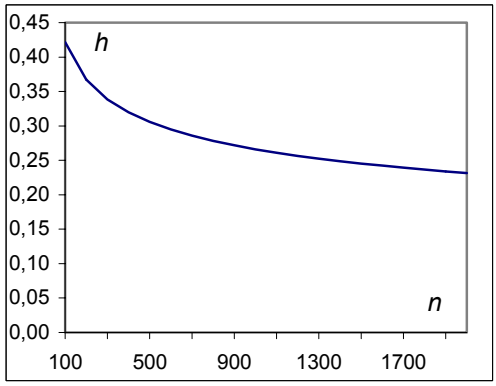
$I_2 = 0.2116$ (computed with Matlab)

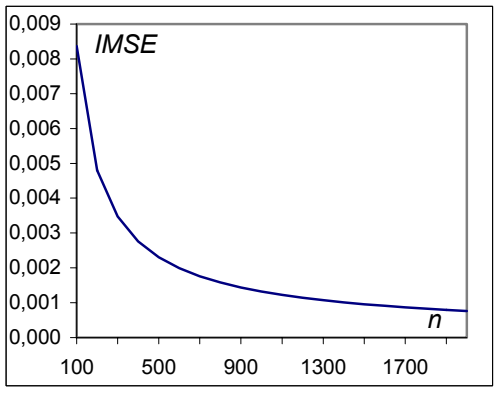
Let us now consider $n = 25$. We obtain:

$h = 0.5564$; $IMSE = 0.0254$

Tapia, RA, Thompson, JR (1978) indicates $h = 0.56$ (p. 67) and an average $IMSE$ in 24 experimental repetitions of 0.0163 (sd =0.119)

n	h	IMSE
25	0.556	0.025350
100	0.422	0.008362
200	0.367	0.004803
300	0.338	0.003472
400	0.320	0.002759
500	0.306	0.002308
600	0.295	0.001994
700	0.286	0.001763
800	0.278	0.001584
900	0.272	0.001442
1000	0.266	0.001325
1100	0.261	0.001228
1200	0.257	0.001145
1300	0.252	0.001074
1400	0.249	0.001013
1500	0.245	0.000958
1600	0.242	0.000910
1700	0.239	0.000867
1800	0.237	0.000828
1900	0.234	0.000793
2000	0.232	0.000761
4000	0.202	0.000437
6000	0.186	0.000316
8000	0.176	0.000251
10000	0.168	0.000210
12000	0.162	0.000182
14000	0.157	0.000160





16000	0.153	0.000144
18000	0.149	0.000131
20000	0.146	0.000121
22000	0.143	0.000112
24000	0.141	0.000104

1.3 The n data points are from N(0,σ)

Evaluation of the above formulas for a large range of σ values shows that *h* can be written as:

$$h = 1.0592\ \sigma\, n^{-0.2}$$

Therefore:

$$IMSE = \frac{0.332911}{\sigma} n^{-0.8}$$

1.4 The n data points are from 0.5N(-1.5,1)+0.5N(1.5,1)

We have:

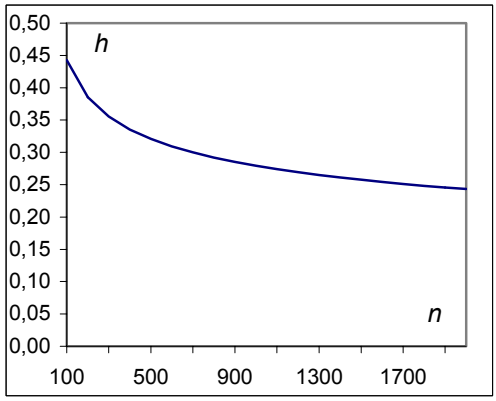
$$I_K = 0.1559;\ I_2 = 0.0918$$
$$\alpha = 0.6896;\ \beta = 1.6122$$

Thus, for $n = 25$:

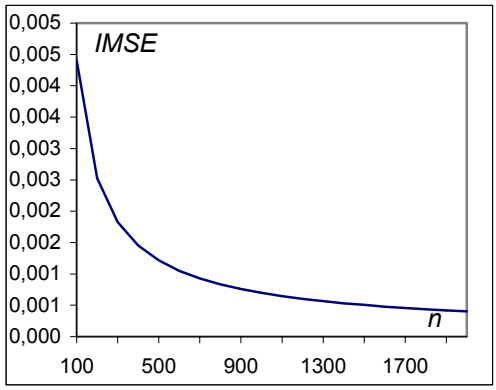
$$h = 0.584;\ IMSE = 0.0133$$

Tapia, RA, Thompson, JR (1978) indicates $h = 0.66$ (p. 67) and an average *IMSE* in 24 experimental repetitions of 0.0095 (sd =0.007)

