

## Accelerating Convergence

### *Aitkin's* Delta<sup>2</sup> Convergence

We illustrate how **Aitkin's method** accelerates convergence by studying its application to the fixed-point method.

```
> restart;  
> Digits:=15;
```

*Digits := 15*

We wish to find a root of the equation

$$x = 6^{-x}$$

in the interval [0.3,0.6] using the fixed-point method. We will use  $p_0 = 0.3$  as an initial approximation.

We enter the right-hand side as a Maple function.

```
> g:=x-> 6^(-x);
```

*g := x → 6<sup>-x</sup>*

We enter Aitkin's formula as a function  $a(n)$ .

```
> a:=n->p[n]-(p[n+1]-p[n])^2/(p[n+2]-2*p[n+1]+p[n]);
```

$$a := n \rightarrow p_n - \frac{(p_{n+1} - p_n)^2}{p_{n+2} - 2p_{n+1} + p_n}$$

We enter the initial value.

```
> p[0]:=0.3;
```

*p<sub>0</sub> := 0.3*

The following loop uses the **fixed-point method** to construct a sequence of approximations to our root until two successive approximations are within  $10^{-12}$ .

```
> n:=0:  
while (n<2 or abs(p[n]-p[n-1])>=10.0^(-12)) do  
n:=n+1:  
p[n]:=evalf(g(p[n-1]));  
od;
```

*n := 1*

*p<sub>1</sub> := 0.584190681067866*

*n := 2*

*p<sub>2</sub> := 0.351084204776959*

*n := 3*

*p<sub>3</sub> := 0.533093499816937*

*n := 4*

*p<sub>4</sub> := 0.384744682784904*

*n := 5*

$p_5 := 0.501892197375820$   
 $n := 6$   
 $p_6 := 0.406866524409851$   
 $n := 7$   
 $p_7 := 0.482387788558386$   
 $n := 8$   
 $p_8 := 0.421336748234447$   
 $n := 9$   
 $p_9 := 0.470041585510377$   
 $n := 10$   
 $p_{10} := 0.430761174577409$   
 $n := 11$   
 $p_{11} := 0.462170959784913$   
 $n := 12$   
 $p_{12} := 0.436878919833623$   
 $n := 13$   
 $p_{13} := 0.457132524794077$   
 $n := 14$   
 $p_{14} := 0.440840771985083$   
 $n := 15$   
 $p_{15} := 0.453898975046680$   
 $n := 16$   
 $p_{16} := 0.443402303551956$   
 $n := 17$   
 $p_{17} := 0.451820511680255$   
 $n := 18$   
 $p_{18} := 0.445056659503266$   
 $n := 19$   
 $p_{19} := 0.450483204738639$   
 $n := 20$   
 $p_{20} := 0.446124352830646$   
 $n := 21$   
 $p_{21} := 0.449622231815234$   
 $n := 22$

$p_{22} := 0.446813100527426$   
 $n := 23$   
 $p_{23} := 0.449067708644906$   
 $n := 24$   
 $p_{24} := 0.447257262193286$   
 $n := 25$   
 $p_{25} := 0.448710468869420$   
 $n := 26$   
 $p_{26} := 0.447543637731103$   
 $n := 27$   
 $p_{27} := 0.448480287371808$   
 $n := 28$   
 $p_{28} := 0.447728256167469$   
 $n := 29$   
 $p_{29} := 0.448331958290509$   
 $n := 30$   
 $p_{30} := 0.447847264735975$   
 $n := 31$   
 $p_{31} := 0.448236368538529$   
 $n := 32$   
 $p_{32} := 0.447923975827319$   
 $n := 33$   
 $p_{33} := 0.448174763658722$   
 $n := 34$   
 $p_{34} := 0.447973420909319$   
 $n := 35$   
 $p_{35} := 0.448135059959680$   
 $n := 36$   
 $p_{36} := 0.448005290638548$   
 $n := 37$   
 $p_{37} := 0.448109470883649$   
 $n := 38$   
 $p_{38} := 0.448025831914261$   
 $n := 39$

