A methodology for estimating joint probability density functions

Mauricio Monsalve

Computer Science Department, Universidad de Chile

Abstract

We develop a theoretical methodology for estimating joint probability density functions from marginals by, transforming the set of random variables into another set of independent random variables is necessary. To test this theory in practice, we developed a software application which implemented a reduced form of this methodology, and worked well with several datasets. Finally, we discuss how to expand this methodology.

Key words: joint probability density funtions, function estimation

1. Introduction

Estimating the distibution of random variables is an essential concern to statistics and its related disciplines, such as machine learning and data mining.

A classic approach to estimating joint probabilities for discrete data are Chow-Liu trees. Chow-Liu trees method consist in specifying which variables are related using a tree structure, much like Bayesian trees [1]. The best property of Chow-Liu trees is that, if they are built as maximum likelihood estimators, then they can be built using a MST algorithm, like Kruskal's.

Later, machine learning techniques appeared. Several modeling techniques deal with the problem of computing joint probabilities, such as Bayesian networks and classification trees, but these are discrete. However, neural networks and support vector machines already support fuzzy input, which makes them somewhat adequate for estimating joint pdfs. And regarding fuzziness, fuzzy logic and sets are somewhat adequate for estimating joint pdfs too. Despite fuzzy methods were designed to work with discrete data, their trick was supporting continuous data. But these methods still do not analytically model joint pdfs. (Yet can extend almost any discrete technique to continuity.)

In what strictly regards to estimating joint pdfs or cdfs, we find several multivariate distributions such as the multivariate normal distribution, multivariate Student distribution, Wishart distribution (generalizes chi-square to multiple dimensions), etc. Note that this approach to joint distributions is the classic one.

A slightly different approach is shown in [2]. In this work, Pages-Laguna and Zamoras apply spectral estimate methods to estimate joint pdfs. Spectral estimate methods are applied to power spectrum density functions, which are positive and real, like pdfs. Note that the approximation techniques also apply to joint cdfs.

However, there is an active research topic concerned with the estimation of cdfs: copulas. A *copula* is a function defined on $[0,1]^n$ (n dimensions) which estimates joint cdfs from marginals cdfs [3], and can naturally be extended to estimate joint jdfs.

Research on copulas started after Sklar's theorem in 1959, when he demonstrated that always exists a copula $C : [0,1]^d \to [0,1]$ which estimates any given joint cdf. Each argument of the copula is marginal cdf evaluated on its parameter, i.e.

$$C(F_1(x_1), ..., F_n(x_n)) = F(x_1, ..., x_n),$$

where F_i, $i = 1..n$, are marginal cdfs, and $F(x_1,...,x_n)$ is the joint cdf.

To make use of copulas, it is necessary to use a function which form is almost completely defined. We claim to have solved this problem in part, with a different approach, which is concerned with joint pdfs instead of joint cdfs.

In the following, we will deduce a method for estimating joint pdfs from sample data, by transforming a set of random variables into a set of independent ones, and by computing the marginal pdfs of the latter. Then we present a software application based on this methodology, and test it using several datasets. Finally, we discuss how to improve the methodology.

2. Our method

Let us recall two particular cases of pdfs. First, if $\vec{X}$ is a vector of independent random variables, then its joint pdf is:

$$f(x_1,x_2,...,x_n) = f_1(x_1)f_2(x_2)...f_n(x_n),$$

and if not, one can generally write the joint pdf as:

$$f(x_1,x_2,...,x_n) = f_1(x_1)f_2(x_2|x_1)...f_n(x_n|x_1,...,x_{n-1}),$$

where

$$f(a|b) \equiv \frac{\partial}{\partial a} \lim_{\varepsilon \to 0} \frac{\Pr(A \leq a \wedge B \in \mathcal{B}(b,\varepsilon))}{\Pr(B \in \mathcal{B}(b,\varepsilon))},$$

and $\mathcal{B}(b,\varepsilon)$ is the ball centered in b with radius ε.

