Silent error analysis in density estimation pipelines

Hadrien Croubois *

ENS de Lyon
Argonne National Laboratory

Abstract

write

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 Introduction

write

...

2 Pipeline description

The main objective of the pipeline is to build an analyse a density field from a sampling representing the density field. In real application this sampling can be data comming from the acquisition of a real experiment or the the result of a simulation program such as the Hacc cosmology simultation code.

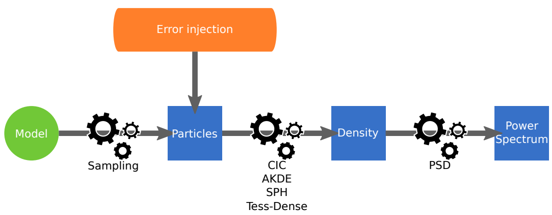

Figure 1: Full pipeline

2.1 Syntetic particles generation

In order for us to have a good idea of the expected results, we will use syntetic particles generated by the sampling of an analytical distribution with caracteristics close to thoses expected from the real application values. That way we have analytical values helping us compared all our results to a known ground truth.

This analytical distribution is a density profil produced by a sum of Navarro-Frenk-White functions [Navarro et al., 1996]. This density profile is further caracterized in .

ref

$$\mathcal{D}(p) = \sum_{\{k_i, c_i\}} \frac{k_i}{\|p - c_i\|(\|p - c_i\| + 1)^2}$$

We will also use values produced by real application in order to confirm the results comming from the synthetic particles.

2.2 Density estimation

Density estimation aims at reconstructing the density function from a set of samples. This is achieved by adding the contribution of all sample, each sample representing a local density profile. The distribution of this local density profile is caracteristed by a weighting function and by a domain, which caracteristics have been extensivelly studied.

ref

However, more accurate methods also tends to be more computationnaly expensive and not so qualitative methods tends to be prefered in cases where intermediate quality results are satisfactory and computation cost needs to be limited.

We therefore will consider different methods representing differents compromise between results quality and computation cost.

2.3 Density analysis

In order to compare different density fields produced by the density estimation methods, we need a metric caracterising the representative elements of those density fields. This metric should differentiate values according to the relevant elements in regard to their futher analysis in real application.

For this step, the use a power spectrum analysis which makes visible variation responses for different caractèristic length, hence detecting bias such as high frequency noise or over smoothness.

*`hadrien.croubois@ens-lyon.fr`

3 Density estimation methods

Several methods exist to perform density estimation. Those methodes give result with different level of quality for different computational cost. This section will focus on describing some of those methods, which represent different compromise.

3.1 Methods description

3.1.1 CIC

CIC [Birdsall and Fuss, 1969] (Cloud in cell) is the simpler and cheapest of all methods presented in this paper.

This method simply distribute each sample weight among the closest grid points. This method has a very low computational cost ($\mathcal{O}(n)$) but suffer from a very low quality in parse areas where each sample should be distributed over a larger area, resulting in a lot of noise.

3.1.2 AKDE

AKDE [Rosenblatt, 1956] [Parzen, 1962] (Adaptive Kernel Density Estimator) methods are the natural variant of CIC.

To try and solve the high quantity of noise, Kernel Density Estimator uses large windows over which they distribute the weight of each sample. Those weight follow a kernel function, typically gaussian function.

In order to avoid noise in parse areas and oversmoothing in dense areas, the windows sizes are follow the local density. Windows adaptation criterions have been extensivelly studied [Heidenreich et al., 2013] but can, in some cases, dramatically increass the computational cost.

Our implementation uses a kd-tree construction for selecting the window sizes. Overall the complexity is $\mathcal{O}(n \log n + g^3)$ with far better results then CIC (cf 3.2).

3.1.3 SPH

write

...

3.1.4 Tess-Dense

Unlike all previous algorithm, Tess-Dense doesn't use fixed shape windows. Indeed, Tess-Dense uses a voronoï tesselation to determine each sample influence domaine. This precomputation is independant of the result grid's resolution therefore reducing the memory footprint on high resolution reconstruction. This cost is however paid even for small resolutions.

The cost of this method is $\mathcal{O}(n^2 + g^3)$, with results suffering from some noise. This noise comes from the NGP(Nearest Grid Point) approche when voronoï cells are to small to include any grid point (high local density) which can be reduced by using CIC in such cases and by geometric instability of voronoï cells in sparse areas.

