Konuralp Journal of Mathematics
Volume 2 No. 2 pp. 109 –119 (2014) ©KJM

KOÇAK'S ACCELERATION METHOD SMOOTHLY GEARS UP ITERATIVE SOLVERS

MEHMET ÇETIN KOÇAK

Abstract. Consider a scalar repetitive scheme symbolically represented by $x_{k+1} = g(x_k)$ where k is the iteration count. Let z and n respectively denote the target fixed-point and convergence order of g. Koçak's method g_K accelerates g by actually solving a superior secondary solver obtained from a fixed-point preserving transformation

$g_K = x + G(g - x) = (g - mx)/(1 - m), m = 1 - 1/G, G = 1/(1 - m)$

where G is a gain and m is the slope of a straight line joining g and $g = x$ line. The method uses derivatives of g in local adjustment of m so as to push g_K towards the ideal solver $g = z$ by annihilating derivatives of g_K. If n is 1, then g_K is of third order. If n exceeds 1, then g_K is of $(n+1)$th order. Variable m improves remote behaviour also. The benefits of g_K amply compensate the cost of extra derivatives. A first-order solver with highly oscillatory divergence shows that the resultant third-order g_K renders a fast and smooth flight from a remote point to z. Newton's second-order, Chebyshev's third-order, and Ostrowski's fourth-order solvers all spin off in contrast

1. INTRODUCTION

A nonlinear equation

$$x = g(x) \tag{1.1}$$

can be solved by a repetitive scheme $x_{k+1} = g(x_k)$ where k is the iteration count. If z satisfies (1.1), then z is called a fixed-point of g. (*In this text, functions are usually written without an argument list when it contains x only.*) To find a z is to locate an intersection of the curve $g = g(x)$ with the straight line $g = x$. Each scheme starts form one or more points supposedly in the vicinity of z and usually ends when the absolute difference between successive iterates falls below a pre-specified tolerance.

Date: January 1, 2013 and, in revised form, February 2, 2013.

2010 *Mathematics Subject Classification.* 65B99, 65H05.

Key words and phrases. Non-linear equations; Iterative methods; Newton's method; Ostrowski's method; Convergence acceleration.

On a plot of x versus g, the iteration process follows a sequence of joined lines which are vertical and horizontal in turn; the first is a vertical line originating from the x-axis and ending on g, the second is a horizontal line extending to $g = x$, then comes a vertical line to g again, and so on. The "ideal" solver is the horizontal line g_{id} which needs just one trial from any starting point. Albeit, z is unavailable until the end! g_{id} can be harnessed however in post priori analysis, research, comparative studies, and troubleshooting.

Let $\varepsilon_k = x_k - z$. If there exist a real number n and nonzero constant c such that

$$\lim_{k\to+\infty} \frac{|\varepsilon_{k+1}|}{|\varepsilon_k|^n} = c,$$

then n and c are respectively called the convergence order and the asymptotic error constant. According to Traub [9], if n is integral, then

$$c = \lim_{k\to+\infty} \frac{\varepsilon_{k+1}}{\varepsilon_k^n} = \frac{g^{(n)}(z)}{n!}.$$

Linear or first order convergence ($n = 1$) means that $g\prime(z) \neq 0$. Quadratic convergence ($n = 2$) means that $g\prime(z) = 0$ but $g\prime\prime(z) \neq 0$. Generalizing, an integral convergence order $n > 1$ means that $g\prime(z) = g\prime\prime(z) = \ldots = g^{(n-1)}(z) = 0$ but $g^{(n)}(z) \neq 0$ and that ε_{k+1} is proportional to ε_k^n in the vicinity of z. (*Thus, nth order solvers are a subset of* $(n-1)$*th order solvers.*)

Many techniques reuse old information and/or harness new information at more than one point. In Traub's methodology [9], *methods with memory* are those which utilize past values and *multipoint solvers* are those which harness new information at a number of points. Furthermore, $g \in I_n$ indicates that g belongs to the class of solvers of order n. The condition for g to converge is that $|g\prime| < 1$ in the vicinity of z [8].

