

RESOLVING THIN BOUNDARY LAYERS IN NUMERICAL QUADRATURE[★]

GH. ADAM¹, S. ADAM², A. TIFREA, A. NEACSU

*Laboratory of Information Technologies
Joint Institute for Nuclear Research, Dubna, Russia
and*

*Department of Theoretical Physics
National Institute of Physics and Nuclear Engineering “Horia Hulubei”,
IFIN-HH, P. O. Box MG-6, R-76900 Bucharest-Măgurele, Romania*

(Received September 14, 2005)

Abstract. The identification of a library code able to solve numerically parametric integrals critically depends on the correctness of the diagnostic concerning the integrand behaviour at the integration domain boundaries. A second degree polynomial least squares fit of data over a minimal integrand sampling inside a mesoscopic boundary layer yields the necessary diagnostics provided solutions to the difficulties following from the lack of hardware and/or software conformity to the IEEE 754 standard. Relevant numerical examples let us infer that the developed code is robust, reliable, efficient, and provides precise outputs.

Key words: numerical integration, mesoscopic analysis, least squares fit, IEEE 754 standard, robustness, reliability, efficiency, output accuracy.

1. INTRODUCTION

A large number of physical models currently under study are characterized by two features: first, the observables are found as integrals which cannot be solved analytically, and, second, the properties of the models depend on one or more specific parameters the variation of which result in critical modification of their behaviour [1–5]. As a consequence, deep understanding of the models needs extensive numerical exploration of the dependence of the observables on the variable parameters.

The numerical solution of the parametric integrals makes use of existing library programs [6, 7, 8]. These programs have been tailored such as to solve

[★] Paper presented at the National Conference of Physics, 13–17 September 2005, Bucharest, Romania.

¹ Corresponding author. E-mail: adamg@jinr.ru, adamg@theory.nipne.ro

² E-mail: adams@jinr.ru, adams@theory.nipne.ro

effectively *specific* classes of integrands, leaving to the user the task of choosing the suitable library code. For the above mentioned models, however, in the presence of a phase transition (occurring at still to be determined values of the variable parameters) it is impossible to predict in advance which will be the right code. Since the trial and error approach has been found to result in a high rate of spurious outputs carrying the diagnostic of being correct, a study aimed at identifying the origins of the numerical codes failure is highly desirable.

It is worthwhile to get an approach to the solution which either results in the inference that a reliable output is to be expected, or finds at an early stage that the assumed quadrature algorithm will be useless. There is a hierarchy of the various sources of failure. We have found that the first point to be correctly solved is the diagnostic of integrand behaviour at the boundaries of the integration domain.

In the present paper we describe a method of analysis based on the use of a suitable integrand sampling inside a *mesoscopic neighbourhood of the boundary*. The principle of the method is described in Section 2. The discussion of the hardware and software influence on the reliability of the solution is done in Section 3. A few relevant examples are considered in Section 4. The paper ends with conclusions in Section 5.

2. MESOSCOPIC ANALYSIS OF BOUNDARY LAYER

Assume that $f : D \rightarrow \mathbb{R}$, $f = f(x)$, $D = [a, b] \subseteq \mathbb{R}$, is a continuous twice differentiable function and let $x_r \in D$ denote a reference argument value. Then there exists a nonvanishing neighbourhood $V(x_r) \subseteq D$ of x_r inside which a *linear* Taylor series expansion holds true within a predefined threshold $0 < \varepsilon \ll 1$.

If $x_r = a$ or $x_r = b$, then $V(x_r) \subseteq [a, b]$ is a *lateral* neighbourhood of x_r .

Numerical check of the continuity of $f(x)$ is done from a *sampling of its computed values*, $\{f_i = fl(f(x_i)) \mid i = 0, 1, \dots, m\}$, over a set of machine number arguments $S_m(x_r) = \{x_i \in V(x_r) \mid i = 0, 1, \dots, m\}$, $m \geq 3$, chosen such that $fl(x_r) \in S_m(x_r)$, where $fl(\zeta)$ denotes the floating point representation of $\zeta \in \mathbb{R}$. Let $\{f(x_i) \mid i = 0, 1, \dots, m\}$ denote the set of *actual* values of $f(x_i)$ over $S_m(x_r)$. In general, due to the round-off, $f(x_i) - fl(f(x_i)) \neq 0$, hence the best information on the smoothness properties of $f(x)$ at x_r following from the set $\{x_i, f_i\}$ is obtained from the scrutiny of the properties of a second degree polynomial least squares fit to the floating point data.

It is convenient to perform the *scale transformation*

$$x_i = x_0 + \xi_i h_r, \quad i = 0, 1, \dots, m, \quad \xi_i \in \mathbb{Z}, \quad (1)$$

where $x_0 \in S_m(x_r)$ may or may not coincide with the reference abscissa x_r , while h_r is the distance from x_r to its nearest machine number inside $[a, b]$. We are then left with the least squares fit problem to the data set $\{\xi_i, f_i\}$, $\xi_i \in \mathbb{Z}$.

