Applying the KSG Estimator to Mixed Continuous-Discrete Data in JIDT. By Daniel Fernandez (490105980)

1. Introduction and Literature Review

The mutual information or MI of a pair of random variables is an extremely widely-used concept in information theory: it is a quantitative measure of the information we gain about one random variable through observing the other. According to Ross (2014)[1]: '*MI will detect any sort of relationship between data sets whatsoever, whether it involves the mean values or the variances or higher moments*.'

The initial goal of this work is to validate a method for computing MI in cases where one variable is continuous, and the other discrete.

When making information-theoretic calculations, it is important to distinguish the two cases: one in which the underlying distribution of the data is known, and one in which it is not. In cases where the exact distribution is known, formulae such as the following may be used. (This is the basic form for two discrete variables; the modified form we use is seen in 2.III.)

$$I(X;Y) = \sum_{x,y} P_{X,Y}(x,y) * \ln \frac{P_{X,Y}(x,y)}{P_X(x)P_Y(y)} \{1\}$$

When the underlying distribution is unknown, an estimation algorithm must be used instead. The Kraskov estimator (known since Kraskov et.al. 2004[2], henceforth simply 'KSG') is a best-in-class solution for estimating mutual information on empirical data. Ross' paper developed a related method which can be used when only one of the variables concerned takes continuous values. It is in this paper that we were primarily interested.

2. Technical Background

2.I Ross' experiment architecture

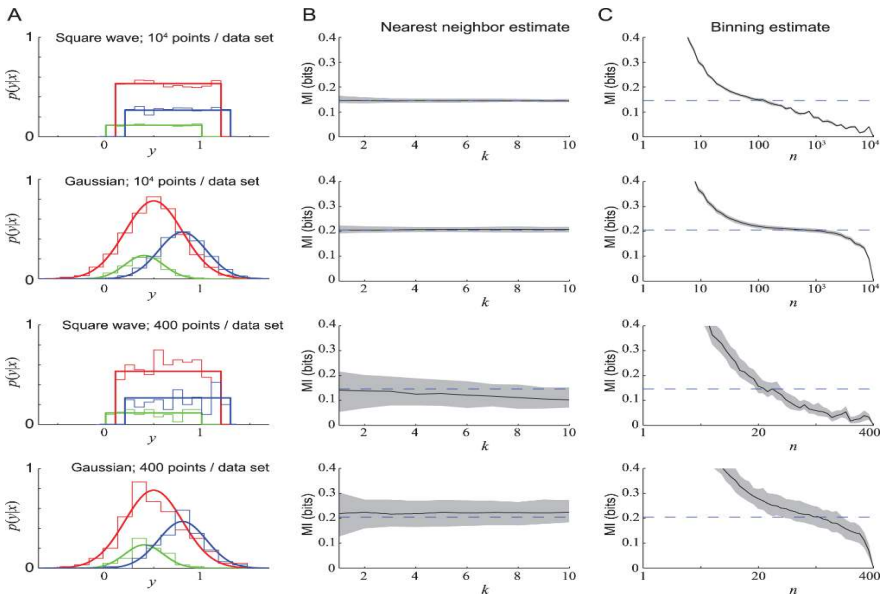


Figure 1: Ross' Mutual Information plots

Ross' main result relates to synthetic datasets whose true mutual information may be computed using a modification of **(1)** and compared with the estimator values. To achieve it, he created bivariate data with three possible discrete x-values (red, blue, green) and different continuous marginal distributions for the y-values such that visually we can see there is non-zero MI between the variables. The 1st column of plots in the figure above illustrates Ross' experiment design, while the 2nd is what we will use to check our estimator.

The red, blue and green curves are from the same 'family' in each case, and they overlap so that the information gained from knowing which curve is in use for generating a particular data point is partial. The square waves Ross used had the following properties:

Start	**End**	**Relative frequency**	**Corresponding X**
0	1	2/17	0
0.1	1.2	10/17	1
0.2	1.3	5/17	2

His Gaussian distributions had the following properties:

Mean	**Standard Deviation**	**Relative frequency**	**Corresponding X**
0.4	0.2	2/17	0
0.5	0.3	10/17	1
0.8	0.25	5/17	2

Each data point was assigned to one or other X-value (and corresponding marginal distribution for Y) by a Bernoulli process with the relevant probability (2/17, 10/17 or 5/17) and so the exact prevalence of each discrete value varies between the samples. He then applied a slight refinement of the KSG algorithm as described below. Then, he compared its results against a true value, obtained by numerical integration (see section 2.III below.)

