DEPARTMENT OF MATHEMATICS
TECHNICAL REPORT
ISSN: 1933-1746

Numerical Differentiation by Iterated Complex Perturbations

Jean-Paul Pemba

MDTRS No. 4

September 21, 2006

COLLEGE OF ARTS & SCIENCES

PRAIRIE VIEW A&M UNIVERSITY

PRAIRIE VIEW, TEXAS 77446-0519

Numerical Differentiation by Iterated Complex Perturbations

Jean-Paul Pemba*

Dept. of Mathematics, Prairie View A&M University, Prairie View, TX 77446-0519

Abstract. A method of approximation for arbitrary order derivatives is presented for smooth real variable functions. This method of successive complexifications is step-independent and avoids the classical Newtonian subtractive cancellation errors. Some numerical examples show how robust and easy to implement this technique is.

Key Words**:** iterated complexifications, complexification function, kth differential ε –complex perturbation**,** subtractive cancellations; finite differences.

AMS subject classifications: 65D25, 30E10, 65-04

1. Introduction

The classical numerical differentiation has been done by finite-differencing on the Newton quotient defining the first and higher derivatives; errors associated with this are due to cancellation triggering possible loss of significance, even for the first derivatives at very small step-sizes. The only way to try to reduce the influence of these errors is to increase the precision for a smaller round-off; this in turn is computationally expensive. Now, for the second and higher derivatives especially in Nonlinear Analysis, this approach is totally prohibitive. The Complex Variable method (CV) in Numerical Differentiation has been successfully carried-out since the 70s, but only for the first derivatives with absolute errors of order $1/n^2$ The Taylor expansion used to get this method working, could not deliver for the higher-order derivatives due to the [*]reappearance of subtractive terms. In this paper, we describe how to go around this

[*] Research Partially Supported by Grant AFORS#91NM062, Dept. of Mathematics, PVAMU
jepemba@pvamu.edu

hurdle and get the CV method work for arbitrarily-high-order derivatives. Here, the absolute errors of the higher derivatives remain in the order of $1/n^2$ for all odd derivatives and $1/n^4$ for even derivatives. Numerical examples show how robust this technique is.

2. Governing Approximations and Error Analysis

The first and second classical centered finite difference approximations (FDD) of an analytic (at least twice continuously differentiable) function f are written

$$f'(x) \approx \left[f(x+h) - f(x-h) \right]/2h \tag{1}$$

and

$$f''(x) \approx \left[f(x+2h) - 2f(x) + f(x-2h) \right]/4h^2 \tag{2}$$

both derived from the defining Newton difference quotient in differentiation .

Now, for $f \in C^n$ (n-times differentiable functions of a real variable with $f^{(n)}(x)$ continuous) and for every complex $z=x+yi$ in standard form, we introduce

$\tilde{f}(z) \equiv \sum_{k=0}^{n} i^k \frac{f^{(k)}(x)}{k!} y^k$ the *complexification function (at x, with size y) of order n* of *f*.

We note that $\tilde{f}(z)$ is well-defined by the Taylor series expansion theorem and that it arbitrarily approximates *f* for *n* large and *y* small enough. In [6], we study analytic properties of the linear operator $f \to \tilde{f}$ on functional spaces.

In the sequel, the following notations are used:

$f_1^y(x) = \frac{\operatorname{Im} \tilde{f}(z)}{y}$, $f_2^y(x) = \frac{\operatorname{Im} \tilde{f}_1}{y}$, and inductively $f_k^y(x) = \frac{\operatorname{Im} \tilde{f}_{k-1}(z)}{y}$ for $k = 2, 3, \ldots$

respectively the *first*, *second*, and the *kth differential y-complex perturbation of f at x.*

Theorem 1(Squire & Trapp,[7])

Let $f \in C^1$. *Fix x and* $\varepsilon > 0$ *small enough. Then,*

$$f'(x) \approx \operatorname{Im} f(x+i\varepsilon)/\varepsilon + O\left(\varepsilon^2 \left| f'''(x) \right|\right).$$

In [7], W. Squire and G. Trapp are solely interested in the numerical significance and illustrations of this result, not in its analysis. We propose to analyze and extend to higher-order numerical differentiation.