The solution involves as standard steps the definition of the set of orthogonal polynomials $\{p_k(\xi_i) | k=0, 1, 2\}$ over the sampling $S_m(x_r)$ and the evaluation of the coefficients of the second degree fitting polynomial

$$y_2(\xi_i) = \alpha_0 + \alpha_1 p_1(\xi_i) + \alpha_2 p_2(\xi_i). \quad (2)$$

The basis polynomials spanning (2) obey the orthogonality relation

$$\sum_{i=0}^m p_k(\xi_i) p_l(\xi_i) = \delta_{kl} \sum_{i=0}^m p_k^2(\xi_i) \equiv N_k \delta_{kl}, \quad k, l \in \mathbb{N}. \quad (3)$$

The standard Gram-Schmidt orthogonalization procedure yields

$$p_0(\xi_i) = 1, \quad p_1(\xi_i) = \delta_i, \quad p_2(\xi_i) = \bar{\delta}^2(\delta_i^2 - \bar{\delta}^2) - \bar{\delta}^3 \delta_i, \quad (4)$$

where

$$\delta_i = \xi_i - \bar{\xi}, \quad \bar{\xi} = \frac{1}{m+1} \sum_{i=0}^m \xi_i, \quad \bar{\delta}^k = \frac{1}{m+1} \sum_{i=0}^m \delta_i^k, \quad k \in \mathbb{N}. \quad (5)$$

For the norms N_k we get

$$N_0 = m+1 \quad (6)$$

$$N_1 = N_0 \bar{\delta}^2 \quad (7)$$

$$N_2 = N_1 [\bar{\delta}^2 \bar{\sigma}^2 - (\bar{\delta}^3)^2], \quad \bar{\sigma}^2 = \frac{1}{m+1} \sum_{i=0}^m (\delta_i^2 - \bar{\delta}^2)^2. \quad (8)$$

This results in the fitting coefficients

$$\alpha_k = N_k^{-1} \sum_{i=0}^m p_k(\xi_i) f_i, \quad k=0, 1, 2. \quad (9)$$

Under negligible second degree coefficient α_2 , the first order derivative of $f(x)$ at x_r is given by

$$f'(x_r) \approx y_2'(\xi_r) = \alpha_1/h_0. \quad (10)$$

The smallest sampling $S_m(x_r)$ suitable for a least squares analysis providing insight on the smoothness properties of $f(x)$ at $x_r = a$ and $x_r = b$ respectively consists of four distinct abscissas (*i.e.*, $m=3$). We chose them such that the set $\{x_0, x_1, x_2\}$ defines a *uniform mesoscopic mesh*

$$\xi_0 = 0, \quad \xi_1 = p, \quad \xi_2 = 2p, \quad \xi_3 = q, \quad q \neq \{0, p, 2p\}. \quad (11)$$

Here, $p \gg 1$, $p\varepsilon_0 \ll 1$, where ε_0 is the largest relative spacing between adjacent machine numbers.

Then all the quantities entering the equations (4)–(10) can be calculated explicitly to yield

$$N_1\alpha_1 = \frac{1}{4}[(p-q)(d_{03} + d_{13} + d_{23}) + 4pd_{20}], \quad (12)$$

$$N_2\alpha_2 = \frac{p^2}{8} \left[q(p^2 - 6pq + 3q^2) \left(\frac{d_{30}}{q} - \frac{d_{20}}{2p} \right) + (2p^2 - 2pq + q^2)(f_0 - 2f_1 + f_2) \right], \quad (13)$$

where

$$d_{ij} = f_i - f_j, \quad i, j \in \{0, 1, 2, 3\}, \quad i \neq j, \quad (14)$$

and

$$N_1 = \frac{1}{4}(11p^2 - 6pq + 3q^2), \quad (15)$$

$$N_2 = \frac{N_1 p^2}{32}(4p^4 - 8p^3q + 15p^2q^2 - 12pq^3 + 3q^4). \quad (16)$$

Equation (13) shows that the validity of a linear Taylor expansion around the reference abscissa x_r holds true within a prescribed accuracy ε provided the minimal sampling (11) simultaneously fulfills two conditions:

- (i) The *scale invariance* of the first order derivative $f'(x)$ approximation at the q -length scale (d_{30}/q) and the $2p$ -length scale ($d_{20}/2p$) respectively.
- (ii) *Negligible curvature* ($f_0 - 2f_1 + f_2$) of $f(x)$ over the uniform p -length scale mesh $\{x_0, x_1, x_2\}$. [As $f_0 - 2f_1 + f_2 = 2p\{d_{20}/(2p) - d_{10}/p\}$, this is equivalent to the requirement of scale invariance of the first order derivative $f'(x)$ as approximated over the p -length scale (d_{10}/p) and $2p$ -length scale ($d_{20}/2p$) respectively.]

3. IMPLEMENTATION OF THE ANALYSIS CRITERIA

The data $\{\xi_i, f_i \mid i = 0, 1, 2, 3\}$ are generated over the mesoscopic distribution $S_3(x_r)$ the extension of which has remained until now undefined. In what follows, we assume that the width of $[a, b]$ is sufficiently large to accommodate any

mesoscopic p - and q -scale lengths. Then the partitions $S_3(a)$ and $S_3(b)$ are *distinct* of each other and the analysis follows *separately* over each of them.

THE IEEE 754 STANDARD

The possibility of doing a *reliable and portable analysis* follows from the conformity of both the *hardware* (RAM, cache, processor) and the *software* (operating system and compiler) with the *IEEE 754 standard* [9, 10] which governs binary floating point arithmetic (number formats, basic operations, conversions, and exceptional conditions).

