GLOMERULAR FILTRATION RATE ESTIMATION BY A NOVEL BINNING-LESS BIVARIATE ISOTONIC STATISTICAL REGRESSION METHOD

SEBASTIAN GILES, SIMONE FIORI*

Abstract. Bivariate statistical regression is a method for finding the relationship between unpaired sets of data based on statistic distribution matching. In the present report an algorithm is proposed to eliminate binning from the previously published procedure. The method is then applied to correlate Glomerular filtration rate (GFR) to serum creatinine concentration. GFR is an important indicator of kidney function, as direct measurement is highly impractical there is considerable interest in developing formulas to estimate it from parameters which are easier to obtain.

1. Introduction.

Several real-world phenomena lack accurate mathematical descriptions. In these cases it is not possible to predict the value of a variable by plugging known quantities in to an analytically derived expression, therefore statistical methods must be employed to develop a model of the observed process. The most common set of tools used to infer a functional relationship between variables is regression analysis, here only bivariate regression techniques will be considered. The amount of available data is assumed to be enough to explain the relevant statistical features of the phenomenon underlying the data.

Traditional forms of regression are based on some parametrised function whose graph is made to lie reasonably close to the points making up the experimental dataset. Values for the parameters are typically found using the least squares method.

Isotonic regression allows greater freedom for the regression curve to fit data by constructing a piecewise linear function, described by a lookup table (LUT). Overfitting is avoided by requiring the function to be monotonic, this is obviously also a limit on the process we want to model. Various algorithms can be used to find the LUT values that satisfy a least squares condition.

Statistical bivariate regression (SBR) constitutes an improvement over isotonic regression, its advantages derive from the fact that it relies on finding the relationship between the statistical distributions of the variables. SBR is not based on a least squares method so it does not require data to be associated in ordered pairs, this makes it ideal for correlating two quantities that cannot be both measured on the same individual. The algorithm presented in [1] independently estimates the probability density functions (PDF) of the two variables by dividing the dataset ranges into bins and populating LUTs for the relative frequencies, the PDFs are then integrated to obtain the cumulative distribution functions (CDF) which are, or can be licitly adjusted to be, bijective and allow for the regression model to be obtained as the map between values with equal probabilities.

This report describes an alternative algorithm developed to make bivariate statistical regression more versatile and faster over large datasets by entirely avoiding the binning and integration operations.

Glomerular filtration rate (GFR) is used as an indicator of kidney function, as such it is relevant for assessing progression of renal disease, it is also frequently required for evaluating optimal dosage for medications. However, determination of true GFR is time-consuming, costly, and difficult to perform. Thus, there is considerable interest in developing formulas to estimate GFR using simpler parameters such as age, weight, height and sex and values which can be more conveniently measured as part of a blood test.

The present report is organized as follows. Section 2 recalls the notion of bivariate statistical regression and explains the main idea and the details about the proposed binning-less bivariate statistical regression algorithm. Section 3 explains an analysis of the glomerular-filtration-rate (GFR) estimation

*S. Giles is with the School of Information and Automation Engineering, Università Politecnica delle Marche, Via Brecce Bianche, I-60131 Ancona (Italy).

S. Fiori is with Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Via Brecce Bianche, I-60131 Ancona (Italy).

This draft is dated October 11, 2017.

based on creatinine levels and illustrates such analysis by means of numerical tests performed on a dataset drawn from a study on pediatric patients in mainland China. Section 4 concludes the paper.

2. Binning-less bivariate statistical regression algorithm. Given the random variables X and Y, for which we expect the existance of a monotonic function f such that $Y = f(X)$, let $D_X \in \mathbb{R}^n$ and $D_Y \in \mathbb{R}^m$ be arrays whose components are realizations of X and Y respectively.

The developed regression algorithm is encapsulated in a function that takes the D_X and D_Y dataset arrays along with an x for which the estimated value of $f(x)$ is returned.