Theorem 2

Let $x_0 \in \mathbb{R}$ and $f \in C^n$ *(n=1, 2,…) be a function of a real-variable such that* $\tilde{f}(z)$ *is holomorphic in some locally convex neighborhood* D_{x_0} *of* $(x_0, 0)$ *in* $\mathbb{C}$.

Then, for all small enough $\varepsilon > 0$ *(so that* $(x_0+\varepsilon, 0) \in D_{x_0}$ *)*

$$f''(x) \approx \operatorname{Im} \tilde{f}_1(x+i\varepsilon)/\varepsilon + O\left(\varepsilon^4 \left| f^{(6)}(x) \right|\right)$$

Proof:

From the above definition of $\tilde{f}(z)$ where $z \in D_{x_0}$, we can write

$$f_1^{\varepsilon}(x) = \frac{\operatorname{Im} \tilde{f}(z)}{\varepsilon} = \sum_{k_0=0,\dots} (-1)^{k_0} \frac{\varepsilon^{2k_0}}{(2k_0+1)!} f^{(2k_0+1)}(x)$$ and then using the derivatives

$f_1^{(k_1)}(x)$ of $f_1^{\varepsilon}(x)$, we complexify again

$$\tilde{f}_1^{\varepsilon}(z) \equiv \sum_{k_1=0,\dots} \frac{(i\varepsilon)^{k_1}}{k_1!} f_1^{(k_1)}(x) = \sum_{k_1=0,\dots} \sum_{k_0=0,\dots} (-1)^{k_0} i^{k_1} \frac{\varepsilon^{2k_0+k_1}}{k_1!(2k_0+1)!} f^{(2k_0+k_1+1)}(x)$$

from which we get

$$\operatorname{Im} \tilde{f}_1^{\varepsilon}(z) = \varepsilon \sum_{k_1=0,\dots} \sum_{k_0=0,\dots} (-1)^{k_1} \frac{\varepsilon^{2k_0+2k_1}}{(2k_1+1)!(2k_0+1)!} f^{(2k_0+2k_1+2)}(x)$$

after splitting the k_1 summation -index into even and odd for the real and imaginary parts. But then, $\operatorname{Im} \tilde{f}_1^{\varepsilon}(z)/\varepsilon = \sum_{k_1=0,\ldots} \sum_{k_0=0,\ldots} (-1)^{k_1} \frac{\varepsilon^{2k_0+2k_1}}{(2k_1+1)!(2k_0+1)!} f^{(2k_0+2k_1+2)}(x)$

$$= f''(x) + \left[-\frac{1}{3!}+\frac{1}{3!}\right]\varepsilon^2 f^{(4)}(x) + \left[\frac{1}{5!}-\frac{1}{3!3!}+\frac{1}{5!}\right]\varepsilon^4 f^{(6)}(x) + \ldots$$

which by earlier notations represents the function $f_2^{\varepsilon}(x) \equiv \operatorname{Im} \tilde{f}_1^{\varepsilon}(z)/\varepsilon$. CQFD.

By inductively continuing the above process, it is tedious to generalize that

$$f_{2p}^{\varepsilon}(x) \equiv \operatorname{Im} \tilde{f}_{2p-1}^{\varepsilon}(z)/\varepsilon = \sum_{k_{2p},\ldots k_0=0,\ldots} (-1)^{k_1+k_3+\ldots+k_{2p-1}} \frac{\varepsilon^{2k_0+2k_1+\ldots+2k_{2p-1}}}{\prod_{j=0}^{2p-1}(2k_j+1)!} f^{(2k_0+\ldots+2k_{2p-1}+2p)}(x)$$

$$= f^{(2p)}(x) + \left[\frac{1}{3!}-\frac{1}{3!}+\frac{1}{3!}-\frac{1}{3!}+\ldots+\frac{1}{3!}-\frac{1}{3!}\right]\varepsilon^2 f^{(2p+2)}(x) + [-p/90]\varepsilon^4 f^{(2p+4)}(x) + \ldots$$

An effective way to sift the expansive coefficients one at a time, is to inductively regroup all indices of total sum $k_0 + k_1 + \ldots + k_{2p-1}$ = 0 (the trivial case that produces $f^{(2p)}(x)$),

then = 1 [eg.:(1,0,…,0), (0,1,…,0),…,(0,…,0,1), all 2*p* cases],

then = 2 [eg.: (2,0,…,0),…,(0,…,0,2), (1,1,0,…,0), (1,0,1,0,…,0),…,

(0,…,0,1,0,…,0,1,0,…,0),…, (0,…,0,1,1)],

2*p* cases followed by 2*p*-1 + 2*p*-2 + …+2+1 rearrangements of the two 1's, where only the odd totals will be nontrivial. This leads to