Assuming that the specifications of this standard are fulfilled by the available computing systems (PCs with Intel 4, AMD 32, or AMD 64 processors, Linux 2.4 or 2.6 operating systems, GNU gcc compiler incorporating Fortran 77), we have developed an analysis procedure of the boundary layer along the lines described in the previous section.

INPUT: THE h_r STEP

In the previous section, the step h_r has been equated to the distance from x_r to the *nearest machine number inside* $[a, b]$. In floating point this assumes two *distinct* measures in terms of whether the quantity $|x_r| \varepsilon_0$ stays above or below the *underflow threshold* u . We define therefore the threshold

$$\tau_u = u/\varepsilon_0, \quad u = 2^{-1023}, \quad \varepsilon_0 = 2^{-53}, \quad (17)$$

by means of which we get the unique definition

$$|h_r| = \max\{|x_r|, \tau_u\} \varepsilon_0. \quad (18)$$

INPUT: THE p PARAMETER VALUE

For the practical implementation, we assume that the analysis is done in *double precision* (*i.e.*, the significand length is $52 + 1$ bits according to the IEEE 754 standard). A convenient starting value of the p parameter is

$$p = 2^{15}, \quad (19)$$

which secures, in principle, five significant figures at least for the first order derivative estimate, Eq. (10), of a smooth function. In fact, the choice of the p value is subject to two contradictory requirements:

- (i) If $f(x)$ is a *smooth* function, then it is desirable to have at our disposal a *larger* extension of the mesoscopic sampling than that following from Eq. (19).
- (ii) If $f(x)$ shows *irregular* behaviour over $S_3(x_r)$, then it is desirable to keep the starting value from Eq. (19) for p .

The need to *adapt the analysis step to the integrand behaviour* is dictated by the circumstance that, if the analysis neighbourhood is *too narrow*, then it is possible to get a *smooth function diagnostic* which, even if *correct in itself*, it can result in an *inefficient code*.

The only information we have at our disposal is that coming from the $\{\xi_i, f_i\}$ discretization. However, computing *all* the quantities asked by this discretization is useless if it proves necessary to increase the value of p . It is possible to *minimize* the number of integrand evaluations provided we start the analysis with two integrand values: *at the reference point* x_r (f_r) and at the *middle point* x_2 (f_2), respectively. If the difference $|f_r - f_2|$ stays under some threshold, then we decide to *increase* the analysis step (via that of the p value) such as to get the largest mesoscopic partition able to result in a correct decision of smooth function. The *confirmation* of the smooth function assumption provided by the *scale invariance test* of the divided differences computed at the two different values of the p parameter results in a sufficiently precise value of the first order derivative Eq. (10). The *disproof* of the smooth function assumption results in the correct diagnostic of irregular behaviour.

CHECKING SOFTWARE CONFORMITY WITH THE IEEE 754 STANDARD

The critical points which can falsify the result of the analysis are:

- (i) deviation of the length of the significand from the standard;
- (ii) *floating point comparison operation* deviations from the standard;
- (iii) unpredictable effects following from *compiler code optimization*.

The scrutiny of each of these points evidenced features contradicting the IEEE 754 standard and resulting in spurious diagnostics.

The assumption that the arithmetic operations are done using the *hidden bit* was invalidated by two observations. First, although the ξ_i values are *machine numbers* (hence unaffected by rounding errors), sometimes a *one bit loss* was detected at the end abscissa of the partition $S_3(x_r)$. Second, in the neighbourhood of a singular point $x_s \neq 0$, the computed values of $f(x)$ were found to be *identical* at assumably *distinct* machine numbers.

The absolute exponent value entering the expression of ε_0 in Eq. (17) defines the effective significand length used for the double precision RAM storage. On a superscalar computer using Intel or AMD processors (having floating point registers which work in *extended double precision*), the attempt to define the characteristic value m of this exponent from the condition that $fl(1. + 2.^{-m-1}) = 1.$ does not work directly due to the code optimization which does not return every computed value to the RAM but rather keeps it within a floating point register for reuse.

By writing down a code which forces the comparison of RAM/cache stored variables, it was possible to get the actual value of m , which is *different* from the IEEE 754 standard value quoted in Eq. (17). We have found that the actual value to be used on our systems is

$$\varepsilon_0 = 2^{-52}. \quad (20)$$

The *comparison of floating point variables* as done by the codes generated by the available compilers was found to infringe the IEEE 754 standard when a same numerical value is, on one side, transferred between different procedures (hence RAM stored) and, on the other side, is locally computed and kept into a floating point register. This showed that floating point data comparison is done in extended double precision. The harmful consequence of this comparison implementation was that singular integrand behaviour at abscissas which are different from the set of the machine numbers could not be resolved correctly. Right solutions have been implemented by forcing the floating point register fillings with RAM stored quantities at both sides of the comparison operand.

In fact, the wrong comparison terms come from inappropriate code optimization by the compiler. Our observation is not singular [11].

To get a solution insensitive to spurious compiler generated decisions, we have identified the sensitive places in the analysis code and we have replaced the straightforward decisions by suitable alternatives.