2.II The Pure-Continuous KSG Algorithm

For two continuous variables, the KSG estimator uses the following algorithm:

For each point in a dataset of N points:

- *Define the smallest rectangle centred upon it such that its k nearest neighbours are contained within. Usually k=4 suffices, but more on this later.*
- *Extend the rectangle in the y-direction and count the number of points that fall within this extended rectangle to obtain n_y*
- *Extend the rectangle instead in the x-direction and do likewise to obtain n_x*

Take digamma functions of all results from steps II and III, then average over all N points.

Combine the results in the following fashion:

$$I_{KSG}^{(k)}(X,Y) = \psi(k) + \psi(N) - \langle \psi\left(n_x^{(k)}\right)\rangle - \langle \psi\left(n_y^{(k)}\right)\rangle \quad \{\mathbf{2}\}$$

Here ψ represents the digamma function (the derivative of the gamma function, which is in turn an analytic continuation of the factorial) and the < brackets denote expectation values.

NB: Due to this analytic continuation, there is no particular problem with having gamma (or digamma) functions of a non-integer, but there is no particular information-theory significance to talking about the 3.5 or 2.74 nearest neighbours of a point. It could be that some such idea is found in the future (similar to fractional calculus, for instance), but the subject matter of the KSG estimator is rich enough for our present purposes without inventing such a complication. Hence, it is only the individual points in the I-K plots that have any significance. It makes the most sense to show these with standard error bars and not with the interpolated 'shaded error area' that Ross does in Figure 1, Column B above.

As a sidenote, there are two versions of the KSG algorithm for pure continuous data. The simpler and more effective (according to CSYS5030) version is where the k neighbours may lie either within the respective rectangles in step 1, or on the boundary. (We will make the analogous assumption in the modification in section 2.IV as well.) Equation **(2)** is then a cute result that arises from exact evaluation of the integral in question.

2.III Modifications to Formulae/True MI

It appears we have left the sum/integral question vague for quite long enough. In the introduction we provided the following formula for MI of two discrete variables:

$$I(X,Y) = \sum_{x,y} P_{X,Y}(x,y) * \log \frac{P_{X,Y}(x,y)}{P_X(x)P_Y(y)} \ \{\mathbf{1}\}(\boldsymbol{Repeated})$$

When one variable becomes continuous, we change sums into integrals specifically as follows, replacing the density functions for X by the marginal or overall density functions:

$$I(X,Y) = \sum_{x} \int \mathrm{dy}.\, \mu_Y(X = x, y).\, P(X = x).\ln \frac{\mu_Y(X = x, y) * P(X = x)}{P(X = x)\mu_Y(y)} \ \{\boldsymbol{3a}\}$$

NB. For completeness, if we were to make X continuous as well, a second integral would be required, over the support of its probability distribution function:

$$I(X,Y) = \int dx. \int \mathrm{dy}.\, \mu_{X,Y}(x,y).\ln \frac{\mu_{X,Y}(x,y)}{\mu_X(x)\mu_Y(y)} \{\boldsymbol{3b, not\ used}\}$$

Equation 3a is the sum of a countable number (in our case, 3) of terms that look like this:

$$I_{PART}(X,Y) = \int \mathrm{dy}.\, \mu_Y(X = x_0, y).\, P(X = x_0).\ln \frac{\mu_Y(X = x_0, y)}{\mu_Y(y)} \ \{\mathbf{4}\}$$

This formula is especially important because we use it for numerical integrations in Section 4 to find the true MI. Note that Ross gives a different form, which lacks a probability mass function in the integrand and so is probably not equivalent. When we come to comparing

with KSG, it turns out that the formula for empirical MI in the mixed case is identical to that for the pure-continuous case (i.e. equation **(2)**) but the algorithm is subtly different.

2.III Modifications to KSG Algorithm

In section 2.I we discussed how an algorithm would work for pure continuous data. Obviously, there is a problem when it comes to drawing 'rectangles' in the discrete probability space (steps 2 and 3 of the algorithm), so the following modification is needed.

