A tutorial on the Aitken convergence accelerator

By Cedrick Collomb[1]

Abstract. This tutorial introduces the concept and usefulness of convergence accelerators hopefully in a simple manner intelligible to any reader with minimal mathematical and engineering skills. The Aitken convergence accelerator is derived as an example with both intuitive explanations (every step is purposely made unnecessarily detailed for ease of understanding) and a simple demonstration. Source code is provided as a way to skip the potential formula obfuscation in order to help the reader implement and use the Aitken convergence accelerator immediately after reading this document.

1. Introduction

There exists in Mathematics and Computer Science a large number of iterative algorithms whose goal is usually to reach a solution to a problem within a certain tolerance within a given number of iterations. Iterating means going over a pattern of steps and procedures that can sometimes be complex and sometimes take a substantial amount of time even for fast modern computers. Examples of common iterative algorithms are root solvers, matrix inversion, linear equation solvers, and integrators. Each of these have deep implications in domains such as Engineering, in general; Computer Graphics; Video Games where runtime performance is critical; and also surprisingly in Internet search engines for features such as page rank [4]. What if there existed methods to reach the same result faster with less iteration? This is the purpose of convergence accelerators.

The rest of the tutorial is organized as follow: Section 2 introduces the Aitken formula graphically. Section 3 introduces the Aitken formulas analytically. Section 4 gives an example use of the Aitken formulas. Section 5 demonstrates that the Aitken formula accelerates convergence under given conditions. Section 6 introduces iterative Aitken formulas and Meyers iterated Aitken formulas. Section 7 extends Meyers formulas. Section 8 concludes this tutorial.

2. A graphical explanation of the Aitken δ^2 accelerator[2] formula

Imagine you have a convergent sequence $(x_n)_{n\in\mathbb{N}}$ iteratively defined by $x_{n+1} = f(x_n)$, and that this sequence converges to $\bar{x}$ with $f(\bar{x}) = \bar{x}$.

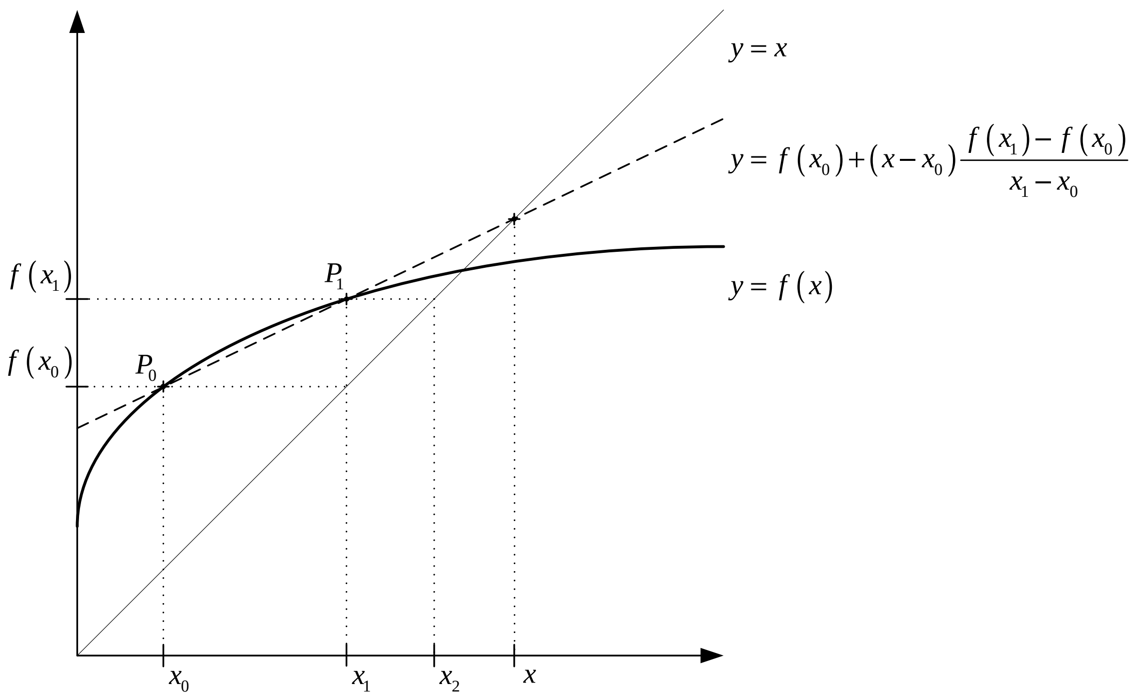
Given an initial start point x_0, you can construct the sequence $(x_n)_{n\in\mathbb{N}}$ graphically. Figure 1 shows how to proceed. Place x_0 on the x axis, find the point of the curve $y = f(x)$