EXPECTED RESULTS

The analysis of the boundary layer yields, at each domain integration end, the following results:

- (1) The diagnostic concerning the behaviour of the analyzed function, returned as an integer value having the following meanings:
 - 1 – irregular behaviour inside the analysis neighbourhood;
 - 0 – smooth function;
 - 1 – inward monotonically decreasing function, pointing to the probable occurrence of a singularity in $f(x)$ or $f'(x)$, either at the analyzed domain end or at a nearby outer point, resulting in slow convergence;

- 2 – inward monotonically increasing function, pointing to the probable occurrence of an inner singularity in $f(x)$ or $f'(x)$, resulting in slow convergence.
- (2) Under a smooth function diagnostic, the analysis also returns an estimate of the first order derivative $f'(x_r)$, computed from the least squares formula (10).
 - (3) The number of function evaluations needed to infer the diagnostic associated to a boundary layer (that is, at both ends a and b of the analyzed interval) provides a measure to the efficiency of the proposed procedure.

ERROR ESTIMATE USING THE UNIFIED AFFINE MEASURE [12]

The modulus of the *absolute error* associated to the approximation of the first order derivative $f'(x_r)$ by means of Eq. (10) is given by

$$e_d = |f'(x_r) - y'_2(\xi_r)|. \quad (21)$$

This yields a good measure of the error estimate provided $|f'(x_r)| \leq 1$. If, however, $|f'(x_r)| > 1$, then a better measure to the approximation error is given by the modulus of the *relative error*,

$$\varepsilon_d = |[f'(x_r) - y'_2(\xi_r)] / f'(x_r)|. \quad (22)$$

At an arbitrary reference value $f'(x_r)$, the conventional characterization of the *precision* associated to the approximation $y'_2(\xi_r)$ is given by

$$\delta_d = \min\{e_d, \varepsilon_d\}. \quad (23)$$

A recent error analysis in an affine space [12] showed that the distance

$$d_V(v, w) = \ln \left(\frac{\sqrt{1+w^2} + w}{\sqrt{1+v^2} + v} \right). \quad (24)$$

from the ‘exact’ value v to the ‘approximate’ value w provides a unified measure of the error which tends towards the absolute error (21) at $|v| \ll 1$ and towards the relative error (22) at $|v| \gg 1$.

Under $|v - w| \ll 1$, Eq. (24) reduces to the simpler formula

$$d_V(v, w) = \frac{|v - w|}{\sqrt{1 + v^2}} \quad (25)$$

which intuitively illustrates the mentioned correspondences.

4. NUMERICAL RESULTS

GENERAL CHARACTERIZATION OF THE CASE STUDY OUTPUTS

The above analysis has been tested on several classes of parametric functions simulating various possible behaviours inside the boundary layer: continuous function, boundary singularity, outer singularity, inner singularity, finite jump, oscillatory function. The range of the parameters was varied over the whole machine range with respect to the function values such as to get a measure of the code robustness. The main conclusions of the analysis can be summarized as follows:

- (i) *Robustness of the procedure:* Correct outputs have been obtained for function ranges going from the underflow threshold to the overflow threshold.
- (ii) *Reliability of the procedure:* For all the investigated families of functions and all the considered sets of the parameters, the procedure resulted in truthful inferences.
- (iii) *Output precision of first order derivative estimates:* For all the families of smooth functions, Eq. (10) yielded at least five significant decimal figures in the sense of the affine error (24).
- (iv) *Efficiency of the procedure:* The number of function evaluations needed for diagnostic inference varied from the minimal value of four (under irregular behaviour, manifest endpoint or outer singularity, or smooth function at large arguments) to five (under smooth function analysis at argument magnitudes smaller than unity, inner singularities located near the discretization abscissa x_3 , as well as outer singularities undetectable over the standard input partition).

The details which deserve separated discussion concern the efficiency of derivative approximation by Eq. (10) in the case of continuous functions. Figs. 1 to 8 summarize the most interesting results.

In the case of polynomial functions, $f(x) = x^n + c$, $c \in \{0, 1\}$, the effect of the occurrence of a nonvanishing constant term was found to change significantly the round off error pattern at range end values which are not themselves machine numbers (Fig. 3 vs. Fig. 2 provide a striking illustration of this observation).

Figs. 2 and 3 illustrate the difference between the affine and conventional measures of the error associated to (10). The observed cusps come from the conventional measure cusp.

When the error measure mainly rests on the absolute error (Figs. 4 and 8), the relative error essentially remains constant, showing that the number of nonvanishing figures yielded by (10) mainly resides on the nature of the function under study.

Interesting periodicities related to powers of two are revealed by the log-log scale Figs. 6 and 7.

Fig. 5 illustrates a case where all the functions belonging to the considered family get the same value $f(0)=1$, while the derivative is $f'(0)=p$. Then it is mainly the relative error which controls the output precision, while the magnitude of the absolute error stays well below unity.

5. CONCLUSIONS

The search for effective implementation of numerical quadrature codes has lead to the identification of the boundary layer as a key problem the solution of which secures precision control of the numerical output.

The solution of this problem needed the development of the new concept of mesoscopic analysis, the straightforward implementation of which requires hardware/software conformity to the IEEE 754 standard.

The occurrence of significant deviations of the available software (GNU gcc compiler running under Linux OS) from this standard needed substantial reformulation of the algorithm such as to make its decisions insensitive to these deviations.