For each point P in a dataset of N points:

1. *Initially, consider only the points with the same X-value as P.*
   - *Find the smallest closed interval that contains P and its k nearest neighbours.*
   - *Note how many points are in this set (e.g. Ross' red set) and call it n_x.*
   - *This value will not be smaller than k.*

2. *Now, consider the full set of N points (i.e. all colours in Ross' plots)*
   - *Using the same interval as in 1.1 above, note how many points of all X-values lie within and call this n_y.*
   - *In general, this will be different to n_x, and not smaller than k.*

Take digamma functions of n_x and n_y, average over all N points and combine as in 2.II.

3. Python Algorithm

Luckily, it was not required to implement any of this by hand because it was already implemented in JIDT. As mentioned in sections 1 and 2.I, the first goal of this work was to validate the implementation by ensuring that it could replicate plots from Ross' paper. Thus, at this point it suffices to pass k as a parameter to JIDT. The pseudocode for doing so is given below, and the full code is provided on Google Drive & Github. (The generation of random data was nontrivial due to the prevalence values not being precisely fixed.)

Stage 1 (Setup)

I. Start a Java Virtual Machine in JPype (if not already started)
II. Input the desired relative abundances of the values taken by the discrete variable.
III. Input the type and parameters of the corresponding continuous distributions.
IV. Proceed to stage 2, which incorporates the loop over k.

Stage 2 (For each value of k)- so occurs 10 times per run if max(k)=10

I. Repeat the following 100 times:

   A. Generate N points based on the abundances (and on the continuous distributions.)
   B. Initialise and run the JIDT calculator using the new data and current value of k
   C. Use JPype to read off results (1 result per repeat)

II. Find mean and standard deviation and plot this as a single point on the I-K graph.

There follows a code fragment indicating how generation of SW and Gaussian data was done.

```
def GenerateSquareWaveData(ab_norm, start_points=starts,end_points=ends, size=1000):
    x=[] #these are not the same as the x-axis and y-axis that will be plotted later
    y=[]
    a=0
    list_of_sizes=[0]*len(starts)
    many_samples = random.choices(list(range(len(ab_norm))), weights=ab_norm, k=size)
    dict1=Counter(many_samples)
    for key, item in dict1.items():
        list_of_sizes[int(key)]= item
    while a<len(starts):
        i=0
        while i<list_of_sizes[a]: #ensuring our cts and dsc datasets have same len
            x.append(a)
            y.append(random.uniform(start_points[a],end_points[a]))
            i=i+1
        a=a+1
    return x,y, len(list_of_sizes)

#1. Generating continuous data
def GenerateSingleGaussianData (ab_norm, list_of_mu =m, list_of_sigma=s,size=1000):
    x=[] #these are not the same as the x-axis and y-axis that will be plotted later
    y=[]
    a=0
    list_of_sizes=[0]*len(list_of_mu)
    many_samples = random.choices(list(range(len(ab_norm))), weights=ab_norm, k=size)
    dict1=Counter(many_samples)
    for key, item in dict1.items():
        list_of_sizes[int(key)]= item
    while a<len(list_of_mu):
        i=0
        while i<list_of_sizes[a]:
            x.append(a)
            y.append(random.gauss(list_of_mu[a], list_of_sigma[a]))
            i=i+1
        a=a+1

    return x,y, len(list_of_sizes)
```

Figure 2: Data Generation Functions

4. Main Results

In (sub)sections 4.I and II below, I give the 'correct' plots using error-bars. In section III, the method for calculating the true MI will be given, and in Section IV a single pair of plots (the Gaussian data) will be selected for a full emulation of the Ross plots including true MI.

4.I 'Reliable' plots with 10,000 data points

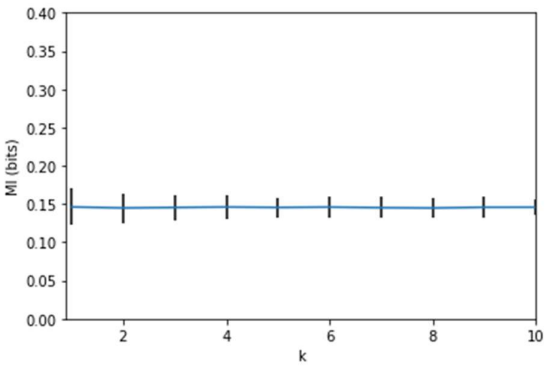


Figure 3: MI of Square Wave (N=10,000)

We can observe the correct behaviour here, in that the error bars become smaller for higher values of k, but the mean MI values are not materially altered by varying k within the range 4-10. Normally, in information theory we would use nats in place of bits, however in this case I wanted to facilitate an easy comparison with the Ross plots in Fig 1.