[1] ccollomb@yahoo.com / http://ccollomb.free.fr/

[2] The Aitken accelerator being called δ^2 process is due to the fact that in (3), the numerator is a difference squared i.e. $(\Delta)^2$ and the denominator is a difference of a difference i.e. Δ^2 as in $\Delta(\Delta)$.

that is vertically aligned with x_0, the y value of this point is $f(x_0)$. Since by definition $x_1 = f(x_0)$ we now have found the value of x_1. In order to place x_1 graphically on the x axis, find the point of the curve $y = x$ that is horizontally aligned with $f(x_0)$ and project this point vertically on the x axis. Repeating the same procedure will give you x_2, then x_3, and you can continue to get as many terms of the sequence $(x_n)_{n\in\mathbb{N}}$ as needed.

Figure 1



Now let's focus on the first two points $P_0\Big|_{f(x_0)}^{x_0}$ and $P_1\Big|_{f(x_1)}^{x_1}$, and the line going through those two points as shown in Figure 1.

The equation of this line is $y = f(x_0) + (x - x_0)\dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$. Replacing $f(x_0)$ by x_1 and $f(x_1)$ by x_2 gives

$$y = x_1 + (x - x_0)\frac{x_2 - x_1}{x_1 - x_0} \tag{1}$$

The idea behind Aitken is to approximate $\overline{x}$ by the intersection of the line going through P_0 and P_1, with the line $y = x$. Finding the intersection is relatively simple and just require solving $x = x_1 + (x - x_0)\dfrac{x_2 - x_1}{x_1 - x_0}$, where the only unknown is x.

Regrouping the terms in x on the left side of the equal sign gives $x\left(1-\frac{x_2-x_1}{x_1-x_0}\right)=x_1-x_0\frac{x_2-x_1}{x_1-x_0}$ and after multiplying both sides of the equation by x_1-x_0 we find $x\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)=x_1\left(x_1-x_0\right)-x_0\left(x_2-x_1\right)$ therefore we have

$$x=\frac{x_1\left(x_1-x_0\right)-x_0\left(x_2-x_1\right)}{\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)} \tag{2}$$

Subtracting and adding $x_0\left(x_1-x_0\right)$ in the middle of the numerator of (2) gives $x=\frac{x_1\left(x_1-x_0\right)-x_0\left(x_1-x_0\right)+x_0\left(x_1-x_0\right)-x_0\left(x_2-x_1\right)}{\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)}$ which first two terms of the numerator can be factored by $\left(x_1-x_0\right)$ and which last two terms of the numerator can be factored by x_0 which gives $x=\frac{\left(x_1-x_0\right)^2+x_0\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)}{\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)}$.

Simplifying by $\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)$ gives $x=\frac{\left(x_1-x_0\right)^2}{\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)}+x_0$. Therefore

$$x=x_0+\frac{\left(x_1-x_0\right)^2}{\left(\left(x_1-x_0\right)-\left(x_2-x_1\right)\right)} \tag{3}$$

This is the Aitken formula to find an approximate x of $\overline{x}$ given three consecutive terms of a sequence.

3. An analytical explanation of the Aitken formulas

As for the previous section you have a convergent sequence $\left(x_n\right)_{n\in\mathbb{N}}$ iteratively defined by $x_{n+1}=f\left(x_n\right)$, and that this sequence converges to $\overline{x}$ with $f\left(\overline{x}\right)=\overline{x}$. Now let's admit that $f\left(x\right)$ has a Taylor development at $\overline{x}$ of order 2.

$$f\left(\overline{x}+h\right)=f\left(\overline{x}\right)+hf'\left(\overline{x}\right)+O\left(h^2\right) \tag{4}$$