What if we could reduce a joint pdf of the second form to a pdf of the first form? If we could do that, then the problem of computing the pdf is reduced to reducing the problem and computing the marginals.

2.1. Joint pdfs from marginals

Let us suppose we can define variables Y_k so that

$$Y_k = \phi_k(X_1,...,X_k), \forall k \leq n,$$

so that $\{Y_k\}$ is a set of independent random variables and $\{\phi_k\}$ are increasing in their last parameter. Then, the joint pdf of $\vec{Y} = (Y_1,...,Y_n)$ is

$$g(y_1,...,y_n) = g_1(y_1)...g_n(y_n).$$

In particular $g_k(y_k) = g_k(y_k|y_1...y_{k-1})$ because of their independence.

Now, let us focus on $f_k(x_k|x_1...x_{k-1})$. We first see that:

$$f_k(x_k|x_1...x_{k-1}) = \frac{\partial}{\partial x_k} \Pr(X_k \leq x_k|(\forall i < k)X_i = x_i).$$

As ϕ_k is increasing in its last parameter:

$$\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i) = \Pr(\phi(X_1,...,X_k) \leq \phi(X_1,...,X_{k-1},x_k)|(\forall i < k)X_i = x_i).$$

Replacing $\phi_k(X_1,...,X_k)$ by Y_k:

$$\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i) = \Pr(Y_k \leq \phi(X_1,...,X_{k-1},x_k)|(\forall i < k)X_i = x_i).$$

By using $(\forall i < k)X_i = x_i$, we can replace $\phi(X_1,...,X_{k-1},x_k)$ by $\phi(x_1,...,x_k)$:

$$\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i) = \Pr(Y_k \leq \phi(x_1,...,x_k)|(\forall i < k)X_i = x_i).$$

By recognizing that the $\{X_i\}$ and $\{Y_i\}$ sets share the same information, we can replace $(\forall i < k)X_i = x_i$ by $(\forall i < k)Y_i = y_i$:

$$\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i) = \Pr(Y_k \leq \phi(x_1,...,x_k)|(\forall i < k)Y_i = y_i).$$

And recalling that $\{Y_i\}$ is a set of independent random variables, we get:

$$\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i) = \Pr(Y_k \leq \phi(x_1,...,x_k)).$$

Thus, going back to f_k:

$$f_k(x_k|x_1...x_{k-1}) = \frac{\partial}{\partial x_k}\Pr(X_k \leq x_k|(\forall i < k)X_i = x_i)$$

$$f_k(x_k|x_1...x_{k-1}) = \frac{\partial}{\partial x_k}\Pr(Y_k \leq \phi_k(x_1,...,x_k))$$

$$f_k(x_k|x_1...x_{k-1}) = g_k(\phi_k(x_1,...,x_k))\frac{\partial \phi_k}{\partial x_k}.$$

Therefore, the joint pdf of $X_1,...,X_k$ is:

$$f(x_1,...,x_n) = \Pi_k\left(g_k(\phi_k(x_1,...,x_k))\frac{\partial \phi_k}{\partial x_k}\right).$$

Note that computing f was reduced to computing each g_k and ϕ_k. In particular, computing g_k corresponds to the classic problem of estimating univariate pdfs from samples. So, the difficulty of estimating f_k was really reduced to computing ϕ_k.

Note that we cannot use this approach to compute joint cdfs. If it holds that

$$F(x_1,...,x_n) = G(\phi_1(x_1),...,\phi_n(x_1,...,x_n)),$$

then it also holds that

$$\int_{\Omega_F} g(Y_1,...,Y_n)d\vec{Y} = \int_{\Omega_G} g(Y_1,...,Y_n)d\vec{Y},$$

where

$$\Omega_F = \{\phi_1(X_1), \phi_2(X_1,X_2),...,\phi_n(X_1,...,X_n) : X_1 \leq x_1 \wedge ... \wedge X_n \leq x_n\}$$

and

$$\Omega_G = \{Y_1, Y_2,..., Y_n : Y_1 \leq \phi_1(x_1) \wedge Y_2 \leq \phi_2(x_1,x_2) \wedge ... \wedge Y_n \leq \phi_n(x_1,...,x_n)\}.$$

The above condition requres that $\Omega_F = \Omega_G$, a condition we cannot guarantee. For example, consider that $X_1 = Y_1$ and $X_2 = Y_2 - Y_1$; $X_1 \leq 0$ and $X_2 \leq 0$ is equivalent to $Y_1 \leq 0$ and $Y_2 \leq Y_1$ (Ω_F), which is different to $Y_1 \leq 0$ and $Y_2 \leq 0$ (Ω_G).