The value in both this plot and the next (0.15 bits and 0.21 bits) match Ross' values. Based on the consistency in the results for each value of k, as well as the standard deviations, we may be confident that we have achieved 2sf of accuracy. This will be checked against the true value (calculated from the underlying distribution) in section 3.III.

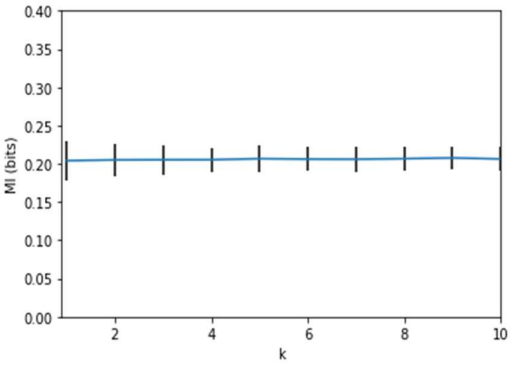


Figure 4: MI of Gaussian (N=10,000)

4.II Coarser plots with 400 data points

Compared to the larger plots, we should expect that these coarser plots exhibit greater uncertainty surrounding the value of the mutual information for both types of data. In particular, the central limit theorem (CLT) indicates that the error bars should be approximately 5 times as large in this case.

However, *a priori*, we still expect the values to be constant with k, which turns out to not be the case! A brief discussion of these plots follows in section 5.I. Also interesting was the work by Carrara (2020) [4], which identifies other failing conditions for KSG estimators.

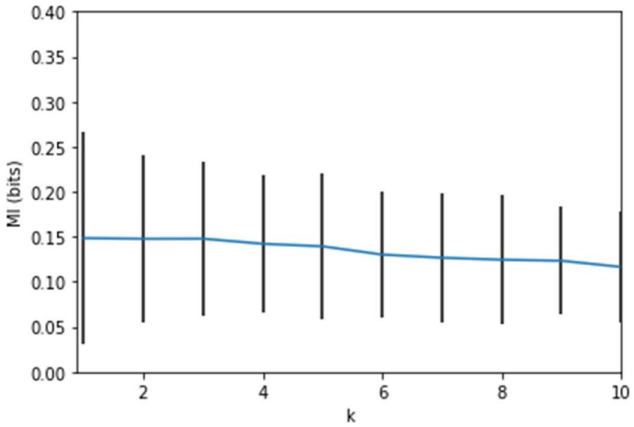


Figure 5: MI of Square Wave (N=400)

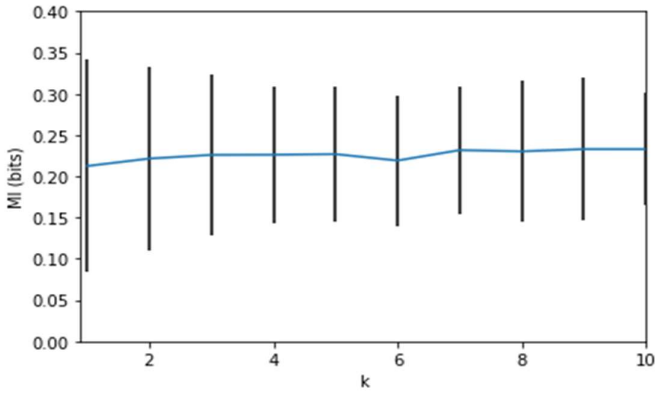


Figure 6: MI of Gaussian (N=400)