Numerical tests on relevant families of integrand functions result in the conclusion that the developed procedure is robust, reliable, precise, and efficient.

The figures were created using GNUPLOT Version 4.0.

Acknowledgments. The investigation has been supported partially from the CERES contract 3-16/2003 in Bucharest and partially from the JINR theme 09-6-1060-2005/2007 in Dubna.

REFERENCES

1. N. M. Plakida, L. Anton, S. Adam, and Gh. Adam, Exchange and spin-fluctuation superconducting pairing in cuprates, Preprint JINR, E-17-2001-59, Dubna, 2001; arXiv:cond-mat/0104234.
2. N. M. Plakida, L. Anton, S. Adam, and Gh. Adam, Exchange and spin-fluctuation pairing in the two-band Hubbard model. Application to cuprates, *In New Trends in Superconductivity*. J. F. Annett and S. Kruchinin, Eds. (Kluwer Academic Publ., New York, 2002) pp. 29–38.
3. N. M. Plakida, L. Anton, S. Adam, Gh. Adam, Exchange and spin-fluctuation mechanisms of superconductivity in cuprates (in Russian), *ZhETF* **124**, 367–378 (2003). English transl.: *JETP* **97**, 331–342 (2003).
4. P. A. Marchetti, Zhao-Bin Su, and Lu Yu, $U(1)\times SU(2)$ Chern-Simons gauge theory of underdoped cuprate superconductors, *Phys. Rev.* **B 58**, 5808–5824 (1998).
5. P. A. Marchetti, Jian-Hui Dai, Zhao-Bin Su, and Lu Yu, Gauge field theory of transport and magnetic relaxation in underdoped cuprates, *J. Phys.: Cond. Matter* **12**, L329–L336 (2000).
6. R. Piessens, E. deDoncker-Kapenga, C. W. Ueberhuber, and D. K. Kahaner, *QUADPACK*, a subroutine package for automatic integration, Springer, Berlin, 1983.
7. NAG Fortran Library, Mark15, FLH9715D, Hewlett-Packard 9000 Series 700 Double precision, DTP IFIN-HH, Bucharest.

-
8. A. R. Krommer and C. W. Ueberhuber, Computational integration, SIAM, Philadelphia, 1998.
 9. D. Goldberg, What every computer scientist should know about floating-point arithmetic, Computing Surveys (March, 1991), available at web site <http://www.validlab.com/goldberg/paper.pdf>
 10. W. Kahan, Lecture notes on the status of IEEE Standard 754 for binary floating-point arithmetic, May, 1996, available at web site <http://www.cs.berkeley.edu/wkahan/ieee754status/ieee754.ps>
 11. W. Kahan, How futile are mindless assessments of roundoff in floating-point computation (in progress) available at the web site <http://www.cs.berkeley.edu/wkahan/Mindless.pdf>
 12. J. Schiff, S. Shnider Lie groups and error analysis, Journal of Lie Theory, **11**, 231–254 (2000).

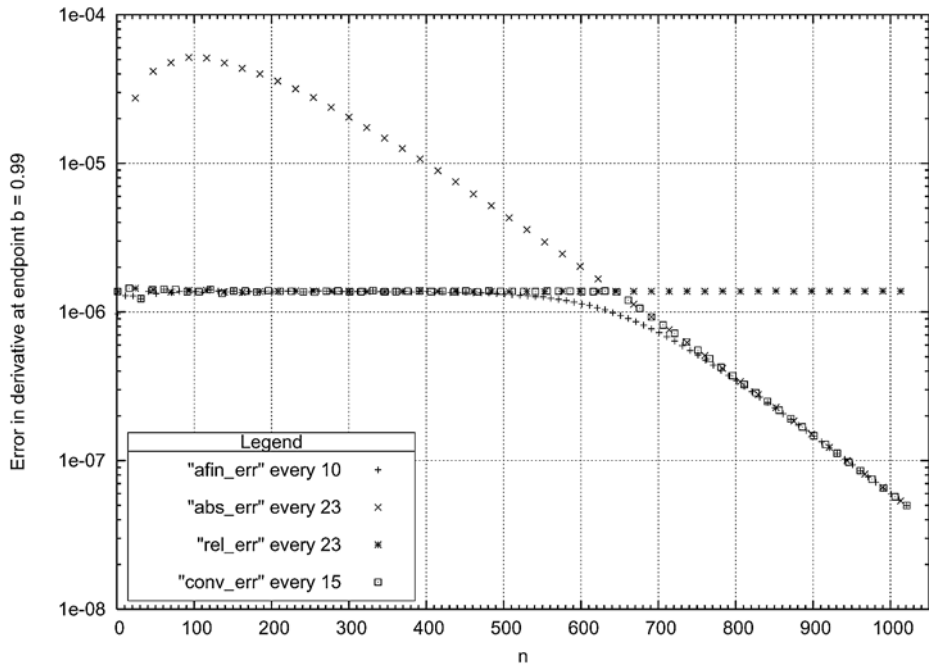


Fig. 1 – Error measures of the first order derivative approximation (10) of $f(x) = x^n$, at $b = 0.99$ inside $[a, b] = [0.01, 0.99]$, in terms of the exponent values n .