To address the problem of computing joint cdfs, we suggest using Monte Carlo methods, which are not affected by the *curse of dimensionality* as their convergence rate is independent of the number of dimensions (but instead are very slow) and are simple to implement.

2.2. Choosing ϕ_k

Which ϕ_k functions should we use? We propose reducing the problem of computing $\phi_k(x_1,...,x_k)$ to the problem of computing $\psi_k(y_1,...,y_{k-1},x_k)$, a function which creates a new random variable by making x_k independent from $y_{i<k}$. Note that ϕ_k can be reduced to ψ_k,

$$\phi_k(x_1,...,x_k) = \psi_k(\psi_1(x_1),\psi_2(\psi_1(x_1),x_2),\psi_3(\psi_1(x_1),\psi_2(\psi_1(x_1),x_2),x_3),...,x_k),$$

and it holds that

$$\frac{\partial \phi_k}{\partial x_k} = \frac{\partial \psi_k}{\partial x_k}.$$

Dealing with such complicated form of ψ_k is not hard. We just have to keep a record of our transformations, to use the already computed variables in the next steps:

$$y_1 = \psi_1(x_1),$$

$$y_2 = \psi_2(y_1,x_2),$$

$$y_3 = \psi_3(y_1,y_2,x_3),$$

$$y_4 = \psi_4(y_1,y_2,y_3,x_4),$$

$$...$$

$$y_n = \psi_n(y_1,...,y_{n-1},x_n).$$

Now that the problem is reduced to using ψ_k, the next problem consists in estimating these functions. (We leave this problem open.)

3. Implementation

We implemented the simplest case of this methodology, which is limiting ourselves to the linear case. In particular, we will assume that independence is achieved when covariance is zero. Naturally, this assumption will not work is many cases, so we also decided to evaluate the accuracy of our estimation in our software.

Note that, to build the independent random variables, we use functions

$$y_k = \psi_k(y_1,...,y_{k-1},x_k) = x_k - \sum_{i<k} \alpha_i y_i,$$

so y_k can be obtained using a Gram-Schmidt method. Also, $\partial \psi_k / \partial x_k = 1$, which simplifies the implementation a bit.

The application was written in Java and reads CSV files. Its appearance can be seen in figure 1.

3.1. Supported pdfs

We supported the following pdfs in our software to compute the marginals:

Normal distribution. We had to include this distribution because it appears commonly.

Powered normal distribution. It is also common to find skewed bell shaped distributions, like lognormals. To address this problem in its generality, we include the *Box-Cox method*, which enables our program to handle these cases.