3.2 Methods results

To compare thoses different methods results we plot the frequencial reponse of their result and compare them to the analytical result. One should not forget that the input set of those methods are sampling of a density function. As those sampling are inherently random, we cannot just observe on sampling and the associated result.

In order to take this randomness into account we computed several samplings of our density function model and then display both the median of the frequencial results overs thoses samplings as well as extreme values. That way we can not only describe the expected quality of those methods but also the natural variability we can expect. Figure 2 present thoses results for all prevously described methods.

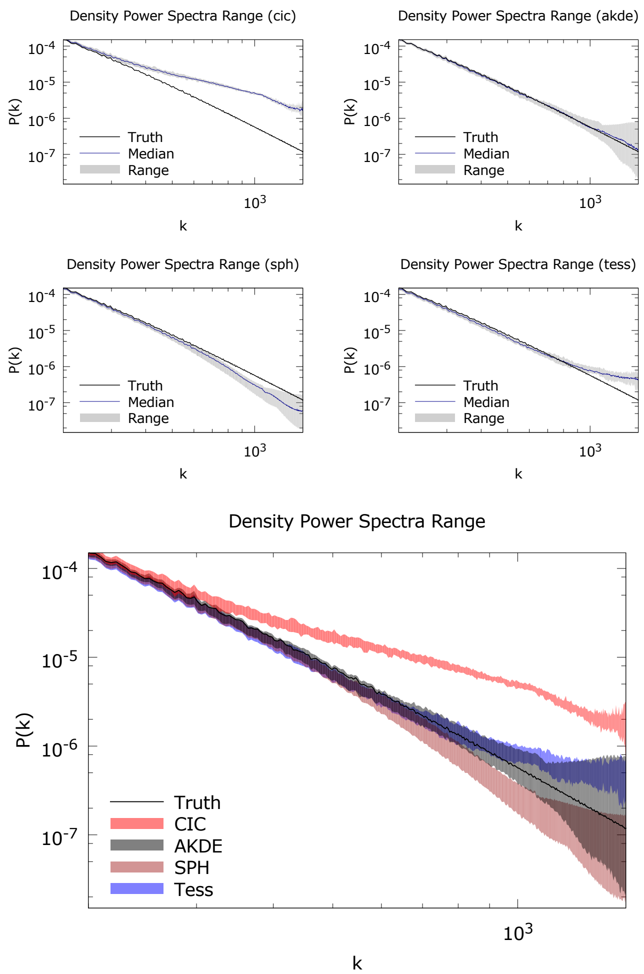

Figure 2: Spectral analysis of density estimation methods

Thoses results clearly shows the noisiness of CIC. As for the other methods, they give different shape of results with different level of variability while not showing one as clearly better than the others.

Tess-Dense's slight noisiness might be related to the use of flat weight distribution among each cell's inner grid point. Secondary methods used for weight distribution in dense area are also subject to some aliasing and therefore noise.

On the AKDE/SPH side, we mostly notice a high intrinsic variability. That may be caused by the sampling's random variation affecting the window size computation in such a way that we could locally encounter noisyness or oversmoothness.

4 Error modelisation

Memory corruption related error can happen in different ways. Cosmic rays and radiation have, for a long time, been suspected of creating random data . More rescently we realised that thoses error can also happen because of hardware issues.

Beyound the question of those corruptions causes, we are here focussing on the impact such event have on our pipeline.

4.1 Error classification

Depending on a large number of factor, memory corruption can have very different concequences. Such factor include many things from hardware components (ECC memory one of the most well know mecanism against memory correption) to software certification and computation redondancy.

As for the consequences, they can be divided into two different categories :

Hard errors: Memory corruption affect the system of the program flow will most likely cause dramatic errors such as the program stopping abruptly or the whole system failling. In such cases we do not get any results back, redering irrelevant the question of the result's validity. Some memory corruptions affecting critical data such as table indices also falls into this category.

Silent errors: Memory corruption in some application's data may not cause any crash of the application while affecting the results if not corrected by the hardware. This is particularly the case of large array's contents like particles positions in our pipeline.

In this section we will try to caracterise silent errors' impact on our pipeline's results.

4.2 Error injection

Simulating random memory corruption can be done by voluntarly modifying our applications data by randomly flipping bits. While this isn't hard to do, studying silent errors means we have to enshure that those random bit flips will not cause hard errors.

Differents software quality implies different level of tolerence. For exemple some code can handle particles positions being outside of our considered domain while other might crash. Form here on we will mostly focus on our AKDE implementation as it very resistant to such errors and therefore more suitable to create and analyse silent errors.

rderror
io for
s ?

