EMD performance comparison: Single vs Double floating points

Dawid Laszuk, Oswaldo Cadenas, Slawomir J Nasuto
University of Reading, Reading, United Kingdom
Email: d.laszuk@pgr.reading.ac.uk, {o.cadenas, s.j.nasuto}@reading.ac.uk

Abstract—**Empirical mode decomposition (EMD) is a data-driven method used to decompose data into oscillatory components. This paper examines to what extent the defined algorithm for EMD might be susceptible to data format. Two key issues with EMD are its stability and computational speed. This paper shows that for a given signal there is no significant difference between results obtained with single (binary32) and double (binary64) floating points precision. This implies that there is no benefit in increasing floating point precision when performing EMD on devices optimised for single floating point format, such as graphical processing units (GPUs).**

Index Terms—**Empirical Mode Decomposition, Floating Point Arithmetic, Intrinsic Mode Function, Performance Test, Signal Decomposition**

I. Introduction

Nowadays, computers used for signal processing are so powerful, that many researchers do not think about the amount of data or its format. Most calculations may be performed in very high precision format, like double floating point (DFP) precision. This often is unnecessary, but since it often has little impact on computational time, it is kept for the sake of high precision. For some systems changing format, into i.e. single floating point (SFP), can greatly reduce the computation time. An example of such a device is graphical processing units (GPUs), which are reported to work several times faster using SFP precision instead of DFP [1].

Empirical mode decomposition (EMD) [2] can be computational intensive and not suitable for real-time analysis [3]. Introduced over 15 years ago it is still under constant exploration and development. Due to its empirical nature, i.e. being described by an algorithm, the method might to be very susceptible to data format. This can be seen, for example, in discussion lead by Rilling et al. in [4], where the smallest sampling frequency was considered in order for EMD to work.

The aim of this paper is to show whether and how data representation affects EMD. This is demonstrated on three examples – a sum of harmonic functions and a Gaussian noise. Section II introduces EMD algorithm, section III describes floating point precisions, section IV provides numerical experiments with conclusions in section V.

Manuscript received April 15, 2015; revised May 4, 2015; accepted May 12, 2015.

II. Empirical Mode Decomposition

Empirical mode decomposition (EMD) is a data-driven method for decomposing signals into oscillatory components called intrinsic mode functions (IMFs) [2]. The algorithm for EMD can be described as follows:

1) Identify all local extrema (both minima and maxima) in a given time series $s(t)$, that is the points, at which the derivative is zero, $ds(t)/dt = 0$.
2) If the number of extrema is less or equal than two then $s(t)$ is considered as a trend – a low frequency modulation – and the algorithm stops (trend $r(t) = s(t)$).
3) Use local maxima and local minima to compute respectively upper (e_{max}) and lower (e_{min}) envelopes. Interpolations are performed using natural cubic splines.
4) Calculate the instantaneous mean of the signal, defined as an average of both envelopes, $m(t) = \frac{1}{2}(e_{max}(t) + e_{min}(t))$.
5) Remove computed mean from the input time series, $h_j(t) = s_j(t) - m_j(t)$. This step is called *sifting*, because it subtracts the previous trend from fast varying components.
6) If residue $h_j(t)$ fulfils a given stopping criteria, then it is considered to be an IMF (component $c_j(t)$) and the procedure is repeated for modified time series, $s_{j+1}(t) := s_j(t)$.

As a result, EMD decomposes signal $s(t)$ into a set of N oscillatory components $c_j(t)$ (IMFs) and a trend function, $r(t)$. In the original paper, a stopping criterion is defined as a moment, when standard deviation of two consecutive iterations is below a predefined threshold. Each obtained component has an oscillatory form $c(t) = a(t)cos(\omega(t)\,t)$, where $a(t)$ and $\omega(t)$ are in general functions of time. In order to classify a function as an IMF, it needs to satisfy two conditions: 1) the number of extrema must be even or differ at most by one to the number of zero-crossings; 2) at any point, mean of top and bottom envelopes must be zero.