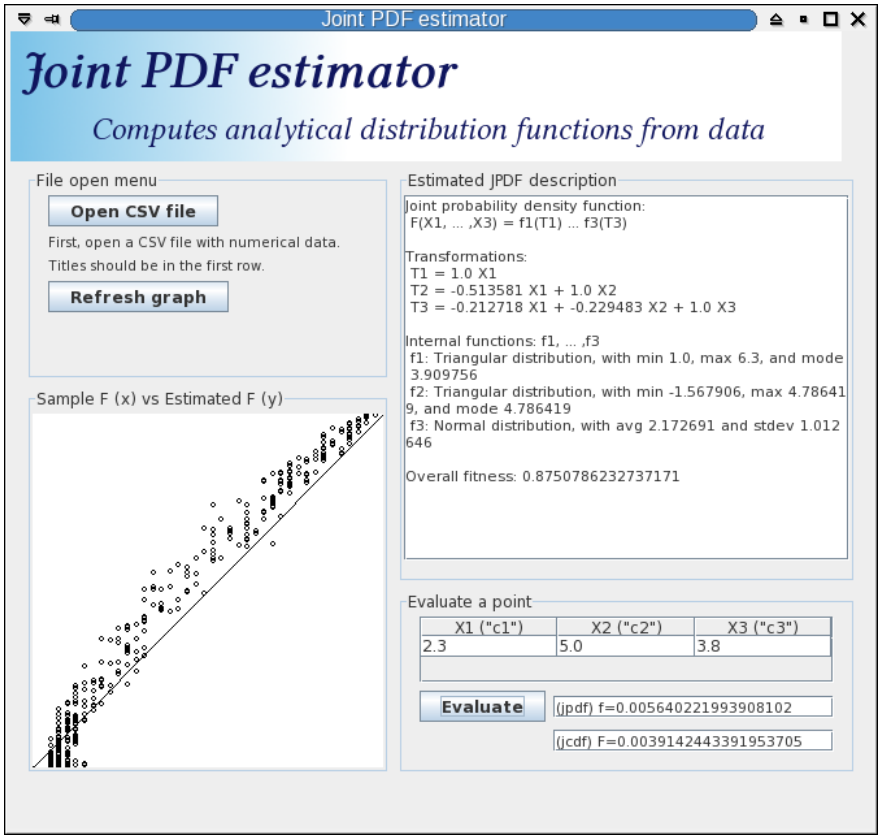


Figure 1: Joint PDF Estimator. Evaluated with cc42a dataset.

Power law. Power laws are also common distributions. They appear often when ”inequality” appears.

Exponential distribution. This distribution is common is memoryless processes, like waiting times.

Triangular distribution. We included this distribution to handle spiked distributions.

Uniform distribution. Uniform distributions are also common. And they are very simple to implement.

3.2. Goodness of fit

To assess the quality of the estimation, we decided to avoid using Pearson or similar goodness of fit tests. Basically, our data has several dimensions and we cannot contrast a joint pdf to the data; there are not joint pdfs from samples since discrete values are not differentiable. Besides, histogram-based tests are computationally expensive with high dimensions, so we decided to compare the estimated joint cdfs to the sample joint cdfs using observations. If the estimation was good, then both cdfs should be similar, i.e. $F_e(x_1,...,x_n) = F_s(x_1,...,x_n) + \varepsilon$ (where ε is a small error).

To measure the similarity of F_e (our joint cdf) and F_s, we generate a large sample. For each observation, we compute both F_e and F_s (we need the original data to compute this function) and then we perform a linear regression between them. In particular, we expect:

- A high correlation. That is, $r^2 \approx 1$. $\left(r^2 = \frac{Cov^2(F_e,F_s)}{Var(F_e)Var(F_s)}.\right)$

- A null intercept. That is, in $F_e = aF_s + b$, $b \approx 0$.
- An inclination of one. That is, in $F_e = aF_s + b$, $a \approx 1$.

The above expectations led us design a score for assessing the goodness of fit of our data:

$$q = r^2 \left(\frac{1}{1+|b|} \right) \left(\frac{1}{1+|a-1|} \right)$$

Note that we may have used a Cramer von Mises test to assess goodness of fit, but F_e and F_s do not distribute uniformly. In particular, relations among variable change the way their joint cdfs distribute. So, we decided to use a least squares approach which handles ill distributed data: weighted least squares. We splitted the observations in bins, and the weights per bin are the multiplicative inverse of the number of observations.

3.3. Computing joint cdfs

Monte Carlo methods are slow since their convergence rate is sublinear. In particular, we will use average weighting which converges in order $O(n^{0.5})$, and is computed as:

$$\int_S f(x_1,...,x_n) \approx \frac{\sum_k f(x_1,...,x_n)\mathbb{I}_S(x_1,...,x_n)}{\sum_k f(x_1,...,x_n),}$$

where $\mathbb{I}_S(x_1,...,x_n) = 1$ when $(x_1,...,x_n) \in S$, and is zero otherwise [4]. Note that the sample size must be large to achieve precision.