Denoting by $P_X(x)$ and by $P_Y(y)$ the respective CDFs of X and Y, we recall from our previous [1][3] work that

$$f(x) = P_Y^{-1}(P_X(x)) \text{ if } f \text{ is monotonically increasing,} \tag{2.1}$$

$$f(x) = P_Y^{-1}(1 - P_X(x)) \text{ if } f \text{ is monotonically decreasing} \tag{2.2}$$

We further recall that, since the modeling solution is not unique, in general, we assumed that the center of mass of the X data corresponds to the center of mass of the Y data.

The regression procedure can be separated into two parts: the evaluation of a CDF and the evaluation of an inverse CDF. In this report the former is handled by the *cdf* function defined in Algorithm 2, the latter by *invcdf* defined in Algorithm 3. Besides the argument for the CDF (or inverse CDF) both procedures require a dataset from which to infer the actual distributions. Algorithm 1 provides pseudocode for putting the two parts together in the case that a monotonically increasing model is sought. If the model is expected to be monotonically decreasing then *"P"* on Line 3 should be replaced by it's complement *"1-P"*.

Algorithm 1 Statistical Bivariate Regression

```
1: function STATISTICALREGRESSION($D_X$,$D_Y$, $x_q$)
2:     $P \leftarrow cdf(D_X, x_q)$          ▷ Evaluate CDF for value $x_q$ in data set $D_X$
3:     $y_q \leftarrow invcdf(D_y, P)$       ▷ Evaluate inverse CDF for probability $P$ on data set $D_Y$
4:     return $y_q$
5: end function
```

3. Cumulative distribution function estimation algorithm. This section explains a procedure to estimate the value of the CDF $P(q)$ of a generic random variable for which n realizations are stored as the components of the array D. The main idea to avoid binning is to estimate the cumulative distribution function of the dataset without resorting to an estimate of the probability density function first, this can be done by embracing the definition of CDF itself, which simply leads us to counting the number of realizations that are less than or equal to q and dividing by n. The solution shown in algorithm 2 expands this idea to allow for a continuos, strictly monotonic interpolation for values which are not included in the original dataset. Please mind that array indexing is 1-based.

Here are a few comments about Algorithm 2:

- **Line 2**: The algorithm is notably simplified by sorting D into ascending order.
- **Lines 4–13**: This part is essentially a binary search for q in array D. The loop starts with indexes l and r as the extremes of D and ends with l as the index of the last value less than q, and r as that of the first value greater than q. The only exception that may occur is handled in lines 14–16.
- **Line 8**: Making sure that $D[m] \neq D[l]$ is needed to stop l and r converging to 1 and 2 respectively, in the case that q is smaller than all elements in D.
- **Lines 14–16**: This loop is needed to fix l in the case it converges to $n - 1$ as a consequence of q being greater than all values in D.

Algorithm 2 Cumulative distribution function estimation

```
 1: function CDF($D$, $q$)
 2:     $D \leftarrow sort(D)$                    ▷ Dataset is put into ascending order
 3:     $n \leftarrow length(D)$
 4:     $l \leftarrow 1$
 5:     $r \leftarrow n$
 6:     while $r - l > 1$ do
 7:         $m \leftarrow \lfloor (l + r)/2 \rfloor$
 8:         if $D[m] > q \wedge D[m] \neq D[l]$ then
 9:             $r \leftarrow m$
10:         else
11:             $l \leftarrow m$
12:         end if
13:     end while
14:     while $D[r] = D[l]$ do
15:         $l \leftarrow l - 1$
16:     end while
17:     while $r < n \wedge D[r] = D[r + 1]$ do
18:         $r \leftarrow r + 1$
19:     end while
20:     $d \leftarrow (q - D[l])/(D[r] - D[l])$
21:     $p \leftarrow (l + d \cdot (r - l))/n$
22:     if $p < 0$ then
23:         $p \leftarrow 0$
24:     else if $p > 1$ then
25:         $p \leftarrow 1$
26:     end if
27:     return $p$
28: end function
```

- **Lines 17–19**: The previous operations already guarantee that l is the last index for the value $D[l]$, this means there are l elements in D which are less than or equal to $D[l]$. This loop finds r, the count of elements that are less than or equal to $D[r]$.
- **Lines 20–21**: $P(D[l])$ can be estimated by l/n whilst $P(D[r])$ can be estimated by r/n, $P(q)$ is obtained via linear interpolation.
- **Lines 22–26**: These checks limit the CDF for values outside the range of D.