We define $x=\overline{x}+h$, therefore $h=\overline{x}-x$. Let $\alpha=f'\left(\overline{x}\right)$. Replacing x by $\overline{x}+h$, h by $x-\overline{x}$, $f\left(\overline{x}\right)$ by $\overline{x}$, and $f'\left(\overline{x}\right)$ by α in (4), as well as neglecting the order two error $O\left(\left(x-\overline{x}\right)^2\right)$ gives

$$f\left(x\right)=\overline{x}+\alpha\left(x-\overline{x}\right) \tag{5}$$

Note that we do not know $\overline{x}$ and also generally do not know α in (5). Applying the previous formula to two continuous elements of the sequence x_i and x_{i+1}, gives two equations with two unknowns $\overline{x}$ and α.

$$\begin{aligned} x_{i+1} &= f\left(x_i\right) = \overline{x} + \alpha\left(x_i - \overline{x}\right) \\ x_{i+1} &= \left(1-\alpha\right)\overline{x} + \alpha x_i \end{aligned} \tag{6}$$

$$\begin{aligned} x_{i+2} &= f\left(x_{i+1}\right) = \overline{x} + \alpha\left(x_{i+1} - \overline{x}\right) \\ x_{i+2} &= \left(1-\alpha\right)\overline{x} + \alpha x_{i+1} \end{aligned} \tag{7}$$

Subtracting (6) from (7) gives $x_{i+2} - x_{i+1} = \alpha\left(x_{i+1} - x_i\right)$. Therefore gives us α.

$$\alpha = \frac{x_{i+2} - x_{i+1}}{x_{i+1} - x_i} \tag{8}$$

We use (8) to calculate $\frac{1}{1-\alpha}$ and $\frac{\alpha}{1-\alpha}$ that will come handy very soon.

$$1-\alpha = 1 - \frac{x_{i+2} - x_{i+1}}{x_{i+1} - x_i} = \frac{\left(x_{i+1} - x_i\right) - \left(x_{i+2} - x_{i+1}\right)}{x_{i+1} - x_i}$$

$$\text{and } \frac{1}{1-\alpha} = \frac{x_{i+1} - x_i}{\left(x_{i+1} - x_i\right) - \left(x_{i+2} - x_{i+1}\right)} \tag{9}$$

$$\text{Combining (8) and (9) gives } \frac{\alpha}{1-\alpha} = \frac{x_{i+2} - x_{i+1}}{\left(x_{i+1} - x_i\right) - \left(x_{i+2} - x_{i+1}\right)} \tag{10}$$

We use (6) and (9) to find a first formula for $\overline{x}$.

$$\text{Using (6) and solving for } \overline{x} \text{ gives } \overline{x} = \frac{x_{i+1} - \alpha x_i}{1-\alpha} \tag{11}$$

$$\text{Subtracting and adding } x_i \text{ gives } \overline{x} = \frac{x_{i+1} - x_i + x_i - \alpha x_i}{1-\alpha}$$

$$\text{Factoring the right side of the numerator gives } \overline{x} = \frac{x_{i+1} - x_i + x_i(1-\alpha)}{1-\alpha}$$

$$\text{Simplifying gives } \overline{x} = \frac{x_{i+1} - x_i}{1-\alpha} + x_i$$

$$\text{Therefore } \overline{x} = x_i + \frac{x_{i+1} - x_i}{1-\alpha} \text{ and using (9) gives}$$

$$\overline{x} = x_i + \frac{\left(x_{i+1} - x_i\right)^2}{\left(x_{i+1} - x_i\right) - \left(x_{i+2} - x_{i+1}\right)} \tag{12}$$

Using (11) in a different way leads to the second expression for $\bar{x}$.

Subtracting and adding αx_{i+1} in (11) gives $\bar{x} = \dfrac{x_{i+1} - \alpha x_{i+1} + \alpha x_{i+1} - \alpha x_i}{1-\alpha}$

Factoring the numerator gives $\bar{x} = \dfrac{(1-\alpha) x_{i+1} + \alpha (x_{i+1} - x_i)}{1-\alpha}$

Simplifying gives $\bar{x} = x_{i+1} + \dfrac{\alpha (x_{i+1} - x_i)}{1-\alpha}$, and using (10) gives

$$\bar{x} = x_{i+1} + \frac{(x_{i+2} - x_{i+1})(x_{i+1} - x_i)}{(x_{i+1} - x_i) - (x_{i+2} - x_{i+1})} \tag{13}$$

The third expression for $\bar{x}$ comes from (7).