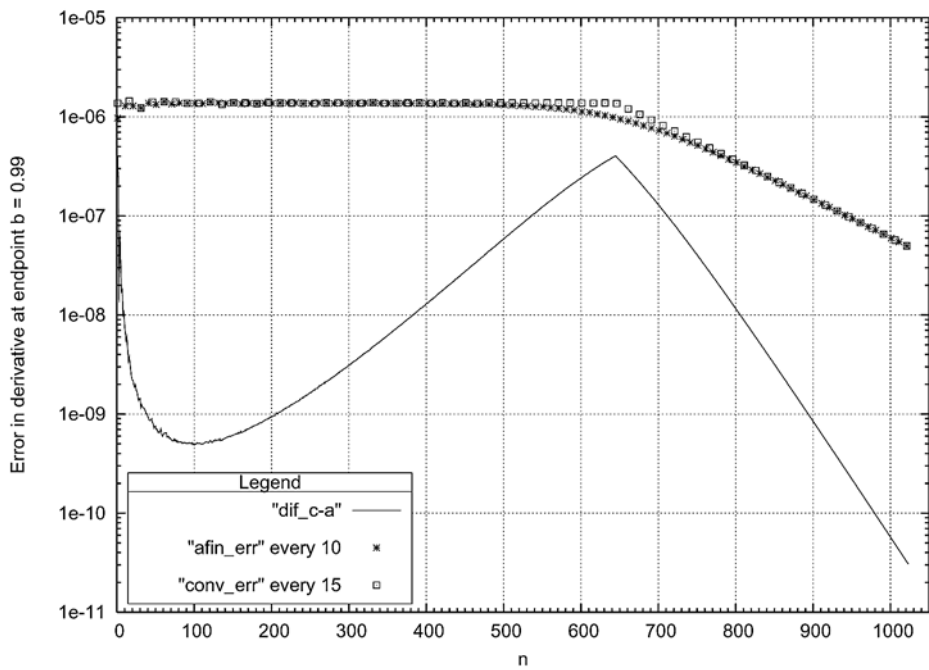


Fig. 2 – Comparison of the affine and conventional error measures for the data on the previous figure.

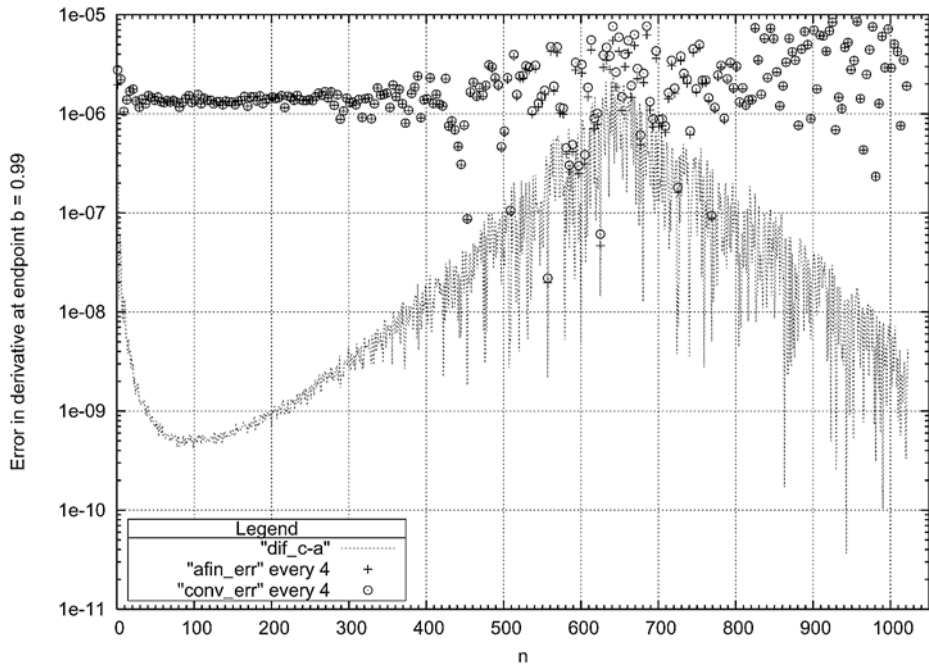


Fig. 3 – Same as Fig. 2, for the function $x^n + 1$.