In our model, simulating memory corruption is achieved by randomly modifying our input data set. In real application this dataset would be provided by another program and would have therefor been send through the network of saved on disk, increasing the probability of memory corruption. This injection model will be controlled by two parameters, on the one hand the number of bit flips and one the second hand the weight of the potentially affected bits.

While the first parameter is used to simulate different degre of corruption, the second one is used to study the impact of differents bit flip position. The construction of IEEE floating point arithmetics [Kahan, 1996] (IEEE 754) is such different bit modification produce different arithmetic modification.

Studies have shown that in some HPC pipelines, some bits' position are critical, the modification of those bits resulting in hard errors while some other bits' modifications are un noticeable as the resulting modification are bellow the accuracy of floating computation.

ref

4.2.1 Single error injection

...

to writ

4.2.2 Multiple error injection

...

to writ

5 Results

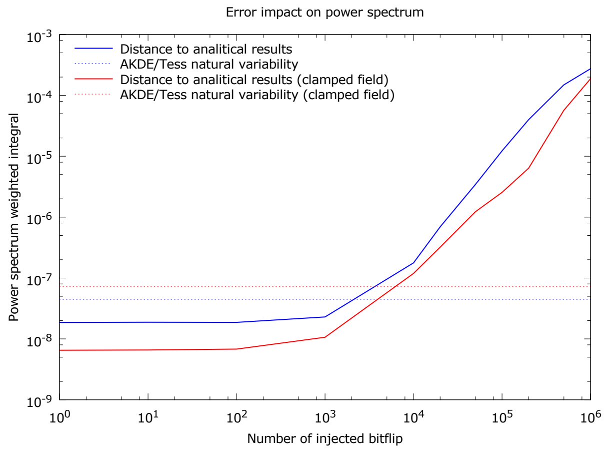

Figure 5: Power spectrum displacement

...

to writ

6 Conclusion

...

to writ

References

[Birdsall and Fuss, 1969] Birdsall, C. K. and Fuss, D. (1969). *Cloud-in-Cell Computer Experiments in Two and Three Dimensions.*

[Heidenreich et al., 2013] Heidenreich, N. B., Schindler, A., and Sperlich, S. (2013). Bandwidth selection for kernel density estimation: A review of fully automatic selectors. *AStA Advances in Statistical Analysis*, 97:403–433.