n	h	IMSE
25	0.584	0.013347
100	0.443	0.004403
200	0.385	0.002529
300	0.355	0.001828
400	0.335	0.001452
500	0.321	0.001215
600	0.309	0.001050
700	0.300	0.000928
800	0.292	0.000834
900	0.285	0.000759
1000	0.279	0.000698
1100	0.274	0.000647
1200	0.269	0.000603
1300	0.265	0.000566
1400	0.261	0.000533
1500	0.258	0.000504
1600	0.254	0.000479
1700	0.251	0.000456



1800	0.248	0.000436
1900	0.246	0.000418
2000	0.243	0.000401
4000	0.212	0.000230
6000	0.195	0.000166
8000	0.184	0.000132



1.5 Weibull distribution

$$w(x) = \frac{b}{a}\left(\frac{x}{a}\right)^{b-1} e^{-(x/a)^b} I_{[0,\infty[}(x)\,; \qquad a > 0, \text{ scale; } b > 0, \text{ shape.}$$

This is the formula used by MATLAB.
Excel (and the M. Sá book) swap a by b.
Wikipedia uses (λ,k) with λ=a and k=b.

$$\mu = a\Gamma(1+1/b)$$
$$\sigma^2 = a^2\Gamma(1+2/b) - \mu^2$$
$$\gamma = \left[\Gamma(1+3/b)a^3 - 3\mu\sigma - \mu^3\right]/\sigma^3$$

γ is the skewness.

Properties:

1 $w_{1/\gamma,1}(x) \equiv \varepsilon_\lambda(x)$; b=1 sets the exponential shape and a, the scale, sets the decay.

2 For $b < 1$ the distribution shape is hyperbolic. It can easily be modeled by an exponential with low L1 distance.

3 For large b the Weibull distribution shape becomes increasingly symmetric (more or less compressed according to a) with $\gamma \approx 0.025$ for b=3.5.

Based on these properties the intervals for a and b that seem most interesting are:
$a \in [0.1, 3]$
$b \in [1, 3.5]$
MATLAB is able to compute the I_2 values for these intervals with increment 0.2 on b. For other values cubic spline interpolation may be used.

1.6 Gamma distribution

$$\gamma(x) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-x/b} I_{]0,\infty[}(x); \qquad a > 0, \text{ shape; } b > 0, \text{ scale.}$$

This is the formula used by MATLAB and EXCEL (notation α, β).
The M. Sá book uses (p, a) with $a = b$ and $p = a$.
The Wikipedia uses (k,θ) with $\theta = b$ and $k = a$.

Note that now a is the shape and b the scale (as opposite to Weibull).
Also note that the gamma distribution is not defined for $x = 0$ (by contrast the Weibull is defined at zero).

$$\mu = ab$$
$$\sigma^2 = \mu b$$
$$\gamma = 2/\sqrt{a}$$

Properties:

1 $\gamma_{1,1/\lambda}(x) \equiv \varepsilon_\lambda(x)$; a=1 sets the exponential shape and b, the scale, sets the decay.

2 For $a < 1$ the distribution shape is hyperbolic. It can easily be modeled by an exponential with low L1 distance.

3 Contrary to what happens with the Weibull distribution the gamma distribution maintains its skewed shape even for large a. For $a = 4$, $\gamma = 1$. For larger a the function exhibits close to the origin an increasingly large interval of nearly zero values; this feature (not so noticeable with Weibull) is uninteresting for pdf modeling.

Based on these properties the intervals for a and b that seem most interesting are:
$a \in [1, 4]$
$b \in [0.1, 3]$
MATLAB is able to compute the I_2 values for these intervals with increment 0.25 on a. For other values cubic spline interpolation may be used.

1.7 MATLAB code for Weibull and Gamma

```
function [h imse] = weibullh(b)
syms x f
Ik = 0.282095;
h = []; imse = [];
av = (0.1:0.1:3)';
for k=1:size(av,1)
    a = av(k);
    g = (b/a)*((x/a)^(b-1))*exp(-(x/a)^b);
    i2s = int( diff(diff(g,x),x)^2, x, 0, inf );
    I2 = eval(i2s);
    hc = (Ik/I2)^0.2;
    h = [h; hc];
```

```
    imse = [imse; 5*Ik/(2*hc)];
end
function [h imse] = gammah(b)
syms x f
Ik = 0.282095;
h = []; imse = [];
av = (1:0.25:4)';
for k=1:size(av,1)
    a = av(k);
    g = exp(-x/b)*(x^(a-1))/(b^a)/gamma(a);
    i2s = int( diff(diff(g,x),x)^2, x, 0, inf );
    I2 = eval(i2s);
    hc = (Ik/I2)^0.2;
    h = [h; hc];
    imse = [imse; 5*Ik/(2*hc)];
end
```

2 Pdf modeling and random data generation software

The following functions are implemented in MATLAB.