= $f^{(2p)}(x)$ +

[$(-1)^{0+\ldots+0} \frac{\varepsilon^{2+0+\ldots+0}}{(3!)(1!)\ldots(1!)} f^{(2+0+\ldots+0+2p)}(x) + (-1)^{1+0+\ldots+0} \frac{\varepsilon^{0+2+0+\ldots+0}}{(1!)(3!)\ldots(1!)} f^{(0+2+0+\ldots+0+2p)}(x)$ +…+
(*p* such alternating pairs)]

+

$$[(-1)^2\frac{\varepsilon^4}{5!}+...+(-1)^2\frac{\varepsilon^4}{5!}(2p\text{ terms})$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}+...+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}+...-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}+...+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}+...-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}+...+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}$$

...

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}+...-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}$$

$$-\frac{\varepsilon^4}{3!3!}+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}+...+\frac{\varepsilon^4}{3!3!}-\frac{\varepsilon^4}{3!3!}]f^{(2p+4)}(x)$$

$$= f^{(2p)}(x) + 0 + \left(\frac{2p}{5!}-\frac{p}{3!3!}\right)\varepsilon^4 \;\; f^{(2p+4)}(x)$$

$= \; f^{(2p)}(x) \; -\frac{p}{90}\varepsilon^4 \; f^{(2p+4)}(x)$, since only the p odd addends do not cancel-out.

Similarly, we establish that

$$f^{\varepsilon}_{2p+1}(x) \equiv \mathrm{Im}\;\tilde{f}^{\varepsilon}_{2p}(z)/\varepsilon = \sum_{k_{2p+1},...k_0=0,...}(-1)^{k_0+k_2+...+k_{2p}}\frac{\varepsilon^{2k_0+2k_1+...+2k_{2p-1}}}{\prod_{j=0}^{2p}(2k_j+1)!}f^{(2k_0+...+2k_{2p}+2p+1)}(x)$$

$$= \; f^{(2p+1)}(x)+\left[-\frac{1}{3!}+\frac{1}{3!}-\frac{1}{3!}+\frac{1}{3!}-\frac{1}{3!}+...+\frac{1}{3!}-\frac{1}{3!}\right]\varepsilon^2 f^{(2p+3)}(x)+...$$

Hence, this is summed up as

Theorem3

Let $x_0 \in \mathbb{R}$ and $f \in C^n$ *(n=1, 2,…) be a function of a real-variable such that* $\tilde{f}(z)$ *is holomorphic in some locally convex neighborhood* D_{x_0} *of* $(x_0, 0)$ *in* $\mathbb{C}$.

Then, for all positive integers p ($2p+1 \le n$ *) and for all small enough* $\varepsilon > 0$

1. $$f^{(2p+1)}(x_0) - f^{\varepsilon}_{2p+1}(x_0) \approx \varepsilon^2 \bullet \left(\sum_{k_0+k_1 \ldots +k_{2p}=1} \frac{(-1)^{k_0+k_2 \ldots +k_{2p}}}{(2k_0+1)!\ldots(2k_{2p}+1)!} \right) \bullet f^{(2p+3)}(x_0)$$

where the only summation indices retained must add-up to 1; and $(x_0+\varepsilon, 0) \in D_{x_0}$

2. $$f^{(2p)}(x_0) - f^{\varepsilon}_{2p}(x_0) \approx \varepsilon^4 \bullet \left(\sum_{k_0+\ldots+k_{2p-1}=2} \frac{(-1)^{k_1+k_3 \ldots +k_{2p-1}}}{(2k_0+1)!\ldots(2k_{2p-1}+1)!} \right) \bullet f^{(2p+4)}(x_0)$$

where the only summation indices retained must add-up to 2 and $(x_0+\varepsilon, 0) \in D_{x_0}$.