Using (7) and solving for $\bar{x}$ gives $\bar{x} = \dfrac{x_{i+2} - \alpha x_{i+1}}{1-\alpha}$

Subtracting and adding αx_{i+2} gives $\bar{x} = \dfrac{x_{i+2} - \alpha x_{i+2} + \alpha x_{i+2} - \alpha x_{i+1}}{1-\alpha}$

Factoring the numerator gives $\bar{x} = \dfrac{(1-\alpha) x_{i+2} + \alpha (x_{i+2} - x_{i+1})}{1-\alpha}$

Simplifying gives $\bar{x} = x_{i+2} + \dfrac{\alpha (x_{i+2} - x_{i+1})}{1-\alpha}$

Using (10) gives the most common formulation of the Aitken accelerator.

$$\bar{x} = x_{i+2} + \frac{(x_{i+2} - x_{i+1})^2}{(x_{i+1} - x_i) - (x_{i+2} - x_{i+1})} \tag{14}$$

To resume there are three equivalent ways to express $\bar{x}$, equations (13) and (14) could have been directly expressed from (12) but it was worth deriving from the original equations for pedagogic reasons. The high level item that is very important to understand is that the Aitken accelerator gives an approximate of the solution of your problem ($\bar{x}$) based on three consecutive terms of a convergent sequence.

(12) $\bar{x} = x_i + \dfrac{(x_{i+1} - x_i)^2}{(x_{i+1} - x_i) - (x_{i+2} - x_{i+1})}$

(13) $\bar{x} = x_{i+1} + \dfrac{(x_{i+2} - x_{i+1})(x_{i+1} - x_i)}{(x_{i+1} - x_i) - (x_{i+2} - x_{i+1})}$

(14) $\bar{x} = x_{i+2} + \dfrac{(x_{i+2} - x_{i+1})^2}{(x_{i+1} - x_i) - (x_{i+2} - x_{i+1})}$

4. Example of use of the Aitken accelerator

The Newton method is a well known method to find roots of the equation $g(x)=0$. The iterative procedure is given by $f(x)=x-\dfrac{g(x)}{g'(x)}$. An extension of the Newton method described in [8] is the Steffensen method that uses for every three iterations two iterations of the Newton method *f* and the Aitken formula (14) on the last three elements of the sequence for the last iteration.

5. A simple demonstration

Definition: $(T_n)_{n\in\mathbb{N}}$ is an accelerating sequence[3] of $(S_n)_{n\in\mathbb{N}}$ if the following properties are true: $(T_n)_{n\in\mathbb{N}}$ converges to the same limit S as $(S_n)_{n\in\mathbb{N}}$, $(T_n)_{n\in\mathbb{N}}$ converges faster than $(S_n)_{n\in\mathbb{N}}$ with the following criterion $\lim\limits_{n\to\infty}\dfrac{T_n-S}{S_n-S}=0$.

Theorem: Let $(S_n)_{n\in\mathbb{N}}$ be a converging sequence to S, verifying $\lim\limits_{n\to\infty}\dfrac{S_{n+1}-S}{S_n-S}=\lambda$ with $\lambda\in]\!]-1;1[\![$. The sequence $(T_n)_{n\in\mathbb{N}}$ defined by $T_n=S_n+\dfrac{(S_{n+1}-S_n)^2}{(S_{n+1}-S_n)-(S_{n+2}-S_{n+1})}$, is an accelerating sequence of $(S_n)_{n\in\mathbb{N}}$.

Demonstration:

a. Let's demonstrate that $\lim\limits_{n\to\infty}T_n=S$

Adding $-S+S$ in all parenthesis and dividing numerator and denominator by $S_{n+1}-S$ gives $T_n=S_n+\dfrac{\left(1+\dfrac{S-S_n}{S_{n+1}-S}\right)(S_{n+1}-S_n)}{\left(1+\dfrac{S-S_n}{S_{n+1}-S}\right)-\left(\dfrac{S_{n+2}-S}{S_{n+1}-S}-1\right)}$ simplifying and taking the limit when $n\to\infty$ gives $\lim\limits_{n\to\infty}T_n=S+\lim\limits_{n\to\infty}\dfrac{\left(1-\dfrac{1}{\lambda}\right)}{1-\dfrac{1}{\lambda}-\lambda+1}(S_{n+1}-S_n)$ since $\lambda\neq 1$, the denominator is not zero therefore $(T_n)_{n\in\mathbb{N}}$ converges and $\lim\limits_{n\to\infty}T_n=S$.