4.III True Value of Mutual Information

To confirm that we have the correct value and not merely a plot that looks like Ross', it is required to integrate equations such as #4 numerically. For instance, the code used to find the true MI of the square-wave data is shown below. The (near-)exact values of the true MI found by this method are 0.14587 bits (SW) and 0.20524 bits (Gaussian.)

```
from scipy.integrate import quad
import numpy as np

starts=[0, 0.1,0.2]
ends=[1,1.2,1.3]
list_abund=[2,10,5] # determines how many discrete values & their relative prevalence
weights=list_abund/sum(np.array(list_abund)) #normalizing

def mu_multiple(y,weights=weights,starts=starts,ends=ends):
    j=0
    subtotal=0
    while j<len(weights):
        length=ends[j]-starts[j]
        ind_result=weights[j]/length
        if y>starts[j] and y<ends[j]:
            subtotal+=ind_result
        j+=1
    return subtotal

def integrand1(y,j,weights=weights,starts=starts,ends=ends):

    den=mu_multiple(y)
    length=ends[j]-starts[j]
    num=1/length
    integrand=np.log(num/den)*(weights[j]/length)

    return integrand


def final(*arg):
    I = quad(integrand1,0,1.3)
    I=(I[0])*np.log2(np.exp(1))
    return I

print(final())
```

4.IV Exact Emulation of Plots With Shaded Error Areas

The following plots are intended to exactly emulate the Ross plots: the area shaded is that between the 10th and 90th percentiles of the obtained results.

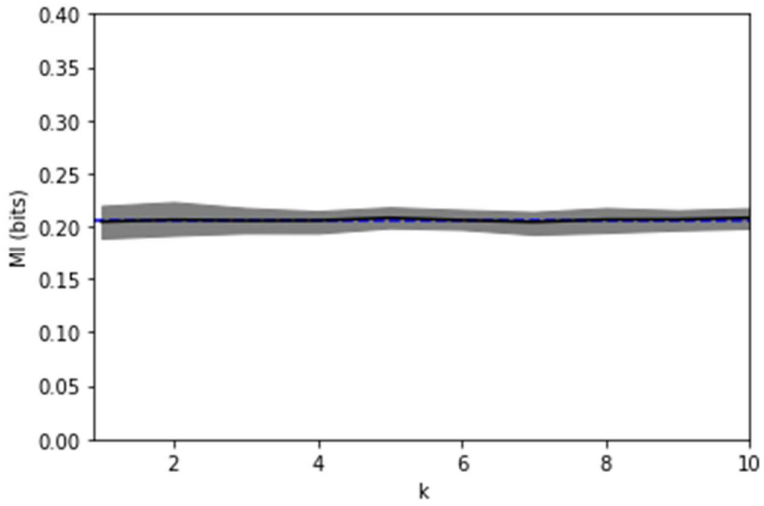


Figure 7: MI of Gaussian (N=10000) with shaded area

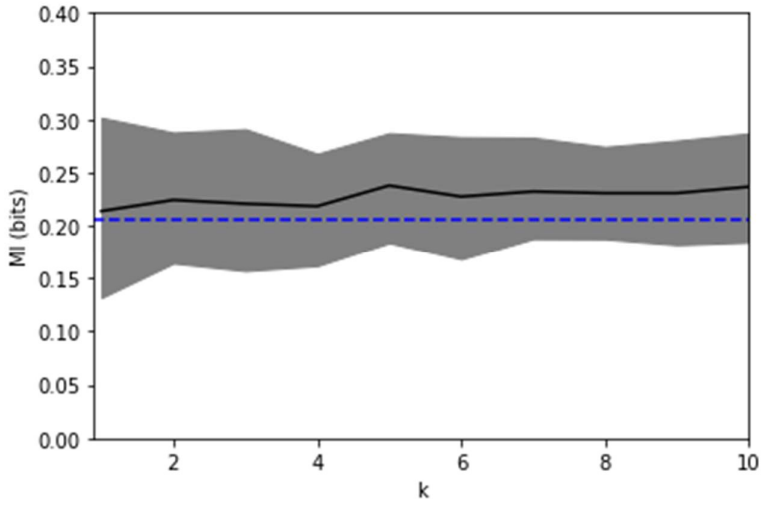


Figure 8: MI of Gaussian (N=400) with shaded area

The remaining 2 plots requiring emulation (square wave data with N=10,000 and with N=400) are provided in Section 7 (Appendix). This concludes the main portion of the report.