However, we can compute a large number of joint cdfs efficiently. To do this, we propose creating a large sample and using it to compute several cdfs. Nevertheless, we may suffer the risk of consistent bias. To avoid this, we may renew the sample after several uses.

In our particular case, we build samples of 2000 observations and renew them after 50 computations, when evaluating large numbers of joint cdfs. By doing this, we saved much computational time; what used to take 10 minutes, now takes about 5 to 15 seconds to compute. Note that we even saved the time of evaluating pdfs, since we can do that at the same time we renew the sample. (When we generate each $(x_1,...,x_n)$, we also compute its joint pdf and record it.)

4. Experiments

We tested our software using several datasets. First, we tested it using four datasets which were random sampled by us. If X,Y,Z are independent random variables, then the columns of Ss1 are $A+0.5B, C-A, B$; these of Ss2 are $A^{1.1}+B^{1.1}, B^{0.9}-10C^{1.1}, A+C+AC/1000$; these of Ss3 are $A+AB/1000, B-\log C, 20\log(C+5)$; and these of Ss4 are $(A+B)^2, (C-B)^2, C^2$. Ss2, Ss3 and Ss4 where designed to evaluate the tolerance of our estimations when slight nonlinear relations appeared.

Second, we used course data. These datasets correspond to the grades of students of Universidad de Chile in three different courses. We expect the presence of linear relations in these datasets.

Third, we used the voting intentions in the Chilean 1988 Plebiscite dataset [5]. It has 2700 observations and involves 5 variables which relations are somewhat obscure. (Linear relations are found but do not fully explain the data.)

And fourth, we used several datasets from the Andrews & Herzberg archive at StatLib [6]. To use them, we removed their first columns and left only numerical sampled data. (We removed text data, row numbers, etc.) The sizes of these datasets range from 10 to 302.

Table 1: Evaluation of Joint PDF Estimator with several datasets.

Dataset	Size	Variables	Fitness	Comments
Ss1	157	3	0.93	Fine, yet slightly biased around $F_s = 0$.
Ss2	157	3	0.88	Works fine, somewhat dispersed.
Ss3	157	3	0.88	Somewhat disperse and biased around $F_s = 0.75$. (Fig.2-a.)
Ss4	157	3	0.78	Small bias. Dispersion increases as F_s decreases.
cc10b	63	4	0.80	Works fine. Last variable ignored.
cc42a	41	3	0.87	Low dispersion, but slightly biased. (Fig.1.)
in50a	84	6	0.80	High dispersion. Seems unbiased.(Fig.2-b.)
voting	2700	5	0.65	High dispersion. Slight bias around $F_s = 0$. (Fig.2-c.)
T01.1	50	12	0.45	High dispersion.
T03.1	38	6	0.73	Small bias.
T07.1	184	4	0.30	Works bad.
T13.1	105	12	0.33	Works bad.
T17.1	127	12	0.28	Works bad.
T25.1	264	4	0.44	Works bad. Strong bias.
T30.1	19	6	0.81	Works well despite sample size. (Fig.2-d.)
T33.1	100	5	0.69	Disperse but overall fine.
T35.2	52	6	0.79	Work fine. Slight bias around $F_s = 1$.
T41.1	10	14	0.38	Works bad. Only 10 variables considered.
T53.1	302	9	0.12	Works bad.
T59.1	42	8	0.56	High dispersion. Seems unbiased.
T60.1	104	4	0.84	Works well. A bit disperse.

Table 1 shows the results of the tests. Note that "Fitness" corresponds to the g measure we already defined, and its value is randomly generated. Also note that F_e is randomly generated too (using Monte Carlo integration) and that F_s is estimated from the dataset by counting (another source of error). We cannot expect g to be close to 1 (for example, 0.97) under these conditions.