If we now consider the constants

$$A = \sum_{k_0+k_1+\ldots+k_{2p} \ge 1} \frac{\varepsilon^{2(k_0+k_1+\ldots+k_{2p})-2}}{\prod_{j=0}^{2p}(2k_j+1)!} \text{ and } P = \sum_{k_0+k_1+\ldots+k_{2p} \ge 2} \frac{\varepsilon^{2(k_0+k_1+\ldots+k_{2p-1})-4}}{\prod_{j=0}^{2p-1}(2k_j+1)!},$$

that depend solely on ε and *p*, then, it is clear that

$\max\{A, P\} \le \sum_{k \ge 2} \frac{1}{k!} \le \sum_{k \ge 2} \frac{1}{2^k}$, a (unconditionally convergent) geometric series.

And thus, the following consequence follows naturally:

Corollary 4

Let $x_0 \in \mathbb{R}$ and $f \in C^n$ *(n=1, 2,…) be a function of a real-variable such that:*

(i) $\tilde{f}(z)$ *is holomorphic in some locally convex neighborhood* D_{x_0} *of* $(x_0, 0)$ *in* $\mathbb{C}$

(ii) $M = \underset{1 \le k \le n}{Sup} \left\| f^{(k)} \right\|_\infty < \infty$

Then, for all positive integers p ($2p+1 \le n$ *), and for all* $\varepsilon > 0$ *;* $(x_0 + \varepsilon, 0) \in D_{x_0}$ *, there exist two positive constants A and P such that*

1. $\left| f^{(2p+1)}(x_0) - f^{\varepsilon}_{2p+1}(x_0) \right| \le \varepsilon^2 AM$
2. $\left| f^{(2p)}(x_0) - f^{\varepsilon}_{2p}(x_0) \right| \le \varepsilon^4 PM$.

3. Numerical Implementations

The following tables present an illustration of the advantage of the complex method in both absolute as well as relative errors. The first test-function we use for fig1-3 $f = \frac{2}{\pi^{1/4}\sqrt{3}}\left(1 - x^2\right)e^{-x^2/2}$ is a mother-wavelet called the $L^2(\mathbb{R})$- normalized Sombrero (or the Mexican hat) which coincides with the second derivative of the Gaussian Density with $\left\| f^{(4)} \right\|_\infty = 13.00987606$ and is very popular in many applications such as vision analysis. In Fig.1, showing a comparative presentation of FDD vs CV numerical calculations of f'', we note how in just a few first steps, the use of the classical finite differences is totally prohibitive. The blue blur representing the absolute errors in CV-method is shown, in Fig.2 to be variable in space and with steps, outperforming FDD by as high as 10^7-FDD to just 10^{-8}-CV or equivalently, 1-FDD to 10^{-15}-CV absolute errors at step h=10^{-4}. Fig.5 gives the illustration of the dominance of the CV method over the FDD for the relative errors, with an additional twist that unlike the absolute values, the relative errors in the FDD remain explosively higher and higher constant within each step while the relative errors in the CV method remain near 0, step after step.

The Fig.3, 4, and 6 show the same better results of the CV method over the classical FDD when we used as test function $g\left(x\right) = e^x / \left(\sin x^3 + \cos x^3\right)$ with $\left\| g^{(4)} \right\|_\infty = 196.5408638$, already experimented in Lyness[3] and in Squire and Trapp[7] for the numerical first derivative, in FORTRAN. In this paper, all the graphics are from MATLAB on a Pentium 4 Hewlett-Packard PC, with 512MB of RAM and 3.21GHz CPU. Note the abuse of notation: throughout, when comparing FDD vs CV, the step *h* in the FDD is real, while pure imaginary in CV.

4. Conclusions

Using the complex variable perturbation to estimate the second (as well as higher-order) derivatives gives sharper approximates than with the classical Finite-Difference approach, due to their large cancellation errors and totally unmanageable jumps under smaller and smaller (real) steps. The CV method is far from being perfect; the sharpest estimate for the test-functions used peaked earlier enough and the program on Pentium 4 crashed by the millionth of the imaginary unit-step for Low virtual memory (672 megabites). We did not have access to more powerful workstations (UNIX, LINUX, …) and think that any present limitation of CV method when competing with itself was machine and software related.The ideal would be to get sharper estimates than 10^{-8} absolute errors under vertical perturbation $h = 10^{-4} i$ that we did not improve with Matlab(symbolic vs Fortran-compiling for example) and Pentium 4, at least for these two particular test-functions. But the relative good news is that the method is still totally (real) step-independent; the estimates are essentially constant along the real samples at arbitrarily small horizontal steps. Moreover, the particular iterated complexification applied here when passing from first to second derivative is primal and far from unique; in this case, as a function the first derivative is actually an $O(h^2)$- approximation of the true derivative. Improvements can be made here. A competitive technique such as Automatic Differentiation (AD) will get sharper estimates at even smaller steps, but at incomparably higher time and computational(programming) costs. In that respect, one of the key advantages of CV over even AD is its extreme simplicity and practicality for all scales computations of the gradient, Jacobian matrix, as well as the Hessian and Laplacian. Multi-dimensional are easy at least componentwise. In Pemba [5], a totally different approach is applied based on optimal sampling and interpolation using neural networks, to move into higher-order numerical differentiation with excellent results so far. In those two new investigations in numerical differentiations, the subtractive methods are outmatched and thus represent a real hope for the overall improvement of the topic with direct implications on the numerical solutions of DEs and PDEs.