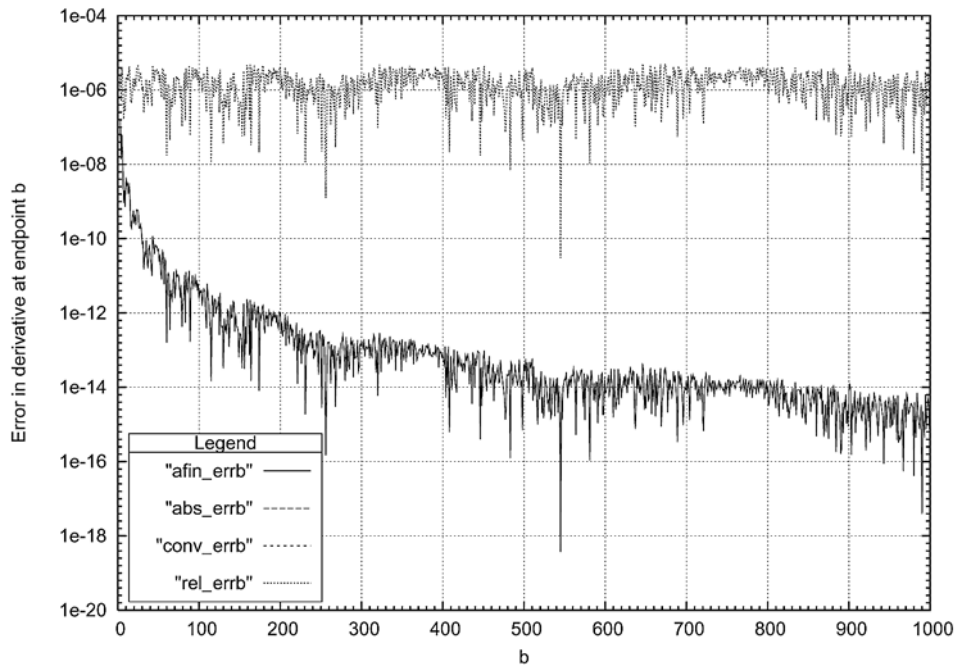


Fig. 4 – Error measures of the first order derivative approximation (10) of $f(x) = 1/(1+x^2)$, at b end inside $[0, b]$ for variable b . Within plot resolution, the affine, conventional and absolute error measures (lower graph) coincide with each other.

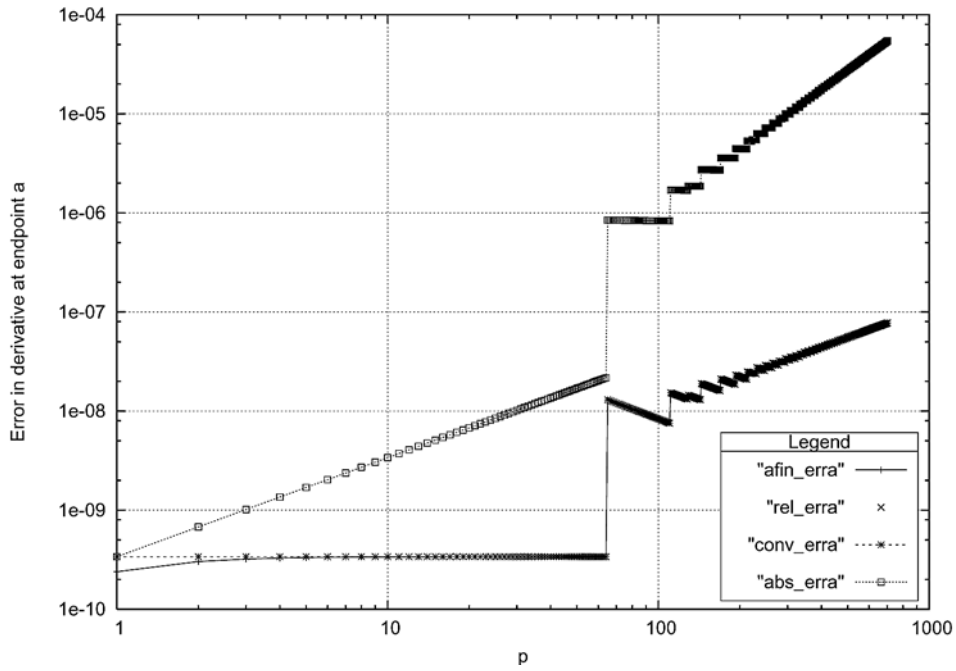


Fig. 5 – Error measures of the first order derivative approximation (10) of $f(x) = e^{px}$, at $a = 0$ inside $[a, b] = [0, 1]$ for variable p . The conventional and relative errors are identical to each other, while the affine one deviates from them at low p .

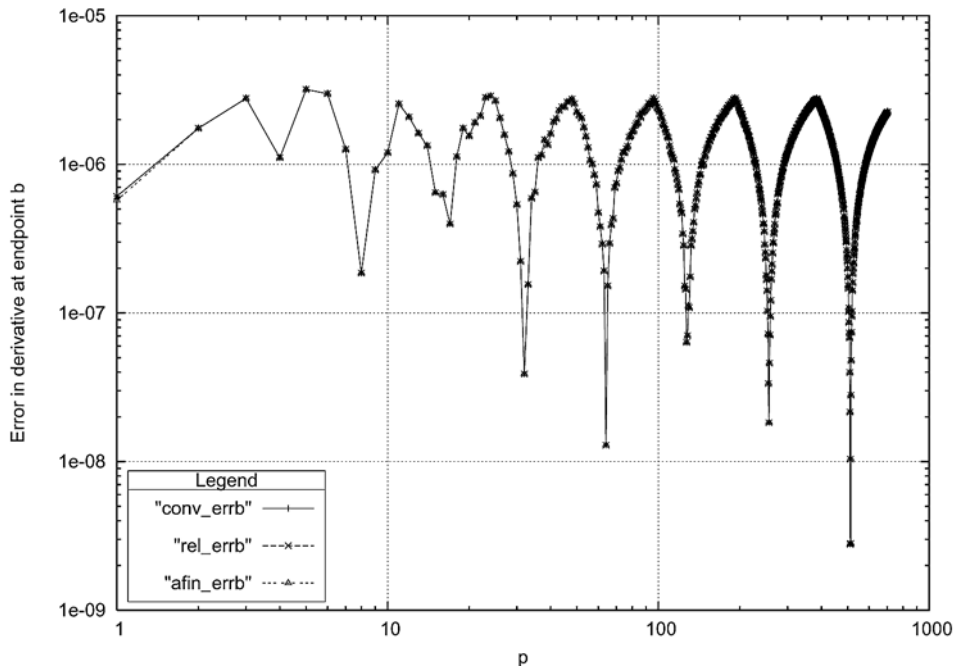


Fig. 6 – Same as previous figure, at $b = 1$ end. Here, the affine, conventional and relative errors are identical within plot resolution.