Evaluation of the inverse CDF is based on the same principle only applied in reverse: the input argument is a probability, it is multiplied by n and rounded to an integer r. If the dataset D is sorted, then there will be r values less than or equal to $D[r]$. Algorithm 3 allows for a continuos, strictly monotonic interpolation to yield values which are not included in the original dataset. Here are a few comments about Algorithm 3:

- **Line 2**: The algorithm is notably simplified by sorting D into ascending order.
- **Line 4**: The input probability is denormalised into the range $[0, n]$.
- **Lines 5–9**: r and l take the value of p rounded to the next integer to be used as an index, this also requires r and l to be non-zero.
- **Lines 10–12**: r is made to point to the last occurence of the smallest value whose CDF is greater than the requested probability.
- **Lines 13–15, 20**: l is made to point to the last occurence of the greatest value whose CDF is

Algorithm 3 Inverse Cumulative distribution function estimation

```
 1: function INVCDF($D$,$P_q$)
 2:     $D \leftarrow sort(D)$                          ▷ Dataset is put into ascending order
 3:     $n \leftarrow length(D)$
 4:     $p \leftarrow P_q \cdot n$
 5:     $r \leftarrow \lceil p \rceil$
 6:     if $r = 0$ then
 7:         $r \leftarrow 1$
 8:     end if
 9:     $l \leftarrow r$
10:     while $r < n \wedge D[r] = D[r+1]$ do
11:         $r \leftarrow r + 1$
12:     end while
13:     while $l > 1 \wedge D[l-1] = D[l]$ do
14:         $l \leftarrow l - 1$
15:     end while
16:     if $l = 1$ then
17:         $l \leftarrow r$
18:         $r \leftarrow r + 1$
19:         while $r < n \wedge D[r] = D[r+1]$ do
20:             $r \leftarrow r + 1$
21:         end while
22:     else
23:         $l \leftarrow l - 1$
24:     end if
25:     $d \leftarrow (p - l)/(r - l)$
26:     $q \leftarrow D[l] + d \cdot (D[r] - D[l])$
27:     return $q$
28: end function
```

less than the requested probability.

- **Lines 16–22**: Alignments are made to allow interpolation for small probabilities.
- **Line 20-21**: $P^-1(l/n)$ can be estimated by $D[l]$ whilst $P^{-1}(r/n)$ can be estimated by $D[r]$, nearby values are obtained via linear interpolation.

Small numerical example: Let $X = [2, 3, 5, 5, 6, 6, 7, 9]$, $Y = [6, 7, 10, 10, 11, 11, 11, 12, 15, 20]$, i.e., $n = 8$, $m = 10$. The actual underlying model is $f(x) = 2x + 2$. Note that the data aren't paired. Both arrays are already sorted for simplicity. Let the query point be $q = 4$, we expect $y_q \approx 10$.

Let's first estimate $P_X(4)$: the *cdf* algorithm will find $l = 2$ and $r = 4$. It then computes the linear interpolation:

$$d = \frac{q - X[l]}{X[r] - X[l]} = \frac{4-3}{5-3} = 0.5 \qquad P_X(4) = \frac{l + d \cdot (r - l)}{n} = \frac{2 + 0.5 \cdot (4-2)}{8} = 0.375$$

Now it's time to estimate $P^-1_Y(0.375)$: the *invcdf* algorithm will find $p = 0.375m = 3.75$, $r = 4$ and $l = 2$. By linear interpolation, it then finds

$$d = \frac{p - l}{r - l} = \frac{3.75 - 2}{4 - 2} = 0.875$$

$$P^{-}1_Y(0.375) = D[l] + d \cdot (D[r] - D[l]) = 7 + 0.875 \cdot (10 - 7) = 9.625$$

4. Application to glomerular filtration rate estimation. Existing multivariate formulas for GFR estimation have been compared and validated in [4] over a dataset of 86 Chinese children and adolescents aged 1 through 18, authors of this research have included the dataset with their publication. The most effective was found to be the Schwartz2009 function:

$$eGFR[mL/min] = 41.3 \cdot \frac{height[m]}{serumcreatinineconcentration[mg/dL]}$$

In order to apply SBR we must first assess the existance of a single dominant variable, this was clearly found to be the serum creatinine concentration, other variables used to estimate GFR are age and height, their effect however is marginal, as the scatter plots in Figure 4.1 reveal no strong statistical features.