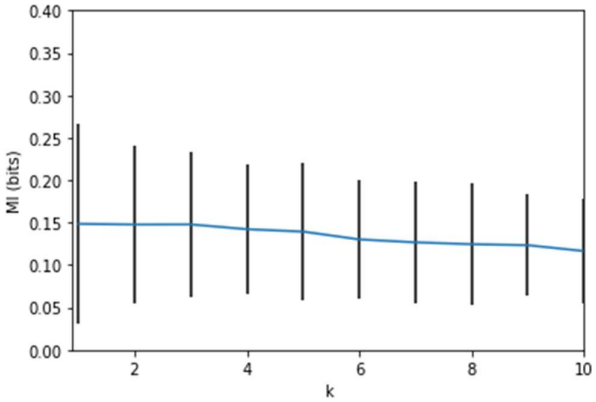
5. Follow-Up Questions And Results

In this section, we will explore extensions to the material above, which indicate some preliminary findings beyond the replication of the Ross paper.

5.I. Plotting Mutual Information Against N

In the original KSG paper, it was suggested that $k \sim \log_{10}(N)$ yielded the most viable estimates of the mutual information. Thus, for N=10,000 it is reasonable to set k=4. However (and also noted by KSG for pure-discrete data) the estimates tended to be robust against increasing k to 10. Where this gets interesting is at low values of N.

In the N=400 plots from section 3.II above, we seemed to be reaching the lower limit of N for usability: unless we keep k to 2 or 3, we saw a very subtle downward trend in the MI when we looked at the square-wave I-k graph. **This should not be happening!**



I was unable to discover what led to the downward trend (as compared to upward for the Gaussian example in 4.II) but the existence of any trend at all is indicative that the MI estimates are no longer stable. Comparing with the third plot in Fig. 1 corroborates the conclusion: Ross, too, experienced instability in the MI estimates for this small dataset.

The trivial k=1 case seems to be the closest to the true value, which is not encouraging. The question then arises of how the reliability of these MI estimates evolves for fixed k (=4, say) and N gradually increasing from 400 to 10,000. The transition from totally unreliable estimates to reliable ones should theoretically take place in a relatively smooth fashion. Of course, the stochastic elements of the sampling ensure that we don't observe this theoretical speed of convergence, but in Figure 9 below, we nevertheless see that it's possible to draw a reasonable best-fit line on the log-log plot (R^2 score: 0.7.) This is also the case for the Gaussian data, with the plot for that included below.

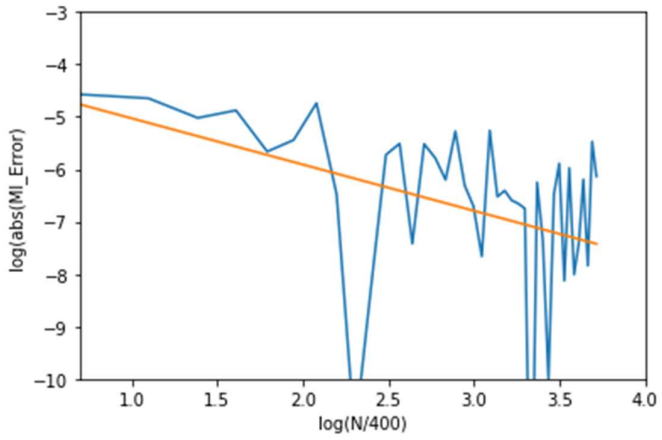


Figure 9: Log-log plot of error in MI estimator against N (Square Wave)

In all cases I investigated for either estimator, the slope of this line was close to -0.7 and the R^2 score of the line (a measure of how well the linear regression explains the data) increased monotonically with increasing the number of repetitions from 100 (see stage 2 under Python Algorithm in Section 3.) On a cautionary note, monotone increases in the R^2 score do not imply that its limiting value is 1.