[3] Using the same notations and definitions as in [2].

b. Let's demonstrate that $(T_n)_{n\in\mathbb{N}}$ converges faster than $(S_n)_{n\in\mathbb{N}}$.

$$\frac{T_n - S}{S_n - S} = \frac{S_n - S + \dfrac{(S_{n+1}-S_n)^2}{(S_{n+1}-S_n)-(S_{n+2}-S_{n+1})}}{S_n - S} = 1 + \frac{\dfrac{(S_{n+1}-S_n)^2}{(S_{n+1}-S_n)-(S_{n+2}-S_{n+1})}}{S_n - S}$$

Adding $-S+S$ in all parenthesis and dividing numerator and denominator by $S_{n+1}-S$

gives $\dfrac{T_n - S}{S_n - S} = 1 + \dfrac{\left(1+\dfrac{S-S_n}{S_{n+1}-S}\right)\left(-1+\dfrac{S_{n+1}-S}{S_n-S}\right)}{\left(1+\dfrac{S-S_n}{S_{n+1}-S}\right)-\left(-1+\dfrac{S_{n+2}-S}{S_{n+1}-S}\right)}$ therefore

$$\lim_{n\to\infty}\frac{T_n - S}{S_n - S} = 1 + \frac{\left(1-\dfrac{1}{\lambda}\right)(-1+\lambda)}{2-\lambda-\dfrac{1}{\lambda}} = 1 - \frac{(\lambda-1)^2}{\lambda^2-2\lambda+1} = 0$$

Therefore $(T_n)_{n\in\mathbb{N}}$ converges faster than $(S_n)_{n\in\mathbb{N}}$.

6. Iterative Aitken Accelerator

One might wonder why should we stop there? The Aitken accelerator gives an estimate of the limit of a convergent sequence given three consecutive terms, what if we created a sequence of the Aitken estimations and applied the Aitken formula on that sequence? What if we created a convergent sequence that used the Aitken accelerator to converge faster to the limit? Those are great ideas and it is indeed possible to use the Aitken iterator iteratively to accelerate the convergence further.

a. Simple iterative method.

Let's define the sequence $(x_n^i)_{n\in\mathbb{N}}$ by $x_n^i = x_{n+2}^{i-1} + \dfrac{\left(x_{n+2}^{i-1}-x_{n+1}^{i-1}\right)^2}{\left(x_{n+1}^{i-1}-x_n^{i-1}\right)-\left(x_{n+2}^{i-1}-x_{n+1}^{i-1}\right)}$ which is the result of creating $(x_n^i)_{n\in\mathbb{N}}$ by applying equation (14) to three consecutive terms of $\left(x_n^{i-1}\right)_{n\in\mathbb{N}}$. Following the assumption that $\left(x_n^{i-1}\right)_{n\in\mathbb{N}}$ converges, it is reasonable to believe that $(x_n^i)_{n\in\mathbb{N}}$ will also converge and will converge faster. This is reasonable but although it works in some instances it is very important to realize it is not always true.

So this is a very simple way to iteratively apply the Aitken accelerator. The advantage of this method is that you can get closer to the limit with few more arithmetic operations. The other advantage is that you can apply this process without the need of using f nor a sequence $(x_n)_{n\in\mathbb{N}}$ iteratively defined by $x_{n+1} = f(x_n)$. The disadvantage is that you need to store a certain number of values of $\left(x_n^{i-1}\right)_{n\in\mathbb{N}}$ that will be used to define the sequence $\left(x_n^i\right)_{n\in\mathbb{N}}$.

Table 1 shows the result of the iterative Aitken accelerator based on 5 steps of the sequences defined by $f(x) = \dfrac{9x - e^x}{10}$ which limit is $-LambertW(1)$ [4]. Reaching the same accuracy without the iterated Aitken accelerator would require calculating x_{33}.