1.1. **A selection of iterative techniques.** In numerous cases, the iterative solver g actually comprises a long chain of equations involving many intermediate variables. If $x = g(x)$ is a rearrangement of another equation $f(x) = 0$, then the fixed-points of g are identical with the zeroes (roots) of f. Sometimes g comes from $g = x - fu$ where u is finite. Newton's popular second-order method [2,4]

$$g_N = f - \frac{f}{f\prime}$$

is an example of this type where u equals $1/f\prime$. (*A subscript starting with a capital letter is assigned in this paper to a solver g, convergence order n, and asymptotic error c so as to indicate the person to whom the pertinent method is attributed.*).

g_N is a piecewise linearization of f since it extends the current tangent to intersect the x-axis and suggests this value as the next approximation to z. (g_N is also called *the variable tangent method.*) As shown by Traub[9], $n_N = 2$ and $c_N = f\prime\prime(z)/(2f\prime(z))$ for simple roots. Repetition of z demotes convergence of g_N from *quadratic* to *superlinear* or *geometrical* and slows down the iteration process. If r is the multiplicity of z, then $g\prime(z) = (r-1)/r \neq 0$ and $g_{Nr} = x - rf/f\prime$ restores second order [4]. Direct differentiation of g_N gives $g'_N = ff\prime\prime/f'^2 = L$ where L is called the logarithmic degree of convexity. So, the convergence condition of g_N is that $|g'_N| = |L| < 1$ in the vicinity of z.

1.1.1. *Secondary solvers generated by partial substitution.* Partial substitution (g_{ps}) employs a variable gain G to amplify the correction to x, that is r

$$g_{ps} = x + G(g - x)$$

Applied to g_N, partial substitution gives $g_{Nps} = x - Gf/f\prime$. Note that g_{Nr} defined above is a g_{Nps} with a fixed gain $G = r$. Besides g_N, this article uses three g_{Nps} methods for comparison with newly geared-up g_K. They are Chebyshev's ($g_C \in I^3$), Halley's ($g_H \in I^3$) and Ostrowski's ($g_H \in I^4$) solvers whose respective formulas are as follows:

$$g_C = x - G\frac{f}{f\prime}, \quad G = 1 + \frac{L}{2},$$

$$g_H = x - G\frac{f}{f\prime} = x - f\frac{f\prime}{f\prime^2 - 0.5ff\prime\prime}, \quad G = \left(1 - \frac{L}{2}\right)^{-1},$$

$$g_O = x - G\frac{f}{f\prime}, \quad G = 1 + \frac{f(g_N)}{f - 2f(g_N)} = \frac{f - f(g_N)}{f - 2f(g_N)}.$$

1.1.2. *Secondary solvers generated by piecewise linearization.* Let K(x,g) be a point on g and let M(p,p) be an arbitrary point on $g = x$. The slope m of the straight line KM is given by $m = (g-p)/(x-p)$. Conversely, given the slope $m = (g-p)/(x-p) \neq 1$, a straight line through K intersects the line $g = x$ at M(p,p). Rearrangement renders $p = (g - mx)/(1 - m)$. A specified solver $g_{pl} = p$ may be regarded as a piecewise linearization of g if p is used to approximate z. Hence,

$$g_{pl} = \frac{g - mx}{1 - m}. \tag{1.2}$$

It is easy to show [5-7] that g_{pl} and g_{ps} are uniquely linked. Indeed,

$$g_{pl} = (g - mx)/(1 - m) \equiv x + G(g - x) \equiv g_{ps}$$

if $m = 1 - 1/G$ or $G = 1/(1 - m)$.

Regardless of the multiplicity of z, the ideal slope for linearization of g is $m_{id} = (g-z)/(x-z)$. If the chosen slope coincides with m_{id}, then $g_{pl} = g_{id} = z$ and M falls upon Z(z,z). Two well-known *one-point accelerators with memory*, namely Aitken's [1] and Wegstein's [3] methods, are examples of g_{pl}. Wegstein's approximating lines are secants of g going through a previous iterate (x_i,g_i) and the current iterate (x_k,g_k), that is

$$m = \frac{g_k - g_i}{x_k - x_i}, \quad g_W = \frac{g_k x_i - g_i x_k}{x_i - x_k - (g_i - g_k))}.$$

g_W is calculable first time at the end of the second iteration and is updated at each iteration afterwards. Aitken's technique similarly uses secants but its updates are every other iteration. Since $i = k - 1$ and $x_k = g_{k-1}$ here,

$$g_A = \frac{g_k x_{k-1} - g_{k-1}^2}{x_{k-1} - x_k - (g_{k-1} - g_k)} = g_k - \frac{(g_k - g_{k-1})^2}{x_{k-1} - x_k - (g_{k-1} - g_k)}.$$