2.1 pdf1model

The `pdf1model` function is called as

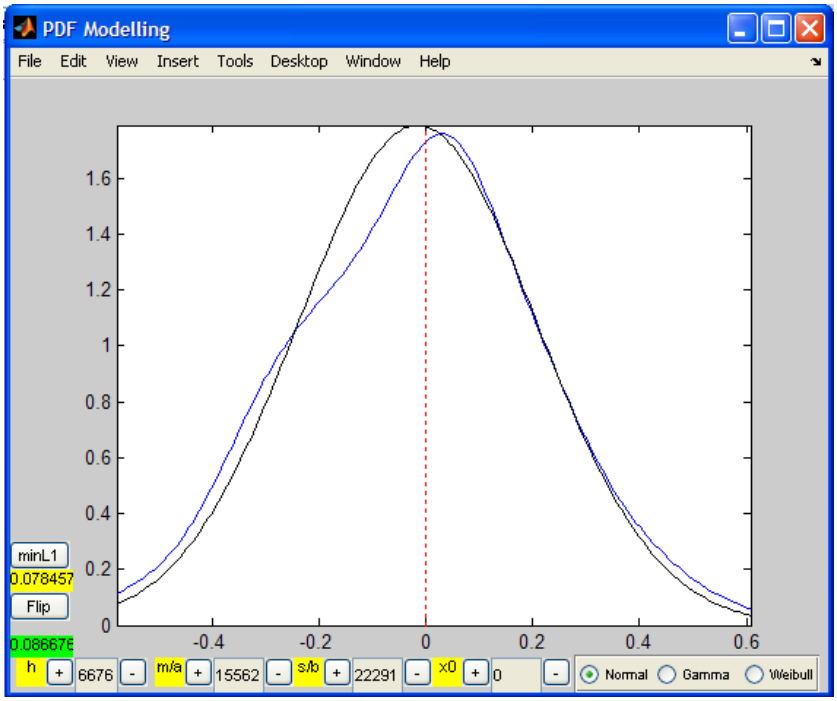
```
pdfstruct = pdf1model(x)
```

and finds a pdf model for the univariate data vector `x`. Pdf modeling is done as follows: first, a Parzen window estimate of the data pdf using a Gaussian kernel is derived; next, a best fit of a known pdf is searched for.
Information on the pdf model in returned through `pdfstruct`, a structure with the following fields:

- `type`, the distribution type: `'Normal'`, `'Gamma'`, `'Weibull'`.
- `params`, a row vector with 6 elements containing the following distribution parameters:
  Elements 1 and 2 are respectively the mean and standard deviation for the `'Normal'`, model or the *a* and *b* parameters for the `'Gamma'` and `'Weibull'` models. These two parameters are denoted `'m/a'`, `'s/b'` in the function GUI (see below).
  Element 3 is a user specified translation, `x0`, assuring the proper position of the `'Gamma'` and `'Weibull'` pdf origin. Note that `x0` is the quantity that must be *added* to the abscissa so that the pdf starts at zero.
  Element 4, `s`, indicates whether or not the data pdf had to be flipped in order be modeled by the `'Gamma'` or `'Weibull'` pdf (1 means 'no flip'; -1 means 'flip'). Note that when a data flip is performed the minimum and maximum abscissa, `xmin` and `xmax`, swap roles.
  Elements 5 and 6 are respectively `xmin` and `xmax`. Their values have to be known in order to retrieve any data generated with the pdf model in the proper position.

- `Stats`, a row vector with the following 4 elements:
  Elements 1 and 2 are the areas subtended by respectively the data pdf and the model pdf.
  Element 3 is the estimate of the L1 distance between the data pdf and the model pdf.
  Element 4 is the estimate of the integrated mean square error (IMSE) between the data pdf and the model pdf.

The function GUI shown below allows interactively designing the pdf model.



`pdf1model` GUI (see text).