Table 1

n	$(x_n)_{n\in\mathbb{N}}$	$(x_n^1)_{n\in\mathbb{N}}$	$(x_n^2)_{n\in\mathbb{N}}$
0	0.00000	-0.51239	-0.56476
1	-0.10000	-0.53054	
2	-0.18048	-0.54239	
3	-0.24592		
4	-0.29953		
Error	0.26762	0.02475	0.00238

Table 2 shows the result of the iterative Aitken accelerator based on 5 values of the serie defined by $x_n = \sum_{j=0}^{n} \dfrac{(-1)^j}{(1+2j)}$ that converges to $\dfrac{\pi}{4}$. Reaching the same accuracy without the iterated Aitken accelerator would require calculating x_{1951}.

Table 2

n	$(x_n)_{n\in\mathbb{N}}$	$(x_n^1)_{n\in\mathbb{N}}$	$(x_n^2)_{n\in\mathbb{N}}$
0	1.00000	0.79166	0.78552
1	0.66666	0.78333	
2	0.86666	0.78630	
3	0.72380		
4	0.83492		
Error	0.04952	0.00091	0.00001

b. Meyers' iterative method

First let us rewrite (13) by replacing x_{i+2} by $f(x_{i+1})$, and by replacing x_{i+1} by $f(x_i)$. It shows the Aitken formula in another form.

[4] See [9] for more details on the Lambert W function.

$$\overline{x} = x_{i+1} + \frac{\left(f\left(x_i\right) - x_i\right)}{\left(f\left(x_i\right) - x_i\right) - \left(f\left(x_{i+1}\right) - x_{i+1}\right)}\left(f\left(x_{i+1}\right) - x_{i+1}\right) \tag{15}$$

Let us now construct a sequence $\left(y_n\right)_{n\in\mathbb{N}}$.

y_0 is the iteration guess start value, chosen sometimes randomly or with a rough estimation depending the problem you are trying to solve.

y_1 is the first iteration and defined for the original sequence by $y_1 = f\left(y_0\right)$. Let's define $a_0 = 1$ then $y_1 = f\left(y_0\right) = y_0 + f\left(y_0\right) - y_0 = y_0 + a_0\left(f\left(y_0\right) - y_0\right)$.

Let's apply (15) to y_0 and y_1 to determine y_2.

$$y_2 = y_1 + \frac{\left(f\left(y_0\right) - y_0\right)}{\left(f\left(y_0\right) - y_0\right) - \left(f\left(y_1\right) - y_1\right)}\left(f\left(y_1\right) - y_1\right)$$

If we define $a_1 = \dfrac{\left(f\left(y_0\right) - y_0\right)}{\left(f\left(y_0\right) - y_0\right) - \left(f\left(y_1\right) - y_1\right)}$ and use the fact that $a_0 = 1$ then

$$y_2 = y_1 + a_1 a_0\left(f\left(y_1\right) - y_1\right)$$

$$y_2 - y_1 = a_1 a_0\left(f\left(y_1\right) - y_1\right) \tag{16}$$

We want to determine y_3 by using (13) to y_1, y_2 and another value y_2'. An interesting value for y_2' is given by

$$y_2' = y_2 + a_1 a_0\left(f\left(y_2\right) - y_2\right)$$

$$y_2' - y_2 = a_1 a_0\left(f\left(y_2\right) - y_2\right) \tag{17}$$

The idea is to interpolate or extrapolate in the direction $f(y) - y$ by the same distance $a_1 a_0$ used for y_2.

Using (13) to y_1, y_2 and y_2' gives $y_3 = y_2 + \dfrac{\left(y_2' - y_2\right)\left(y_2 - y_1\right)}{\left(y_2 - y_1\right) - \left(y_2' - y_2\right)}$

Using (16) and (17) gives $y_3 = y_2 + \dfrac{\left(a_1 a_0\left(f(y_2) - y_2\right)\right)\left(a_1 a_0\left(f(y_1) - y_1\right)\right)}{\left(a_1 a_0\left(f(y_1) - y_1\right)\right) - \left(a_1 a_0\left(f(y_2) - y_2\right)\right)}$