2. KOÇAK'S ACCELERATOR

Koçak's method accelerates a given g by actually solving a superior secondary solver g_K generated through the transformation

$$g_K = x + G(g - x) = \frac{g - mx}{1 - m}, \quad m = 1 - \frac{1}{G}, \quad G = \frac{1}{1 - m}, \quad m \neq 1 \tag{2.1}$$

where G is a gain and m is a slope. Obviously, g_K is piecewise linearization and partial substitution, that is $g_K \equiv g_{pl} \equiv g_{ps}$. The transformation is the ultimate result of three successive operations on the equation $x = g(x)$, namely subtraction of the product mx from both sides, collecting terms, and rearrangement. Symbolically:

$$x - mx = g - mx \implies (1 - m)x = g - mx \implies x = g_K = (g - mx)/(1 - m)$$

The transformation obviously preserves fixed-points, that is $g_K(z) = g(z) = z$. Direct comparison of (2.1) and (1.2) shows that the varying slope is given by $m = (g - g_K)/(x - g_K)$.

Solver comparisons customarily focus on performance in the vicinity of z, paying attention to the number of function evaluations, the number of derivative calculations, convergence order, and asymptotic error constants. As previously published [5-7], if m is adjusted such that

$$m^{(i-1)}(z) = g^{(i)}(z)/i, \quad i = 1, 2, \ldots, n_K - 1, \quad n_K \geq 2, \quad m^{(0)}(z) = m(z) = g\prime(z),$$

then

$$g_K^{(i)}(z) = 0, \quad i = 1, 2, \ldots, n_K - 1$$

and g_K achieves an integral convergence order $n_K \geq 2$.

It is well known that, irrespective of its convergence order, an iterative method is liable to unsatisfactory performance (because of oscillation or slowness) and even to total failure if the starting point is not near enough the target z. The aim of the present phase was to improve remote behavior of g_K by successively zeroing as many of its derivatives ($g_K\prime$, $g_K\prime\prime$,...) as possible. The action forces g_K towards the ideal curve $g_{id} = z$. The case $g_K\prime = g_K\prime\prime = 0$ amply illustrates the approach.

Let $h = g_K - x$. Taylor's expansion of m around x is

$$m(g_K) = m(x + h) = m(x) + m\prime h + \frac{m\prime\prime h^2}{2!} + \frac{m\prime\prime\prime h^3}{3!} + \ldots$$

Truncating after the third term results in

$$m_h = m + m\prime h + \frac{m\prime\prime h^2}{2!} \backsimeq m(g_K).$$

In order to attain $g_K\prime = g_K\prime\prime = 0$, the following conditions must be satisfied:

$$m\prime = \frac{-(m - 1)(m - g\prime)}{g - x}, \tag{2.2}$$

$$m\prime\prime = \frac{2(1 - m)m\prime(m - 1 - g\prime) + m\prime(g - x)}{(1 - m)^2} - \frac{(g\prime\prime - 2m\prime)(1 - m)}{g - x}. \tag{2.3}$$

Suppose the set $\{g, g\prime, g\prime\prime\}$ is available at x. Then $\{g_K, h, m\prime, m\prime\prime\}$ and hence m_h depend on m alone. Recall that the minimal condition for $n_K \geq 2$ is that $\lim_{x\to z} m = g\prime(z)$. The problem now is to tune m so as to annihilate a discrepancy function $f_m(m) = m_h(m) - g\prime(z)$. The target is $m \backsimeq m_{id}$ which makes $g_K \backsimeq z$, $m_h \backsimeq m(g_K) \backsimeq m(z) = g\prime(z)$, and hence $f_m(m) \backsimeq 0$. It seems convenient to set up an inner loop to receive $\{g, g\prime, g\prime\prime\}$ at x, iteratively solve f_m for m (*subject to a tolerance*), and deliver the pertinent g_K to the outer loop as the new x value to test.