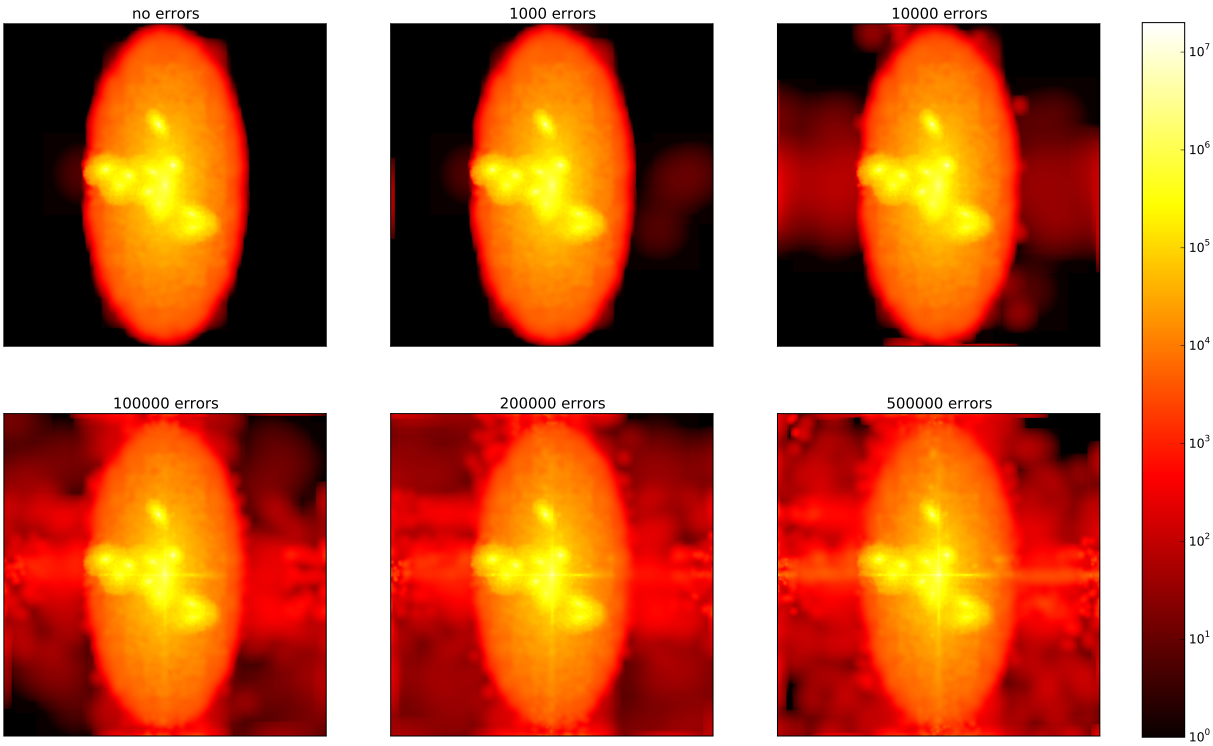

Figure 3: AKDE density fields after error injection

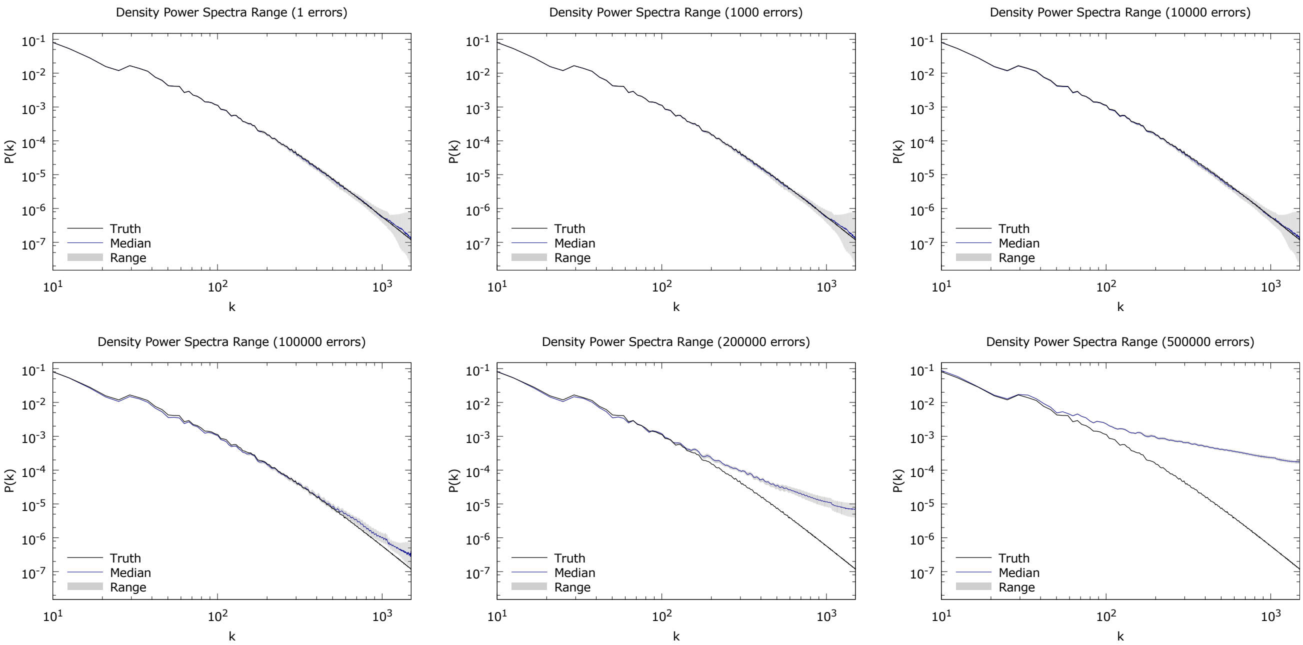

Figure 4: Bitflip influence on AKDE power spectrum range

[Kahan, 1996] Kahan, W. (1996). IEEE standard 754 for binary floating-point arithmetic. *Lecture Notes on the Status of IEEE*, (May 1995):1–23.

[Navarro et al., 1996] Navarro, J. F., Frenk, C. S., and White, S. D. M. (1996). The Structure of Cold Dark Matter Halos. *Astrophysical Journal*, 462:563.

[Parzen, 1962] Parzen, E. (1962). On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33:1065–1076.

[Rosenblatt, 1956] Rosenblatt, M. (1956). Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27.