References

[1] P. Brezillon, J.F. Staub, A.M. Perault-Staub, G. Milgraud, Numerical estimation of the first derivative: approximate evaluation of an optimal step, Comput. Math. Appl., 7(1981), pp. 333-347.
[2] J.N. Lyness, C.B.Moler, Numerical differentiation of analytic functions, SIAM J. Numer.Anal., 4 (1967), pp. 202-210.
[3] J.N. Lyness, Numerical algorithms based on the theory of complex variables, Proc. ACM 22nd Nat. Conf., Thompson Book Co., Washington D.C, 1967, pp.124-134.
[4] J.N. Lyness, G.Sande, Algorithm 413-ENTCAF and ENTCRE: Evaluation of normalized Taylor coefficients of an analytic function, Comm. ACM 14, 10(1971), pp. 669-675.
[5] J.P. Pemba, Computing 2nd derivatives with neural networks, In completion.
[6] J.P. Pemba, On the complexification operator, In preparation.
[7] W. Squire, G.Trapp, Using complex variables to estimate derivatives of real functions, SIAM Rev.40 1(1998), pp. 110-112.
[8] R.S. Stepleman, N.D. Winarsky, Adaptive numerical differentiation, Math.Comput.33, 148(1979), pp. 1257-1264.

Fig.1: FDD vs CV in f'' - absolute errors

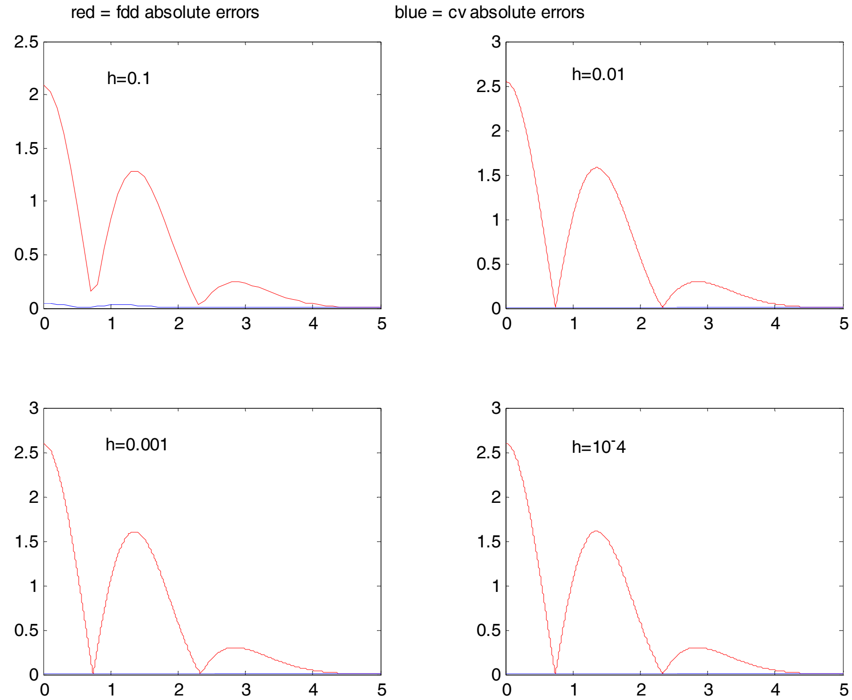
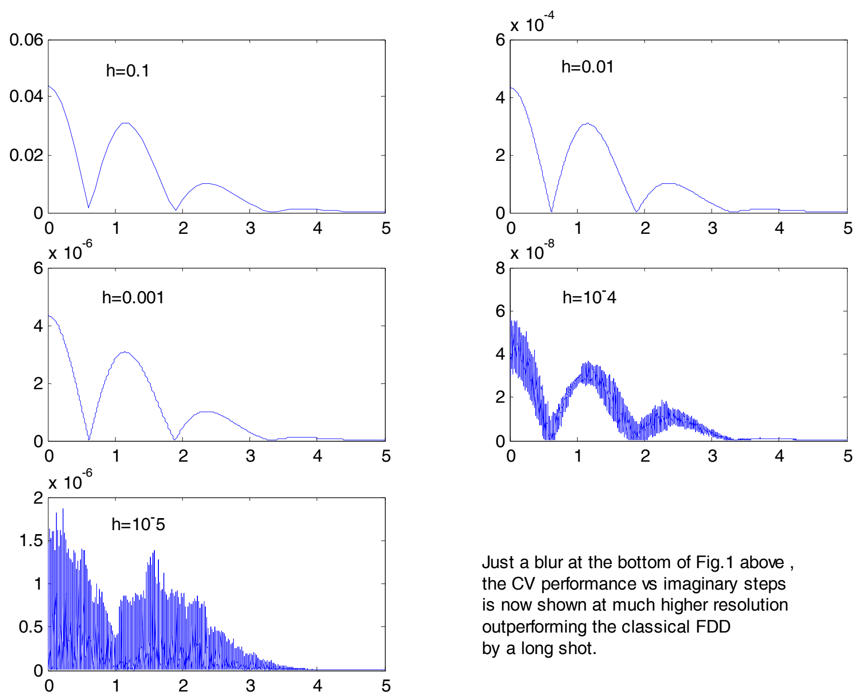