It is easy to see that $m = g\prime$, $m = g\prime(z)$, or a weighted average

$$m = wg\prime + (1 - w)g\prime(z) = g\prime(z) + w(g\prime - g\prime(z)) \tag{2.4}$$

fulfills the minimal requirement for $n_K \geq 2$ since $\lim_{x\to z} m = g\prime(z)$. The scheme can be modified in this case such that having received $\{g, g\prime, g\prime\prime, g\prime\prime\prime\}$ at x the inner loop solves a corresponding discrepancy function $f_w(w) = w_h(w) - w_{lim}$ for w by locally adjusting w subject to a specific end-point limit w_{lim} and return the pertinent g_K. The value of w_{lim} depends on n as explained later. The extra derivative $g\prime\prime\prime$ enters the scene because differentiating (2.4) with respect to x renders

$$w\prime = \frac{m\prime - wg\prime\prime}{g\prime - g\prime(z)}$$

$$w\prime\prime = \frac{m\prime\prime - (2w\prime g\prime\prime + wg\prime\prime\prime)}{g\prime - g\prime(z)}.$$

Equations (2.2) and (2.3) supply $m\prime$ and $m\prime\prime$ as before. From truncated Taylor's expansion again,

$$w_h = w + w\prime h + \frac{w\prime\prime h^2}{2!}.$$

Variable w is the distinguishing feature of the third version of g_K. The forerunner [5] employs constant $w = 1/2$ irrespective of n whereas the second [6] selects a constant w appropriate to n, that is it couples w to n. These constant w values which are coupled to n in the second version are now w_{lim} in the third version and $\lim_{x\to z} w = w_{lim}$. This means that both n_K and c_K remain unchanged in going from the second version to the third. Three different situations exist:

- **a):** If $n = 1$, then $w_{lim} = 1/2$ is used with the result that $n_K = 3$, $c_K = g'''_K(z)/3!$, and $g'''_K(z) = -0.5g\prime\prime\prime(z)/(1 - g\prime(z))$.
- **b):** If $n = 2$, then $w_{lim} = 1/2$ is employed with the result that $n_K = 3$, $c_K = g'''_K(z)/3!$, and $g'''_K(z) = -0.5g\prime\prime\prime(z)$.
- **c):** If $n > 2$, then $w_{lim} = 1/n$ is harnessed which renders $n_K = n + 1$, $c_K = g_K^{(n+1)}(z)/(n + 1)!$, and $g_K^{(n+1)}(z) = -g^{(n+1)}(z)/n$.

(Interesting results accrue [7] from the solution of $g'_K = 0$ assuming that $w\prime(g\prime - g\prime(z))(g - x) \backsimeq 0$. The new approach described above is free from this restricting assumption. There is more information in the appendix.)

2.1. **The algorithm.** The formulation hinged to w has been implemented. After many revisions and runs, the final version of the accelerator is now housed in a function that supervises the whole process once triggered by a call giving necessary initial information, namely, n, $g\prime(z)$, number of g derivatives, starting point, convergence tolerance, iteration limit, and name of the function to supply $\{g, g\prime, g\prime\prime,g\prime\prime\prime\}$at x. The supervisor function has two loops one inside the other. The outer loop is started after setting w_{lim} according to n. It sets $w = w_{lim}$, gets $\{g, g\prime, g\prime\prime,g\prime\prime\prime\}$from the named user function and begins the inner loop which employs g_N to solve $f_w(w) = 0$ for w thereby fixing the pertinent m and g_K. This g_K is then used as x for the next outer iteration.