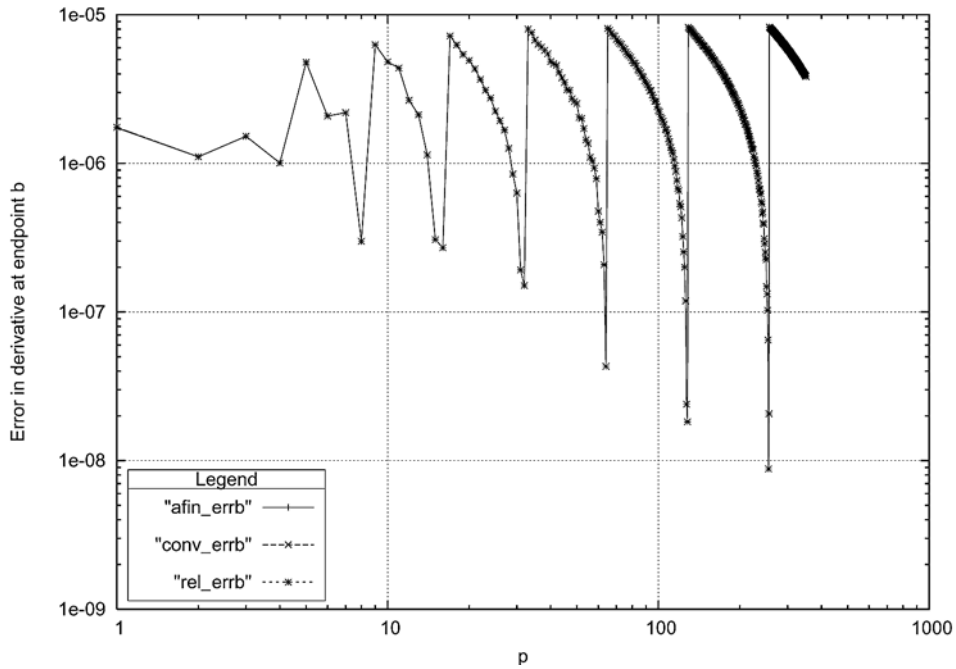


Fig. 7 – Same as previous figure, at $b = 1$ end for $f(x) = e^{p(x+1)}$. All mentioned errors are identical within plot resolution.

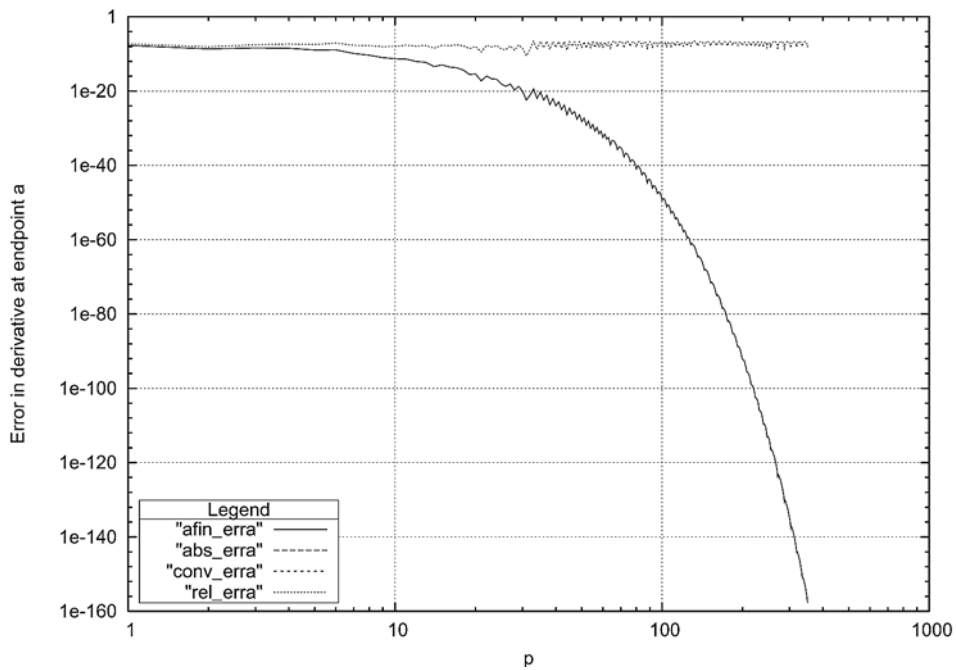


Fig. 8 – Same as previous figure, at $a = 0$ end for $f(x) = e^{p(x-1)}$. The precision is controlled by the absolute error, while the relative one mostly remains in the range $10^{-8} \div 10^{-7}$.