Fig.2: f'' - absolute errors in CV performance vs imaginary steps.



Just a blur at the bottom of Fig.1 above , the CV performance vs imaginary steps is now shown at much higher resolution outperforming the classical FDD by a long shot.

Fig.3: FDD vs CV in *g”* – absolute errors.

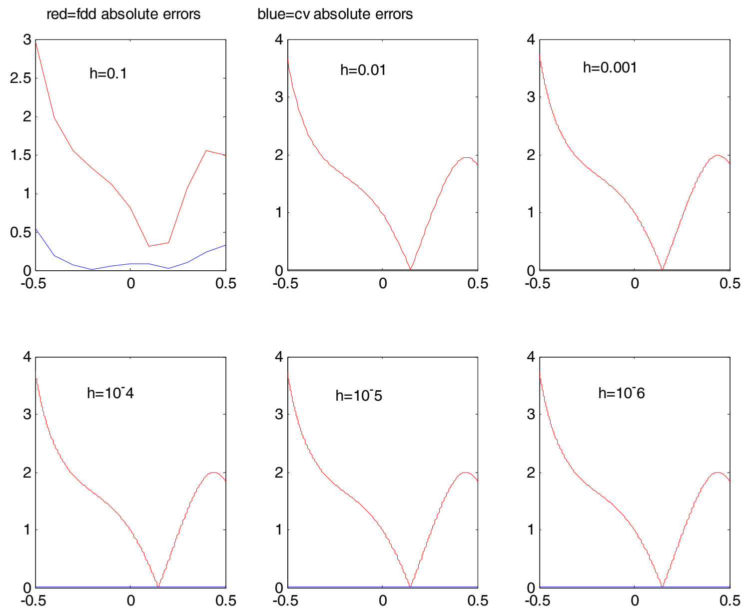
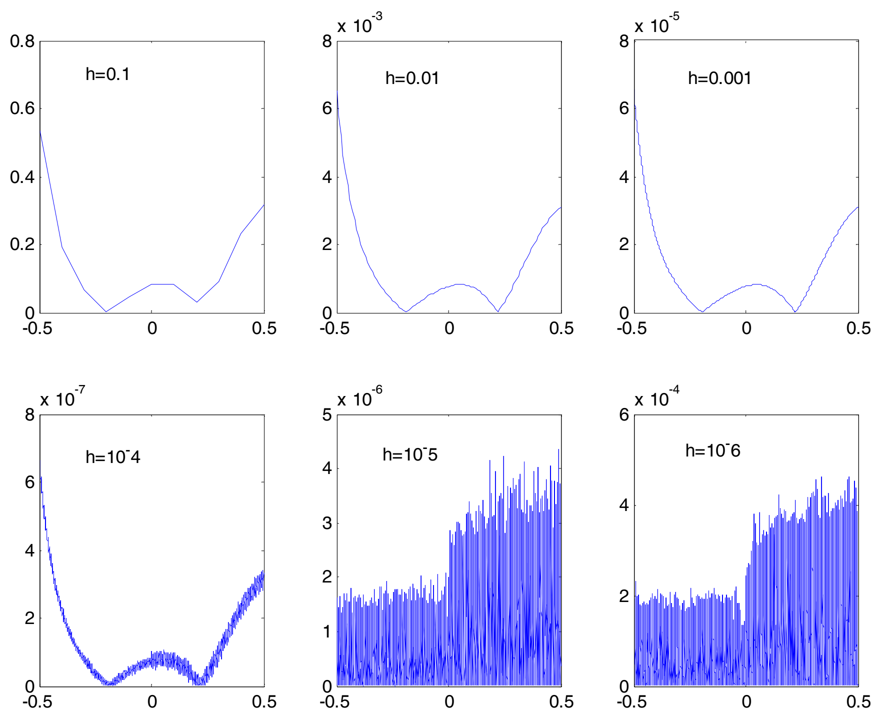