The inner loop embeds an auxiliary function that takes $\{w, x,g, g\prime, g\prime\prime,g\prime\prime\prime\}$ and calculates $\{w_h, g_K\}$as follows. First, it obtains m from (2.4) and g_K from (2.1). It then accrues, $m\prime$,$w\prime$,$m\prime\prime$,$w\prime\prime$, and w_h. If the returned w_h exceeds 1.5 times w_{lim}, then the loop halves w and tries again. Otherwise, it checks $|f_w|$. If this is sufficiently small, then the current w and the resultant g_K are accepted and the inner loop is terminated. If not, then w is updated for the next iteration using $w = w - f_w(w)/f_w\prime(w)$ in accordance with g_N formulation. The derivative $f_w\prime(w)$ is calculated numerically which means an extra $f_w(w)$ per iteration here.

2.2. **Links to other solvers.** The accelerator naturally links to other solvers for it is both g_{ps} and g_{pl}. If $w = 1$, then $m = g\prime$ and the application [5] is equivalent to utilizing g_N to solve a secondary function $g - x = 0$. Indeed,

$$g_N = x - \frac{g - x}{g\prime - 1} = \frac{g\prime x - x + g - x}{g\prime - 1} = \frac{g - g\prime x}{1 - g\prime} = \frac{g - mx}{1 - m} = g_K, \quad m = g\prime$$

In this case, $n_K = n_N = 2$. Piecewise linearization techniques g_A and g_W are in fact a subclass of this case where the slope of a secant approximates $g\prime$. Alas, their popular implementations nullify possible beneficial contribution of $g\prime(z)$. Since secants virtually tend to the tangent as x goes to z, it can be asserted that $n_A = n_W = n_N = 2$ (provided that z is not repeated). The application of g_K converts [5] g_N to g_H if $w = 1/2$. With variable w tending to $w_{lim} = 1/2$, the result should be a *smoother* g_H.

2.3. **A highly oscillatory and divergent first-order test case.** This benchmark is in fact a member of a difficult class of problems keyed to N. Let $N = 7$ and $s = 10^N$. Suppose that $g = s/x^{N-1}$ is to be harnessed for the iterative solution of $f = x^N - s$. For this class, $g\prime = -(N - 1)s/x^N$, $g\prime(z) = -(N - 1)$, $g\prime\prime = -Ng\prime/x$, and $g\prime\prime\prime = N(N + 1)g\prime/x^2$. The new g_K with variable w will be applied to accelerate the process. The performance of g_K will be compared with those of $g_N \in I^2$, $g_C \in I^3$, $g_H \in I^3$, and $g_O \in I4$. (Remember that Chebyshev's, Halley's and Ostrowski's solvers are partial substitution variants of g_N.) The target fixed-point is $z = 10$, of course.

3. Results and discussion

Table 1 depicts the test results using $x_1 = 2$ as starting point. Note that $w_{h1} = w + w\prime h$ and $w_{h2} = w + w\prime h + w\prime\prime h^2/2!$. The use of $w_h = w_{h2}$ limits the extra information needed to $\{g\prime\prime,g\prime\prime\prime\}$. (As can be expected, w_{h2} is better to use than w_{h1}.) It is obvious that g_K with variable w superbly pilots the iteration process; the flight from a remote point to z is so fast and smooth despite the fact that g is a first-order solver with highly oscillatory divergence! Consider the first iteration now. New g_K accrues its largest correction here taking x from 2 to 7.950162903588. Notice that there are 4 internal iterations where w respectively takes the values

0.5, 0.25, 0.125, 0.056 compared with the ideal $w_{id} = 0.041652$. (w is halved twice before applying g_N.) In contrast, the solvers $\{g,\, g_N,\, g_C,\, g_O\}$ all spin off at $x = 2$. Only g_H takes a small step in the right direction. (*Note the previous two versions use $w = 1/2$ when $n = 1$ and with this class of problems this is equivalent to harnessing g_H.*) This proves the immense improvement of the third version over its predecessors. g_K continues to lead the other contestants in the second iteration, taking x to 10.003627135093 which is very close to the target. Not surprisingly, close to the finish $g_O \in I^4$ overtakes $g_K \in I^3$!

Without doubt, g_K with variable w amply compensates the extra cost of $\{g\prime\prime, g\prime\prime\prime\}$. In fact, its contribution is invaluable since it converts a divergent solver to a flyer. Needless to say, Koçak's acceleration method is also an important tool to analyze scalar iterative processes.