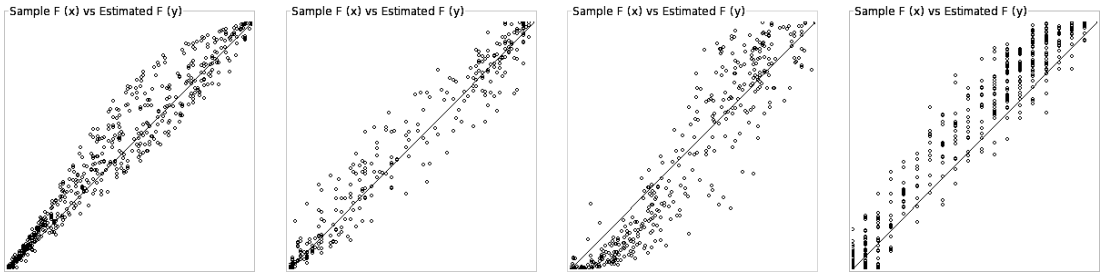
As we can see, our application worked well with several datasets. The results of datasets Ss2, Ss3 (fig.2-b) and Ss4 show that it is somewhat tolerant to nonlinear relations. The result of dataset T30.1 (fig.2-d) shows that it can approximate joint pdfs even when samples are small. (By taking relations into account, the *curse of dimensionality* should be reduced. For example, if $corr(X,Y) \approx 1$, we need as many observations as these needed to estimate X solely when estimating $f(x,y)$. Dimensions are relevant as the uncertainty increases.) Also, the results of the estimations give validity to the measure g included in the program.

5. Conclusions

We developed a methodology for estimating joint pdfs, which can be summarized to:

1. Compute functions $\phi_1 ... \phi_n$ such that variables $Y_k = \phi_k(X_1, ..., X_k)$ form a set of independent random variables.
2. Compute the marginal pdf g_k for each Y_k.
3. The joint pdf of $X_1, ..., X_n$ is $f(x_1, ...x_n) = g_1(y_1)...g_n(y_n)\frac{\partial \phi_1}{\partial x_1}...\frac{\partial \phi_n}{\partial x_n}$, where $(\forall k)\, y_k = \phi_k(x_1, ..., x_k)$.

To put this methodology into practice, we developed a software application which works with linear ϕ_k functions. We found that its estimations worked well (the estimated joint cdfs were similar to the sample joint cdfs) in several datasets, which confirms our theory.



(a) Ss3 (b) in50a (c) voting (d) T30.1

Figure 2: Scatter plots generated by Joint PDF Estimator.

In spite of the simplicity of our methodology, finding suitable functions ϕ_k is not a simple problem. By improving these functions, we may be able to estimate better joint pdfs. However, we should design simple functions as complicated pdfs are hard to use in theory. Naturally, this constraint also applies to $\partial\phi_k/\partial x_k$, as it is part of the built pdf.

Acknowledgments.. We would like to thank project FONDECYT 1070348 for their support.

References

[1] C.K. Chow, C.N. Liu. Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory, 14(3): 462-467. 1968.

[2] A. Pages-Zamora, M.A. Lagunas. Joint probability density function estimation by spectral estimate methods. In *Proceedings of the Acoustics, Speech, and Signal Processing, 1996. on Conference Proceedings., 1996 IEEE international Conference - Volume 05* (May 07-10, 1996). ICASSP. IEEE Computer Society, Washington, DC, 2936-2939. 1996.

[3] T. Schmidt. Coping with copulas. In *Copulas: From theory to application in finance.* 2007.

[4] C.P. Robert, G. Casella. Monte Carlo Statistical Methods (Springer Texts in Statistics). Springer-Verlag New York, Inc. 2005.

[5] Voting intentions of the Chilean 1988 Plebiscite dataset. In John Fox, *Applied Regression, Generalized Linear Models, and Related Methods*. 2nd edition, Sage Publications. 2008. `http://socserv.mcmaster.ca/jfox/Books/Applied-Regression-2E/datasets/Chile.txt`

[6] Andrews and Herzberg archive. StatLib. `http://lib.stat.cmu.edu/datasets/Andrews/`