For the purpose of this paper, as originally suggested, EMD was used with natural cubic splines as an envelope's interpolation method. Local extrema were

considered to be vertices of parabolas interpolated on potential extrema, as was suggested in [5]. As a stopping criterion we have chosen moment, when potential component fulfils IMF conditions for ten consecutive sifting iterations.

III. Double vs single floating point precision

Floating point precision is related to the number of bits used to represent a digit in a computer. According to IEEE 754-2008 standard [6], floating point digits are represented as

$$V_{decimal} = (-1)^S M \cdot 10^E \quad (1)$$

where S is a sign bit, M is a mantissa value and E is an exponent. Depending on the precision format different numbers of bits are assigned to correspond to mantissa and to exponent. For single floating point precision (officially referred as binary32) and double floating point precision (binary64) standard dedicates 32 and 64 bits respectively. The exact number of bits for each part is presented in Table I.

TABLE I.
NUMBER OF BITS DEDICATED FOR DIFFERENT FORMAT REPRESENTATIONS.

name	sign	exponent	mantissa
binary32	1	8	23
binary64	1	11	52

Due to a finite number of bits representing a digit, it might happen that there is no representation for a given real value. If this occurs, then the value will be rounded to the nearest possible value. This also means that for each precision format a value ϵ exists, below which values will not change the result. The smallest positive value ϵ, which when added to one increases its value, i.e.

$$\epsilon: \ 1 + \epsilon > 1, \quad (2)$$

is called *machine epsilon*. For floating point with base b and precision p the machine epsilon is represented with a formula

$$\epsilon = 0.5 \cdot b^{1-p}. \quad (3)$$

Formats binary32 and binary64 have actually one bit more of precision than it is implied by their mantissa. This gives, according to IEEE 754-2008 standards [6], machine epsilon $\epsilon_s = 2^{-24} \approx 5.96 \cdot 10^{-8}$ and $\epsilon_d = 2^{-53} \approx 1.11 \cdot 10^{-16}$ respectively for single (binary32) and double (binary64) floating point precision.

IV. Experiment

Examples were generated and analysed in Python programming language. Numerical manipulations were performed using NumPy [7] scientific package. Source code of EMD implementation can be obtained from one of the authors' web-page [8].

It needs to be pointed out, that the interpolation techniques depend both on points' values and their positions. This means that the difference between two sets will be even greater if one compares values at different positions. When analysing signals, one is advised to scale independent variable appropriately, so that it has exact numerical representation. In binary floating point precision this means to assign a step value to be a multiple of power of 2 ($m\ 2^p$).

Please note, that conducted experiments are meant to show whether the difference between two floating points formats exists and what is the scale of difference.
The authors do not intend to comment on the meaning of the decomposition as it has already been discussed elsewhere in the literature [9], [10], [11].

A. Example 1

As a first experiment signal $s_1(t)$ was generated as a sum of cosines with different frequencies and phases, i.e.

$$s_1(t) = A \sum_{i=1}^{5} cos\,(2\pi f_i t + \varphi_i), \quad (4)$$

where frequencies and phases are respectively f_i = {6.1, 9.4, 12.7, 16, 19.3} and φ_i = {0, 1, 2, 3, 4}. Moreover, value A was assigned such that the $max(|S|) = 1$. This normalisation was performed for easier comparisons between results of presented examples. The particular set of frequencies and phases was chosen so that components are not harmonic of one another and their initial values are different. The signal was generated with time t in the range [0, 1] with sampling frequency of 1024 Hz and is visualised in Fig. 1. Its EMD decomposition is shown in Fig. 2, where solid line and dashed indicate DFP and SFP respectively. As it can be seen, two sets are visually ideally overlapping each other. In order to visualise the difference more clearly, the set obtained with SFP was projected onto DFP, since it has higher precision, and subtracted from the DFP set. The difference between corresponding IMFs is presented in Figure 3. The biggest difference is in order of 10^{-6} which, although almost two orders of magnitude bigger than the machine epsilon for SFP (sec III), is still five orders of magnitude smaller than the $s_1(t)$ signal. Thus, unless such small values are expected from analysis of the experiment, it can be considered as a negligible noise; they have no meaningful effect on the results.