In summary, w reaches w_{lim} as solely determined by n. Presently, g_K needs $\{g, g\prime, g\prime\prime, g\prime\prime\prime\}$ and n. If preferred, $g_K\prime$ may be estimated numerically at the expense of an extra g per iteration [5]. Alternatively [5], g'_K may be replaced by the slope of a secant when $k \geq 2$. Note that this option envelops previously introduced piecewise linearization techniques with memory, namely g_A and g_W. The original formulations of these forego the beneficial contribution of $g\prime(z)$ since they harness $w = 1$. However, hinging w to n as described above should improve both of them. The requirement of $g\prime(z)$ is a handicap only when $n = 1$ for $g\prime(z) = 0$ when $n \geq 2$.

Table 1.g_K provides a fast and smooth flight to z from a point where others fail.

k	1	2	3	4
x	2.000000000000	7.950162903588	10.003627135093	10.000000001908
g	156250	39.604436076333	9.978264790570	9.999999988554
g_N	22323	12.472201928266	10.000003943018	10.000000000000
g_C	-747327804	4.755817803051	10.000000006190	10.000000000000
g_H	2.666646755895	9.621034770843	10.000000001908	10.000000000000
g_O	11162	10.431874480623	10.000000000002	10.000000000000
w_{id}	0.041652	0.353391	0.500242	0.000000
		Inner loop iterations		
1st w	0.500	0.500	0.500	0.500
w_{h1}	1.167	0.920	0.499	0.500
w_{h2}	1.833	1.341	0.499	0.500
g_K	2.666646755895	9.621034770843	**10.000000001908**	**10.000000000000**
2nd w	0.250	0.250		
w_{h1}	0.667	0.172		
w_{h2}	1.083	-0.269		
g_K	3.333253692267	10.390290953486		
3rd w	0.125	0.339		
w_{h1}	0.417	0.460		
w_{h2}	0.708	0.408		
g_K	4.666348122867	10.047938453298		
4th w	0.056	0.352		
w_{h1}	0.278	0.501		
w_{h2}	0.498	0.499		
g_K	**7.950162903588**	**10.003627135093**		

4. Conclusions

Old g_K needs g, $g\prime$, $g\prime(z)$ and n. The previous two versions use constant w throughout. The forerunner g_K sets w to $1/2$ for any g. The second version couples w to the convergence order n. If $n = 1$, then $w = 1/2$ is used and g_K is of third order. If $n > 1$, then $w = 1/n$ is harnessed and g_K is of $(n+1)$th order. The notion in the third version is to get additional higher derivatives $\{g\prime\prime, g\prime\prime\prime, ...\}$ at each step and deploy them within an embedded loop to fix $\{w, w\prime, w\prime\prime, ...\}$ at x such that $\{g_K\prime, g_K\prime\prime, ...\}$ are zero and the projected w at $x = g_K$ tends to its *(constant)* value in the second version. This action forces g_K towards the ideal solver $g = z$. The resultant g_K is of the same order as in the second version but the move to z is now a fast and smooth flight even from a remote starting point where other solvers fail.

A highly oscillatory and divergent first-order case demonstrated the super performance of $g_K$with variable w. The rewards of utilizing a variable w amply compensate the cost of the extra derivative information. It seems sufficient to get $\{g\prime\prime,$ $g\prime\prime\prime\}$ only, iteratively determine the appropriate set $\{w, w\prime, w\prime\prime\}$ at x which zeroes $g_K\prime$ and $g_K\prime\prime$ simultaneously and projects w to its expected limit at g_K.

Koçak's accelerator g_K has been upgraded with great success; it is now faster, smoother, and more robust. As implemented, g_K is a powerful one-point solver without memory. Ramifications are possible. For instance, just two iterations with g_K may be sufficient to reach a safe point from which other solvers may take over. If higher derivatives are difficult or expensive to calculate, then they may be replaced by finite difference formulae leading to one-point solvers with memory. Extension to multivariable solvers is another opening to investigate in the future. It is clear that Koçak's method provides a super tool for numerical analysis.

References

[1] Atkinson, K.E., *An introduction to numerical analysis*, John Wiley and Sons, New York, 1978.

[2] Fausette, L.V. Numerical methods: algorithms and applications, Prentice-Hall, New Jersey, 2003.