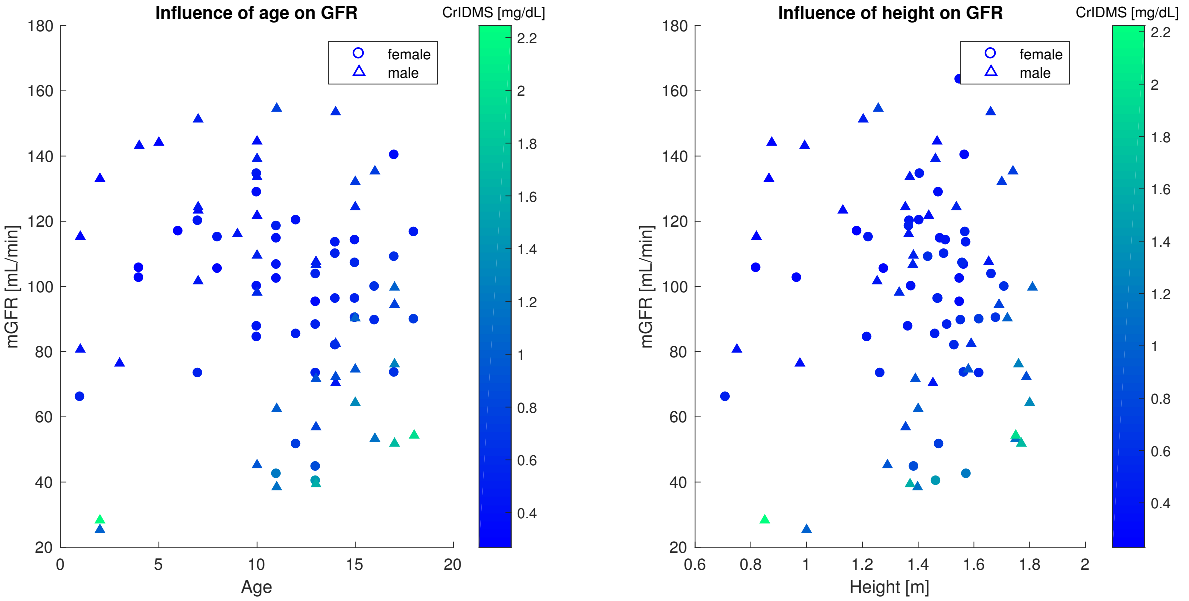


Fig. 4.1. *Plotting GFR as a function of Age or Height does not show very strong correlation.*

As SBR generates a univariate model, for the sake of the comparison, the Schwartz2009 equation was simplified to be independent of height, this was done by replacing the variable with a constant equal to the mean height of all individuals in the dataset. This model was plotted in dark blue in Figure 4.2, along with the datapoints, the regression curve obtained by SBR using the Numerical-Algebraic Neural System (NANS) method explained in [1] and that obtained using the binning-less method described in Section 2.

The four curves in Figure 4.2 are compared on prediction performance, i.e. how close along the vertical axis the regression functions come to the measured data, an index (MSE) for this is calculated by averaging the squares of all errors. The Shwartz2009 curve has a low MSE as it also takes height into account, the SBR regressions are both better as they we're obtained from the same data we are now validating them on, anyway this shows that SBR is very effective at fitting data. Comparisons were also made to evaluate the generality of the models obtained, a good way to evaluate this is by calculating the