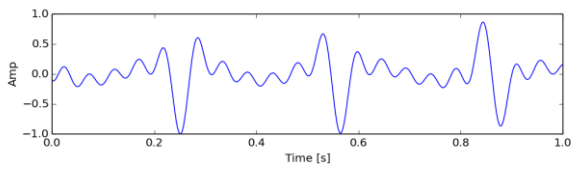

Fig. 1. Signal used in example 1. It is generated according to formula (4).

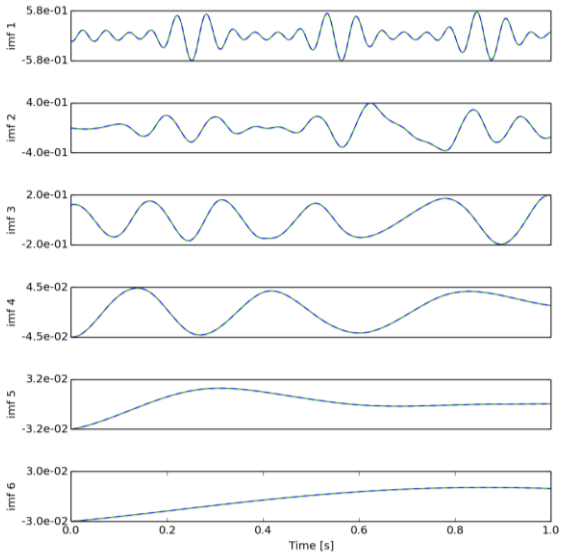

Fig. 2. EMD decomposition of signal from example 1 (Fig. 1). Decomposition for DFP and SFP are overlapped respectively with solid and dashed lines.

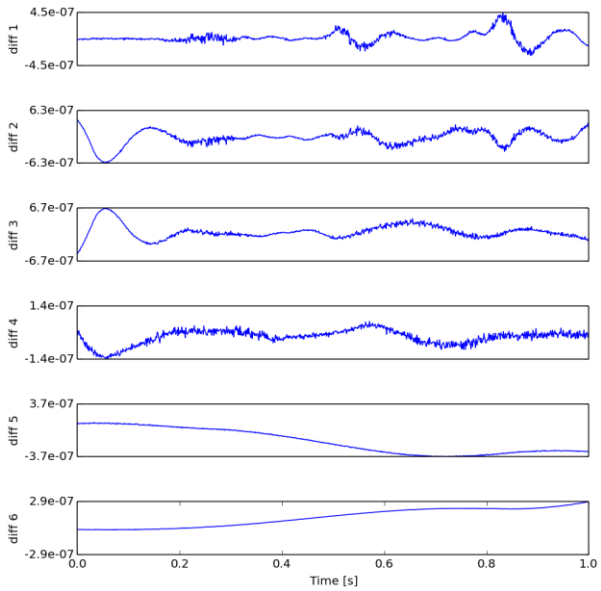

Fig. 3. Difference between SFP and DFP sets of EMD decompositions from example 1. SFP were first projected onto double precision and then subtracted from EMD DFP set.

B. *Example 2*

For the second example we generated random data characterised Gaussian noise, i.e.

$$s_2(t) = \mathcal{N}(\bar{x} = 0, \sigma = 1), \tag{5}$$

with zero mean and standard deviation of 1. The generated signal, consisting of 1024 points, was then normalised so that the biggest amplitude value is one. Such signal is presented in Figure 4. Its EMD decomposition is shown in Figure 5 using solid line and dashed lines for DFP and SFP, respectively. Again, not much difference between two sets is visually noticeable. Additional plot (Fig. 6) was generated, where the difference for each individual IMF is highlighted. In this example the biggest range of the difference has the order of magnitude six. However, again, comparing to the input signal $s_1(t)$ it is six orders of magnitude smaller and can be considered as a noise.