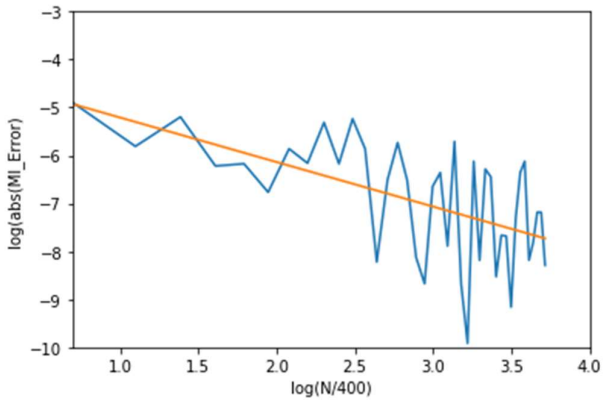


Figure 10: Log-log plot of error in MI estimator against N (Square Wave)

5.II. Conditional Mutual Information

Another related KSG-based estimator for datasets with mixed discrete-continuous types has been implemented in JIDT to calculate CMI or *conditional mutual information*; that is, the mutual information of Y given X, assuming that we already knew Z. In mathematical terms, we condition upon Z throughout equation 3:

$$I(X,Y|Z) = \sum_{x} \int \mathrm{dy}.\, \mu_Y((X=x,y)|Z).\, P(X=x|Z).\ln \frac{\mu_Y(X=x,y|Z) * P(X=x|Z)}{P(X=x|Z)\mu_Y(y|Z)} \quad \{\mathbf{5}\}$$

A couple of sanity tests are reasonable for testing any estimator of CMI:

i) If Z is an unrelated N(0,1) variable, then $I(X,Y|Z) = I(X,Y)$.
ii) If $\mathrm{Y} = \mathrm{Y}_0 + \mathrm{Z}, \mathrm{Z}{\sim}\mathrm{N}(0,1)$, then $\mathrm{I}(\mathrm{X};\mathrm{Y}|\mathrm{Z}) = \mathrm{I}(\mathrm{X};\mathrm{Y}_0)$.

However, I ran into problems with the first sanity test. Using Ross' two distributions as Y in turn, and using an uncorrelated N(0,1) variable for Z, yields the following:

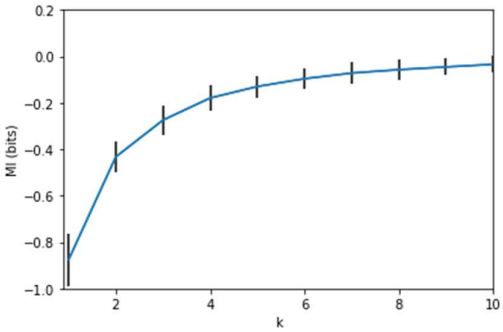


Figure 11: CMI: Y is Ross' square wave, Z is N(0,1)

A similar roadblock occurs if we attempt to apply the method to a Gaussian. The blue dotted line indicates the value from Section 4 above, i.e. what we would hope to see.

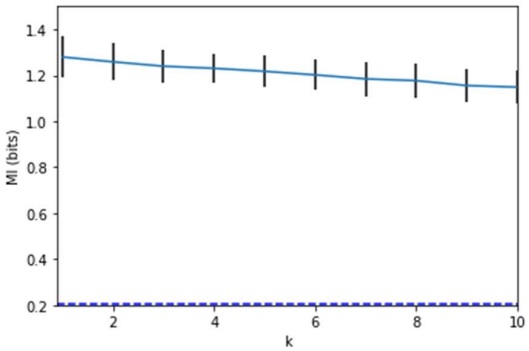


Figure 12: CMI: Y is Ross' Gaussian, Z is N(0,1).

In this case the values look like they could be reassuring if only the MI were 1 bit lower, which initially aroused the suspicion that my Python routine had added a rogue bit somehow. However, this can't be the case for several reasons:

i) The nats-to-bits conversion works in all other cases and occurs only at the plotting stage of the Python script. So the 'rogue' number is actually non-integer.
ii) The downward trend does not end at k=10.
iii) A bizarre MI trend was also seen in the Square Wave case above.

The logical conclusion from the above is that the CMI calculator exhibits a strange bias of unknown origin, if Z is an uncorrelated Gaussian variable. It might, however, be correctly implemented if Z is correlated to Y. I performed some further calculations of Conditional Mutual Information for the case where Y and Z are both drawn from (different dimensions of) a multivariate Gaussian. In this case, the CMI values were believable for k=4 and agreed with numerical integration. However, I think the estimator needs further trials with respect to the above sanity tests before we can assert that the method works for correlated Z.

6. Other Remarks

6.I Conclusions

The main aim of this paper was achieved. **The JIDT estimator achieves the correct answer for Mutual Information of two different mixed discrete-continuous variable setups.** It agrees with both Ross' findings and with an independent numerical integration setup and so we can assert its validity with a high degree of confidence.