Simplifying numerator and denominator by $a_1 a_0$ gives

$$y_3 = y_2 + \frac{\left(f(y_1) - y_1\right)}{\left(f(y_1) - y_1\right) - \left(f(y_2) - y_2\right)} a_1 a_0\left(f(y_2) - y_2\right)$$

If we define $a_2 = \dfrac{\left(f(y_1) - y_1\right)}{\left(f(y_1) - y_1\right) - \left(f(y_2) - y_2\right)}$ then $y_3 = y_2 + a_2 a_1 a_0\left(f(y_2) - y_2\right)$

Repeating the exact same steps to create $y_i' = y_i + \left(\prod_{i=0}^{i-1} a_i\right)\left(f\left(y_i\right) - y_i\right)$ and applying (13) to y_{i-1}, y_i and y_i' gives the Meyers iterative Aitken formulas.

$$y_{i+1} = y_i + \left(\prod_{i=0}^{i} a_i\right)\left(f\left(y_i\right) - y_i\right) \tag{18}$$

With $a_i = \dfrac{\left(f\left(y_{i-1}\right) - y_{i-1}\right)}{\left(f\left(y_{i-1}\right) - y_{i-1}\right) - \left(f\left(y_i\right) - y_i\right)}$ for $1 \le i$ and $a_0 = 1$.

It is very important to realize here that it has not been demonstrated that $\left(y_n\right)_{n\in\mathbb{N}}$ is convergent, nor that it respects the necessary conditions to be accelerated by the Aitken formula. Although in practice the Meyers formula works really well in many cases and its convergence is in many case very impressive, it might be necessary to verify that it applies to the problem at hand. Nevertheless the advantage of this method is that it does not require storing all previous values of the sequence and it does not require more use of the function f.

Table 3 compares the results of the normal sequence, the iterative Aitken sequence and the Meyers method, based on 5 values of the sequence defined by $f\left(x\right) = \cos\left(x\right)$ that converges to approximately 0.73908513. Reaching the same accuracy without Meyers method, requires computing x_{43} with basic iterations and x_1^6 with the iterative Aitken.

Table 3

N	$\left(x_n\right)_{n\in\mathbb{N}}$	$\left(x_n^2\right)_{n\in\mathbb{N}}$	$\left(y_n\right)_{n\in\mathbb{N}}$
0	0.36235775	0.73727953	0.36235775
1	0.93506365		0.70250260
2	0.59376721		0.74284646
3	0.82883886		0.73905422
4	0.67573215		0.73908511
Error	0.06335298	0.00180561	0.25620330 $\times 10^{-7}$

c. Other iterative ideas that do not work as well

There are several creative ways you can try to iterate with the Aitken accelerator. Following are some examples of ideas that do not work really well in practice.

Reusing the method presented in 6.a by using (15) instead of (14). It accelerates the convergence however in most case the method presented in 6.a converges faster.

Moreover this method requires calling the function g during the iterations, and it will require n iterations to reach the final value.

Another method is based on using (15) on two preceding values of the sequence. This method works well in practice and has the same advantages as the Meyers method, no storage is required and no additional calls to f are required. However in most cases the Meyers method converges much faster.

A last method that might sound reasonable at first but is flawed is to use (14) on three preceding values of the sequence. Although this sounds good at first, it does not use the function g and therefore is not likely to converge on the right value. Imagine that two sequences have the same initial three values, with this method they would end-up having the same calculated wrong limit.

7. Using Meyers method to solve $g(x)=0$

Creating a function f that has for fixed point $\overline{x}=f(\overline{x})$ the root of g is relatively easy if we define f by the following $f(x)=x+w(x)g(x)$, $w(x)$ needs to be picked so that the sequence converges around $\overline{x}$.

The Meyers formulas (18) applied to f following this definition give:

$$y_{i+1}=y_i+\left(\prod_{i=0}^{i}a_i\right)\left(w(y_i)g(y_i)\right) \tag{19}$$

With $a_i=\dfrac{\left(w(y_{i-1})g(y_{i-1})\right)}{\left(w(y_{i-1})g(y_{i-1})\right)-\left(w(y_i)g(y_i)\right)}$ for $1\le i$ and $a_0=1$.

Most of the time $w(x)=1$ will be good enough to solve g.