[3] Franks, R.G.E., *Modeling and simulation in chemical engineering*, John Wiley Interscience, New York, 1972.

[4] Fröberg, C-E., *Introduction to numerical analysis,* (Second ed.) Addison-Wesley Publishing Co., Reading, 1972.

[5] Koçak, M.Ç., *Simple geometry facilitates iterative solution of a nonlinear equation via a special transformation to accelerate convergence to third order,* The Proceedings of the Twelfth International Congress on Computational and Applied Mathematics (ICCAM2006), 10-14 July 2006, Leuven, Belgium. Goovaerts, M.J., Vandewalle, S., Van Daele, M., Wuytack, L. (eds) J. Comput. Appl. Math. Vol:218, (2008),350-363.

[6] Koçak, M.Ç., *Acceleration of iterative methods*, in Reports of the Third Congress of the World Mathematical Society of Turkic Countries, Almaty, June 30-July 4, 2009, Kazakhstan (B.T. Zhumagulov, Ed.), ISBN 978-601-240-063-2 (2009)

[7] Koçak, M.Ç., *Second derivative of an iterative solver boosts its acceleration by Koçak's method*, AMC, Vol: 218, No.3 (2011), 893-898.

[8] Quarteroni, A.F. , Sacco, R., Saleri, F., *Numerical mathematics,* Springer-Verlag, New York, 2000.

[9] Traub, J.F., *Iterative methods for solution of equations*, Prentice-Hall, Englewood Cliffs, NJ, 1964.

5. Appendix

Koçak's acceleration method relies on the fixed-point preserving transformation

$$g_K = g_{ps} = x + G(g - x) = (g - mx)/(1 - m) = g_{pl},\ m = 1 - 1/G,\ G = 1/(1 - m)$$

where G is a gain and m is the slope of a straight line joining g and $g = x$ line. Consider the link between $g_{id} = z$ and g_K . Regardless of the multiplicity of z [5], the ideal slope for linearization at the kth iteration is $m_{id} = (g_k - z)/(x_k - z) = \varepsilon_{k+1}/\varepsilon_k$ where $\varepsilon_k = x_k - z$. On the other hand, according to the mean value theorem for derivatives [5],

$$m_{id} = \frac{g - z}{x - z} = \frac{g - g(z)}{x - z} = g'(\xi), \quad \xi \in (x, z)$$

Note that this does not necessarily mean that $g'(\xi) \in (g'(x), g'(z))$.

As previously published [5-7], if m is adjusted such that

$$m^{(i-1)}(z) = g^{(i)}(z)/i,\ i = 1, 2, ...n_K{-}1\ , \quad n_K \geq 2, \qquad m^{(0)}(z) = m(z) = g'(z)\,,$$

then $g_K^{(i)}(z) = 0,\ \ i = 1, 2, ...n_K - 1$ and g_K achieves an integral convergence order $n_K \geq 2$. It is easy to see that a weighted average $m = wg' + (1 - w)g'(z) = g'(z) + w(g' - g'(z))$ fulfills the minimal requirement for $n_K \geq 2$, that is $\lim_{x \to z} m = g'(z)$. In this case, symbolic computations show that

$$m_{id} - m = \sum_i \left(\frac{1}{(i+1)!} - \frac{w}{i!} \right) g^{(i+1)}(z)\, \varepsilon_k^i. \tag{5.1}$$

Suppose that $n = 1$. The use of $w = 1/2$ makes $m = (g' + g'(z))$, annihilates the coefficient in the first summand here, and leads to the results

$$n_K = 3, \quad c_K = \frac{g_K'''(z)}{3!}, \quad g_K'''(z) = -0.5 \frac{g'''(z)}{1 - g'(z)}$$

The improvement is remarkable since the convergence order is raised from $n = 1$ to $n_K = 3$. Amelioration is even better if $g'(z) < 0$ or $g'(z) > 2$.