Additionally, we reached the point of modelling just how the accuracy of the estimator increases (i.e. the error converges to 0) as N increases. At the very last moment before completing the report, I realized why the slope in Figure 10 (~ -0.74) appeared to be uninformative. It was because the mutual information was still scaled in bits! Dividing through by the requisite factor of $\log_2(e)$ gives us a much more believable -0.51. **Thus, in accordance with the Central Limit Theorem, the expected absolute error in the estimator scales as the inverse square root of N.**

6.II Remaining Questions & Further Work

As indicated in section 4.II, it would be interesting to know why the I-K graph exhibits a trend for both distributions when N=400. In particular, coincidences are rare in applied mathematics and so there will be a reason for why these trends are in opposite directions.

Furthermore, in Section 5.II, the square wave and Gaussian CMI graphs again displayed (unwanted) trends in opposite directions. It would be interesting to know whether those two facts are related, and (if so) what properties of their underlying pdf's result in this feature. As an applied mathematician, I am inclined to wonder whether this has anything whatsoever to do with the Gibbs phenomenon in Fourier theory.

7. Appendix: Two more plots to corroborate Ross

In both plots below, the true MI value has been added as a dotted blue line (as in 4.IV.)

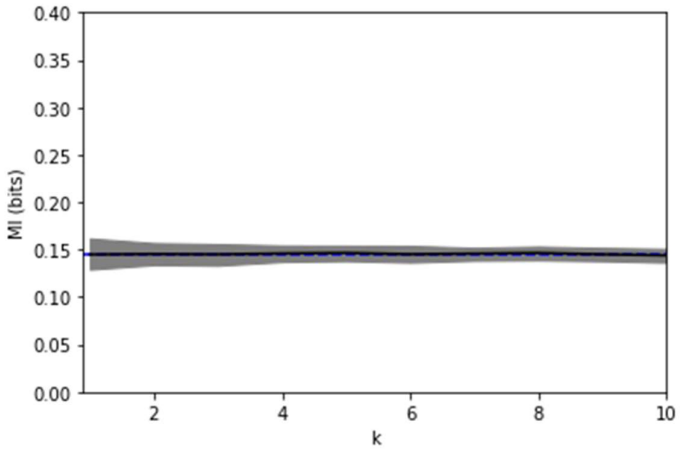


Figure 13: MI of Square Wave (N=10000) with shaded error area

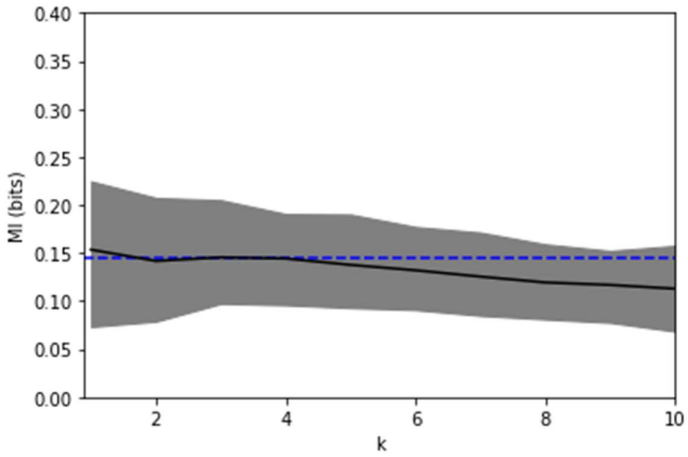


Figure 14: MI of Square Wave (N=400) with shaded error area

References

1. Ross, B. (2014). *Mutual Information between Discrete and Continuous Data Sets*. Online academic article retrieved from https://doi.org/10.1371/journal.pone.0087357
2. Kraskov, A; Stoegbauer, H; Grassberger, P. (2004). *Estimating mutual information*. Phys. Rev. E 2004, 69, 066138
3. Lizier, J. (2014). *JIDT: an information-theoretic toolkit for studying the dynamics of complex systems*. Frontiers in Robotics and AI 1:11, 2014
4. Carrara, N; Ernst, J. (2020). *On the Estimation of Mutual Information.* MDPI Proceedings 2019, 33.