8. Conclusion

A well known result in the domain of data compression is that there is no single compression algorithm that can compress all data. In other words for a given compression algorithm there exist some data that can not be compressed by the algorithm. Delahaye and Germain-Bonne [5] have shown a very similar result in the domain of convergence acceleration by proving that several families of sequences have no algorithm accelerating the convergence of every sequence of the family.

Therefore, as with the multitude of compression methods there are also several other convergence accelerations methods such as Euler's transformation, Wynn's ε algorithm, Brezinski's θ algorithm, Levin's transforms, and extrapolation methods.

The positive side of this incomplete solution is that there is always an interest to discover new acceleration formulas for sequences, and the reader of this tutorial can take part in this research effort.

9. References

1. C. Brezinski, Algorithmes d'Accélération de la Convergence, Étude Numérique, Editions Technip (1978)
2. C. Brezinski, M.R. Zaglia, Extrapolation Methods, Studies in Computational Mathematics, North Holland (1990)
3. A. Meyers, Accelerating Aitken's convergence accelerator, World Wide Web site at the address http://www.tm.bi.ruhr-uni-bochum.de/profil/mitarbeiter/meyers/aitken.pdf
4. S.D. Kamvar, et al., Extrapolation Methods for Accelerating PageRank Computations, World Wide Web site at the address http://www.stanford.edu/~sdkamvar/papers/extrapolation.pdf, (2003)
5. J. P. Delahaye and B. Germain-Bonne, Resultats negatifs en acceleration de la convergence, Numer. Math, Vol. 35 (1980), pp. 443-457,
6. X. Gourdon and P. Sebah, Convergence acceleration of series, World Wide Web site at the address http://numbers.computation.free.fr/Constants/Miscellaneous/seriesacceleration.html, (2002)
7. David A. Smith; William F. Ford, Numerical Comparisons of Nonlinear Convergence Accelerators, Mathematics of Computation, Vol. 38, No. 158. (Apr., 1982), pp. 481-499
8. J. Mathews, K. Fink, Numerical Methods Using MATLAB 3rd edition, Prentice Hall, (1999)
9. Robert M. Corless, et al., On the Lambert W Function, Advances in Computational Mathematics, volume 5 (1996), pp. 329-359.

10. Appendix. Non optimized C++ code.

```
// Returns an estimate of the limit given three consecutive terms of a sequence
float AitkenExtrapolation( const float x0, const float x1, const float x2 )
{
	float d1 = x1 - x0;
	float d2 = x2 - x1;
	return x0 + d1 * d1 / ( d1 - d2 );
}

// Produce another sequence of n-2 terms using the Aitken extrapolation
void IteratedAitken( float* const dst, const float* const src, const long n )
{
	for ( long i = 0; i < n - 2; i++ )
	{
		dst[ i ] = AitkenExtrapolation( src[ i ], src[ i + 1 ], src[ i + 2 ] );
	}
}

// Typedef used for Meyers functions
typedef float (*pSimpleFunction)( float x );
```

```
// Fixed point iteration using Meyers method. To accelerate Newton method use f=x-u(x)/u'(x)
float MeyersFixedPoint( pSimpleFunction f, const float startPoint, const long maxIteration, const
float epsilon )
{
	long iter = maxIteration;
	float res = f( startPoint ); // First iteration
	float last = res - startPoint;
	float prod = 1.0f;
	float step = FLT_MAX;

	while ( ( 1 < iter ) && ( epsilon <= fabsf( step ) ) ) // One iteration has already been done
	{
		float delta = f( res ) - res;
		float diff = last - delta;
		prod *= last / diff;
		step = prod * delta;
		res += step;
		last = delta;
		iter--;
	}

	return res;
}

// Root finding using Meyers method. To accelerate Newton method use g=u(x)/u'(x)
float MeyersRoot( pSimpleFunction g, const float startPoint, const long maxIteration, const float
epsilon )
{
	long iter = maxIteration;
	float last = g( startPoint );
	float res = startPoint + last; // First iteration
	float prod = 1.0f;
	float step = FLT_MAX;

	while ( ( 1 < iter ) && ( epsilon <= fabsf( step ) ) ) // One iteration has already been done
	{
		float delta = g( res );
		float diff = last - delta;
		prod *= last / diff;
		step = prod * delta;
		res += step;
		last = delta;
		iter--;
	}

	return res;
}
```