GUI elements:

- Overlayed plot of the Parzen estimated pdf (blue line) and the specified pdf model (black line) with a sliding red line for `x0`.
- Buttons and editable text for specifying the smoothing factor, `h`. Optimal value assuming a normal pdf is shown in green text box (`0.086676`).
- Buttons and editable text boxes for specifying `m/a`, `s/b` and `x0`.
- Radio buttons for choosing the pdf model type: `'Normal'`, `'Gamma'`, `'Weibull'`. `'Normal'` model is the starting option.
- Button for flipping the estimated pdf (needed for `'Gamma'` and `'Weibull'`).
- Button for finding the best (`m/a`, `s/b`) values in the min(L1) sense. Current L1 distance is displayed in yellow box.

One should leave the pdf1model call by typing any key while in the MATLAB worksheet.

Technical notes:

- The "optimal" smoothing factor (kernel bandwidth) shown in the green text box is derived assuming normally distributed data and with the variance estimated from the data. This is usually an advisable value, which is also shown as the starting value in the `h` editable text

box (press the "Home" key to see this value; unfortunately MATLAB centers the text in this control causing it to be often masked).

- When the pdf model parameters are somewhat "hand"-adjusted it is advisable to press the `minL1` button. A best L1-distance model is then derived.

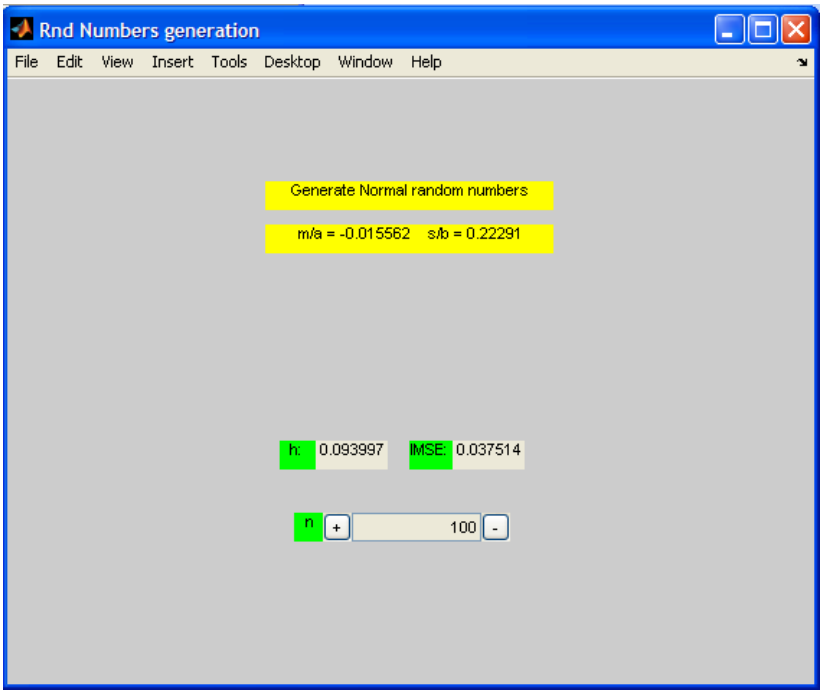
2.2 pdf1rnd

The `pdf1rnd` function is called as

```
z = pdf1rnd(pdfstruct,n)
```
or
```
z = pdf1rnd(pdfstruct)
```

and returns a column vector `z` containing data points generated according to the pdf model specified by the `pdfstruct` structure.
When `z = pdf1rnd(pdfstruct,n)` is used exactly `n` points of the distribution are generated.
When `z = pdf1rnd(pdfstruct)` is used the user has the possibility of choosing `n` with the guidance provided by the GUI shown below.
The GUI has a header describing which model is being used and its main parameters. It also has the means to specify the value of `n`, at the same time showing the values of IMSE and `h` corresponding to that `n`. Usually one specifies a value of `n` achieving a sufficiently low IMSE (when using the indicated `h`).

Technical note:
The Gamma and Weibul tables used for computing the optimal IMSE and `h` are for the parameter intervals mentioned in 1.5 and 1.6.



`pdf1rnd` GUI (see text).

2.3 data1clone

A combination of the above functions allowing the generation of a clone dataset of a given dataset in one step.

2.4 data2clone

A version of the preceding function for bivariate datasets. In this case the data undergoes first a principal component transformation which will assure the independence of the marginal pdfs in the case of bivariate normal distributions.
In the end the data undergoes the inverse transformation guaranteeing the estimated covariance matrix.
With this tool one can generate bivariate pdf models guaranteeing reasonable marginals (with minimum L1 distances from the original pdfs) and covariance close to the original one. Whenever the uncorrelated marginals provided by the principal component projections can be assumed as being independent, the clone data should closely resemble, in a statistical sense, to the original data.
Before leaving `data2clone` scatter plots of the original and clone datasets are shown for visual comparison purposes.