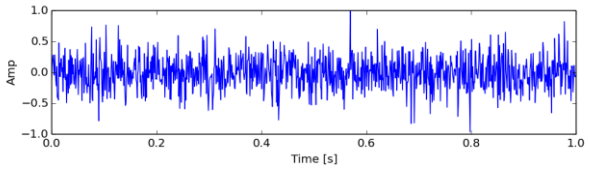

Fig. 4. Generated signal used in example 2. It is made of 1000 random points drawn from Gaussian distribution with mean 0 and standard deviation of 1.

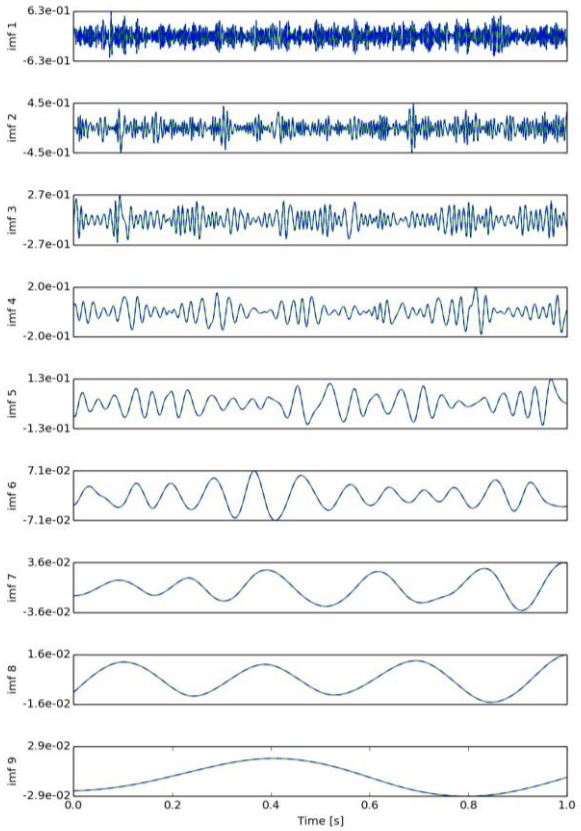

Fig. 5 EMD decomposition of signal from example 2 (Fig. 4). Decomposition for DFP and SFP are overlapped respectively with solid and dashed lines.

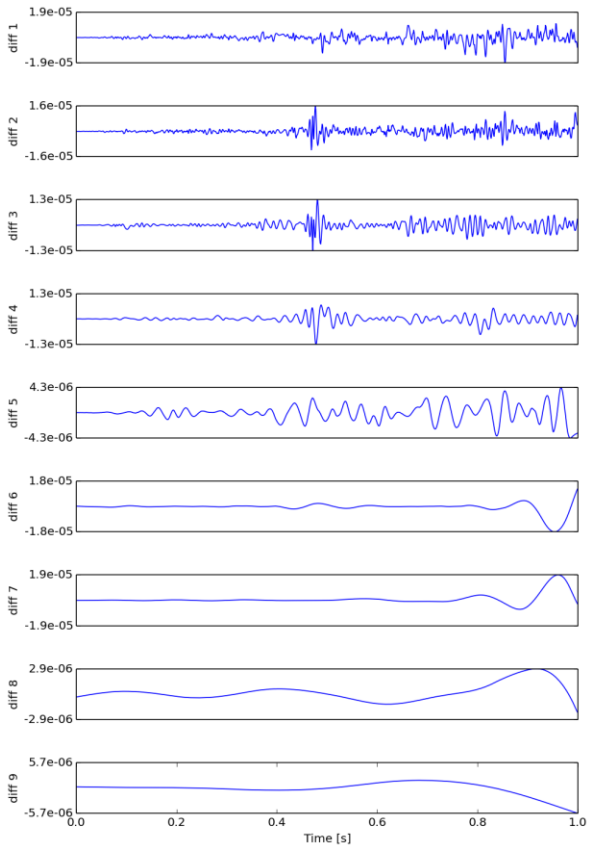

Fig. 6. Difference between SFP and DFP sets of EMD decompositions from example 2. SFP were first projected onto double precision and then subtracted from EMD DFP set.