If $n = 2$, then $g'(z) = 0$ and so $w = 1/2$ renders

$$n_K = 3, \quad c_K = \frac{g_K'''(z)}{3!}, \quad g_K'''(z) = -0.5 g'''(z)\,.$$

If $n = 3$, then $g''(z) = 0$ and the first summand above is already zero and the second summand can be annihilated by choosing to employ $w = 1/3$ which makes $n_K = 4$. Similar reasoning leads to the deduction that if $n \geq 2$, then $w = 1/n$ makes $n_K = n + 1$. In fact, symbolic computations reveal [6] that if $n \geq 2$, then

$$g_K^{(i)}(z) = g^{(i)}(z)(1 - wi)\,, \quad i = 3, 4, ...$$

This equation not only corroborates that $w = 1/2$ gives a $g_K \in I_3$ when n was 1 or 2, but also shows if $w = 1/2$ is used when $n > 2$, then

$$n_K = n, \quad g_K^{(n)}(z) = (1 - \frac{n}{2})\, g^{(n)}(z)\,, \quad c_K = \frac{g_K^{(n)}(z)}{n!} = (1 - \frac{n}{2})\ c. \tag{5.2}$$

So, it is unwise to harness $w = 1/2$ when $n > 3$. If $w = 1/n$ instead of $w = 1/2$, then

$$n_K = n + 1, \quad c_K = \frac{g_K^{(n+1)}(z)}{(n+1)!}, \quad g_K^{(n+1)}(z) = -\frac{g^{(n+1)}(z)}{n!}. \tag{5.3}$$

Clearly, (5.3) is better than (5.2) when $n > 3$ since it increases n_K by 1 and greatly diminishes c_K.

The forerunner of g_K [5] employs constant $w = 1/2$ irrespective of n. The second version is identical with the first when n was 1 or 2. They differ when $n > 2$ since the second version begins to employ $w = 1/n$ (instead of $w = 1/2$) thereby improving both n_K and c_K. *Variable* w is the distinguishing feature of the third version of g_K. It locally tunes w so as to annihilate derivatives of g_K subject to the condition that w tend to its value in the second version as x approaches z. This means that the remote behavior is greatly improved but both n_K and c_K remain unchanged.

An ideal w, symbolized by w_{id}, annihilates the sum on the right-hand side of (5.1) and renders $m = m_{id}$. The definition of m_{id} directly leads to $w_{id} = (m_{id} - g'(z)/(g' - g'(z))$. Therefore, with the help of Hospital's rule [6],

$$w_{\lim} = \lim_{x \to z} w_{id} = \begin{cases} 1/2, & n = 1 \\ 1/n, & n > 1 \end{cases}$$

This fact can be coupled to a simple problem with a known z to estimate integer convergence orders if $n \geq 2$. In the neighborhood of z:

$$n = \frac{1}{w_{\lim}}, \quad w_{\lim} \approx \frac{m_{id} - g'(z)}{g' - g'(z)}, \quad m_{id} = \frac{g - z}{x - z}.$$

6. Abbreviations

c	Asymptotic error constant
c_K	Asymptotic error constant of Koçak's solver
f	Nonlinear function to be solved
f_w	Discrepancy function, $f_w = w_z - w_{\lim}$
g	Nonlinear solver
g_A	Aitken's solver (accelerator)
g_{id}	Ideal solver, $g_{id} = z$
g_C	Chebyshev's third-order solver
g_H	Halley's third-order solver
g_K	Koçak's solver
g_N	Newton's solver
g_{Nps}	Newton's solver with partial substitution
g_{Nr}	Newton's solver for repeated roots
g_O	Ostrowski's fourth-order solver
g_{pl}	Piecewise linearization
g_{ps}	Partial substitution
g_W	Wegstein's solver (accelerator)
k	iteration counter
L	Logarithmic degree of convexity, $L = ff''/f'^2$
m	Linearization slope
m_{id}	Ideal linearization slope
n	Convergence order
n_K	Convergence order
r	Multiplicity of z
x	(Default) independent variable
w	Weight for derivative
w_h	Weight projected to g_K
$w_{\lim}$	Limit for w at z
z	Fixed-point
ε_k	Eror at the kth iteration, $\varepsilon_k = x_k - z$

Chemical Engineering Department, Engineering Faculty, Ankara University, 06100 Tandoğan, Ankara, Turkey

E-mail address: `mckocak@ankara.edu.tr`