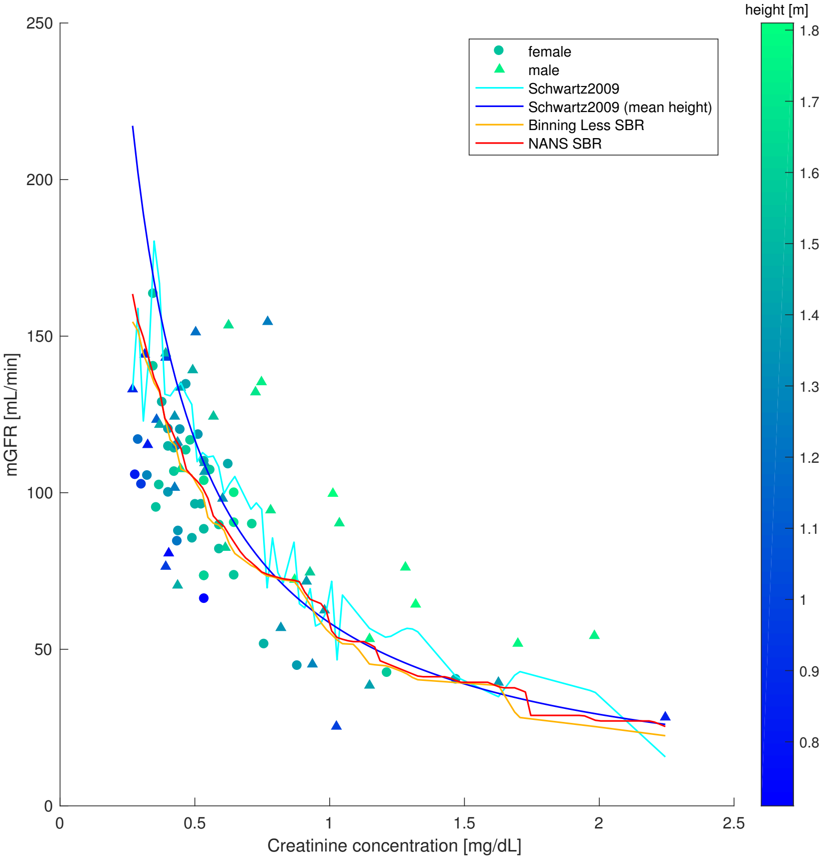


Fig. 4.2. *Data set with overlaid regression and estimation curves.*

roughness index

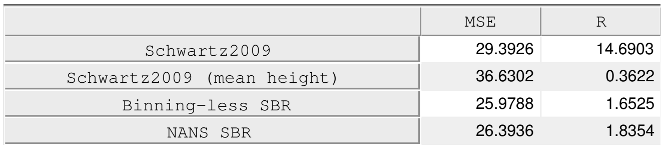
$$R = \sqrt{\sum_{i=3}^{n} \frac{(y_i - 2y_{i-1} + y_{i-2})^2}{n-2}}.$$

R is the root mean square of the second order differences of the sequence y_n so sharp changes in slope increase the value of R. An index similar to R is also used in [2] to prevent overfitting. The value of R is expected to be greater for irregular curves and indeed it is close to zero for the simplified Schwartz2009 model, which is essentially a hyperbola, graph of a smooth function. The MSE and R indices can be read in the table of figure 4.3.

5. Conclusions. Questa sezione riassume i risultati, sia concettuali che pratici, ottenuti.

REFERENCES

[1] S. Fiori, “Fast statistical regression in presence of a dominant independent variable”, *Neural Computing and Applications*, Vol. 22, No. 7, pp. 1367 – 1378, 2013



	MSE	R
Schwartz2009	29.3926	14.6903
Schwartz2009 (mean height)	36.6302	0.3622
Binning-less SBR	25.9788	1.6525
NANS SBR	26.3936	1.8354

Fig. 4.3. *Mean squared error with respect to measured GFR (MSE) and roughness(R) for the four estimation models.*

[2] C.M. Bishop, "Training with noise is equivalent to Tikhonov regularization", *Neural Computation*, Vol. 7, No. 1, pp. 108 – 116, 1995

[3] S. Fiori, T. Gong and H.K. Lee, "Bivariate nonisotonic statistical regression by a lookup table neural system", *Cognitive Computation*, Vol. 7, No. 6, pp. 715 – 730, 2015

[4] K. Zheng, M. Gong, Y. Qin, H. Song, X. Shi, Y. Wu, F. Li and X. Li, "Validation of glomerular filtration rate-estimating equations in Chinese children", *PLoS ONE*, Vol. 12, No. 7, pp. e0180565 (doi:10.1371/journal.pone.0180565), 2017