C. *Example 3*

The final example is presented on a single channel of real EEG data. Recordings were obtained during resting state, i.e. when person was not involved in any physical, nor mental activity. For analysis, a four seconds segment of signal, sampled at rate of 128 Hz, were chosen randomly. Before the EMD decomposition was performed the signal was preprocessed, i.e. the mean value was removed and the amplitude was scaled, so that the highest amplitude is 1. Also, to decrease the error along time axis, values were scaled into range $t \in [-1, 1]$ with sampling frequency 256 Hz. Signal used for decomposition is presented in Fig 7.

Set of IMF components obtained from EMD is shown in Fig. 8 using solid line and dashed lines for DFP and SFP, respectively. The difference between corresponding

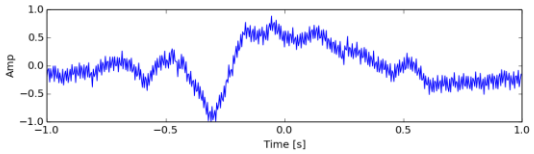

Fig. 8. EEG data used in third example. Processing involve removing mean and scaling amplitude, so that the maximum deflection is 1. Time scale changed to span from -1 to 1 with sampling frequency 256 Hz.

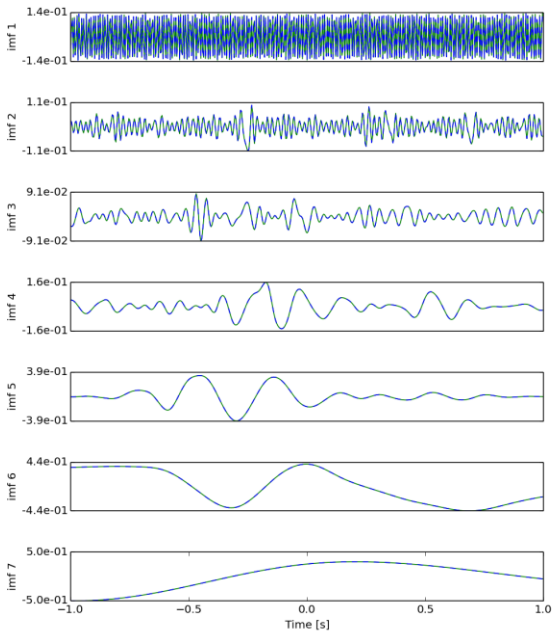

Fig. 7. EMD decomposition of EEG signal from example 3 (Fig. 7). Decomposition for DFP and SFP are overlapped respectively with solid and dashed lines.

IMFs is displayed in Fig. 9. In this example, as it was also shown in the two previous, the difference is very small, when compared to the amplitude of input signal. Again, the range of difference has the order of magnitude -6 and it is similar for all comparisons.