Fig.4: *g”*- absolute errors in CV performance vs imaginary steps.



g" - cv abs errors in much higher resolution from Fig.3 above.

Fig.5: FDD vs CV in f'' - relative errors.

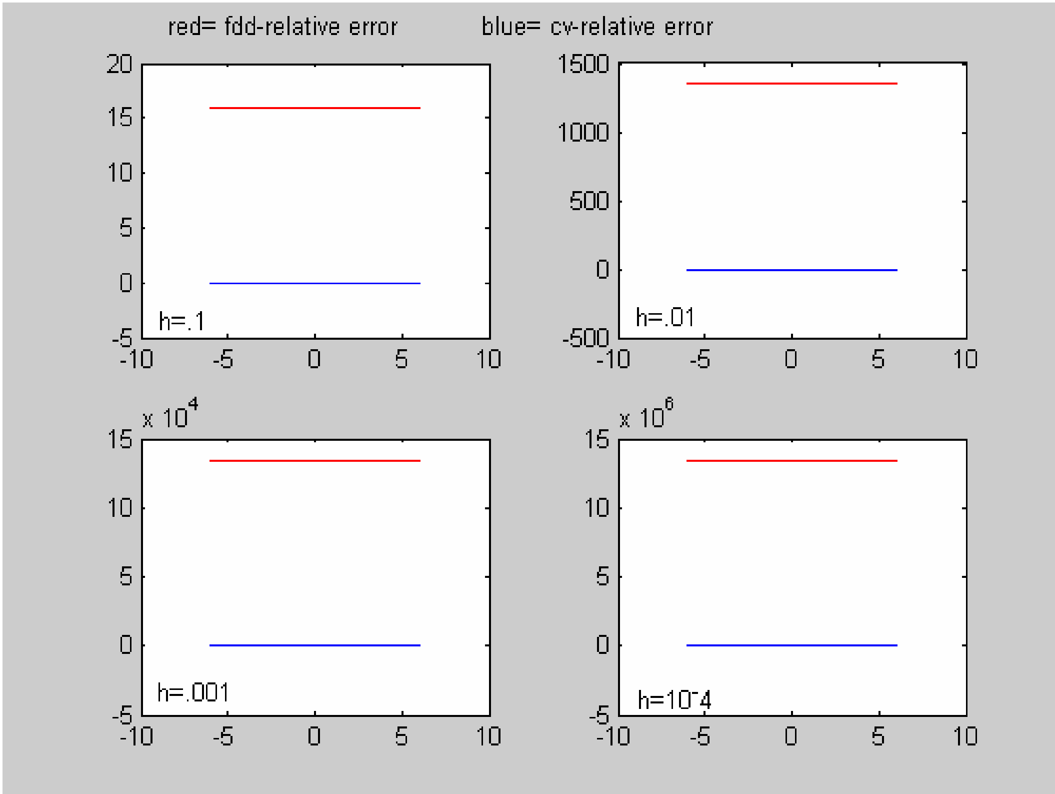


Fig.6: FDD vs CV in *g*” – relative errors.