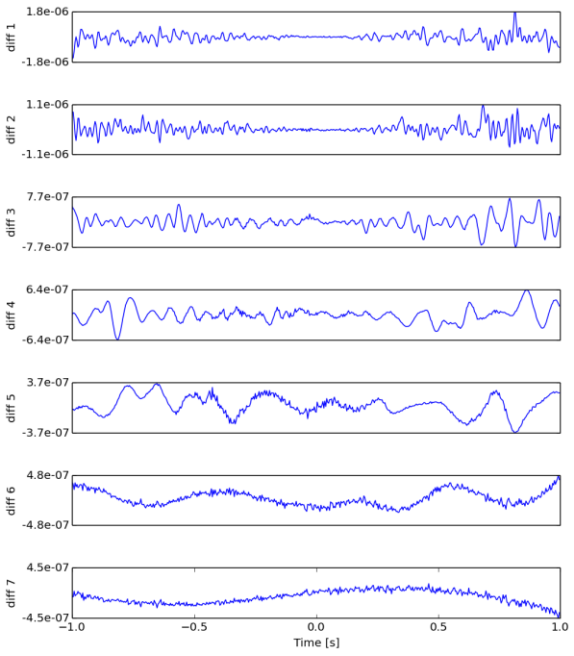

Fig. 9. Difference between SFP and DFP sets of EMD decompositions from example 3. SFP were first projected onto double precision and then subtracted from EMD DFP set.

V. CONCLUSION

As reported in section IV, there is a difference between decomposition obtained for different precision formats, namely single and double floating point precisions. These differences can be seen clearly in Figures 3, 6 and 9. It needs to be pointed out, that both absolute values and variance of error are small near $t = 0$ and increase when approaching $|t| = 1$. This is due to the fact that extrema positions are determined with parabolic interpolation, thus not necessarily falling onto the exact numerical representation grid. Such pronounced effect comes from the fact, that binary floating point representation has much bigger resolution close to zero and decreases with distance [12].

In summary, in all three experiments obtained differences are very small compared to the average amplitude of each component. Corresponding IMFs, produced in two different data formats, are visually indistinguishable. This means that using systems or devices, such as NVIDIA GPU [1], which perform faster on a single floating point compared to double floating point precision, one should be able to decrease computational time without a loss of meaningful content.

REFERENCES

[1] NVIDIA, NVIDIA CUDA Toolkit Release Notes, 2015 (accessed April 1, 2015). [Online]. Available: http://docs.nvidia.com/cuda.

[2] N. E. Huang et al., "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 454, no. 1971, pp. 903–995, Mar. 1998.

[3] N. E. Huang and Z. Wu, "A review on Hilbert-Huang transform: Method and its applications to geophysical studies," Reviews of Geophysics, vol. 46, no. 2, p. RG2006, Jun. 2008.

[4] G. Rilling and P. Flandrin, "On the Influence of Sampling on the Empirical Mode Decomposition", 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings.

[5] R. Rato, M. Ortigueira, and A. Batista, "On the HHT, its problems, and some solutions," Mechanical Systems and Signal Processing, vol. 22, no. 6, pp. 1374–1394, Aug. 2008.

[6] IEEE, "IEEE Standard for Floating-Point Arithmetic," pp. 1–70, 2008.

[7] E. Jones, T. Oliphant, P. Peterson et al., "SciPy: Open source scientific tools for Python," 2001–. [Online]. Available: http://www.scipy.org/

[8] D. Laszuk, "Python implemention of Empirical Mode Decomposition algorithm," 2014–. [Online]. Available: http://www.laszukdawid.com/codes

[9] K. Coughlin and K. Tung, "11-Year solar cycle in the stratosphere extracted by the empirical mode decomposition method," Advances in Space Research, vol. 34, no. 2, pp. 323–329, Jan. 2004.

[10] P. Flandrin and P. Gonc¸alv´es, "Empirical mode decompositions as data-driven wavelet-like expansions," International Journal of Wavelets Multiresolution and Information Processing, vol. 2, no. 4, pp. 1–20, 2004.

[11] N. Tsakalozos, K. Drakakis, and S. Rickard, "A formal study of the nonlinearity and consistency of the Empirical Mode Decomposition,"Signal Processing, vol. 92, no. 9, pp. 1961–1969, Sep. 2012.

[12] D. Goldberg, "What every computer scientist should know about floating point arithmetic," ACM Computing Surveys, vol. 23, no. 1, pp. 5–48, 1991.