$p_{39} := 0.448092978506753$   
 $n := 40$   
 $p_{40} := 0.448039071440087$   
 $n := 41$   
 $p_{41} := 0.448082348950687$   
 $n := 42$   
 $p_{42} := 0.448047604697752$   
 $n := 43$   
 $p_{43} := 0.448075498027722$   
 $n := 44$   
 $p_{44} := 0.448053104608197$   
 $n := 45$   
 $p_{45} := 0.448071082482036$   
 $n := 46$   
 $p_{46} := 0.448056649437278$   
 $n := 47$   
 $p_{47} := 0.448068236576075$   
 $n := 48$   
 $p_{48} := 0.448058934164140$   
 $n := 49$   
 $p_{49} := 0.448066402331407$   
 $n := 50$   
 $p_{50} := 0.448060406723561$   
 $n := 51$   
 $p_{51} := 0.448065220122181$   
 $n := 52$   
 $p_{52} := 0.448061355821612$   
 $n := 53$   
 $p_{53} := 0.448064458163091$   
 $n := 54$   
 $p_{54} := 0.448061967536637$   
 $n := 55$   
 $p_{55} := 0.448063967064018$   
 $n := 56$

$p_{56} := 0.448062361800612$   
 $n := 57$   
 $p_{57} := 0.448063650540000$   
 $n := 58$   
 $p_{58} := 0.448062615912489$   
 $n := 59$   
 $p_{59} := 0.448063446533367$   
 $n := 60$   
 $p_{60} := 0.448062779693213$   
 $n := 61$   
 $p_{61} := 0.448063315046631$   
 $n := 62$   
 $p_{62} := 0.448062885253503$   
 $n := 63$   
 $p_{63} := 0.448063230300549$   
 $n := 64$   
 $p_{64} := 0.448062953289435$   
 $n := 65$   
 $p_{65} := 0.448063175679841$   
 $n := 66$   
 $p_{66} := 0.448062997140091$   
 $n := 67$   
 $p_{67} := 0.448063140475595$   
 $n := 68$   
 $p_{68} := 0.448063025402804$   
 $n := 69$   
 $p_{69} := 0.448063117785686$   
 $n := 70$   
 $p_{70} := 0.448063043618743$   
 $n := 71$   
 $p_{71} := 0.448063103161542$   
 $n := 72$   
 $p_{72} := 0.448063055359316$   
 $n := 73$

$p_{73} := 0.448063093735960$   
 $n := 74$   
 $p_{74} := 0.448063062926374$   
 $n := 75$   
 $p_{75} := 0.448063087660964$   
 $n := 76$   
 $p_{76} := 0.448063067803510$   
 $n := 77$   
 $p_{77} := 0.448063083745496$   
 $n := 78$   
 $p_{78} := 0.448063070946931$   
 $n := 79$   
 $p_{79} := 0.448063081221891$   
 $n := 80$   
 $p_{80} := 0.448063072972934$   
 $n := 81$   
 $p_{81} := 0.448063079595372$   
 $n := 82$   
 $p_{82} := 0.448063074278738$   
 $n := 83$   
 $p_{83} := 0.448063078547045$   
 $n := 84$   
 $p_{84} := 0.448063075120357$   
 $n := 85$   
 $p_{85} := 0.448063077871375$   
 $n := 86$   
 $p_{86} := 0.448063075662800$   
 $n := 87$   
 $p_{87} := 0.448063077435891$   
 $n := 88$   
 $p_{88} := 0.448063076012416$   
 $n := 89$   
 $p_{89} := 0.448063077155211$   
 $n := 90$

$p_{90} := 0.448063076237751$   
 $n := 91$   
 $p_{91} := 0.448063076974308$   
 $n := 92$   
 $p_{92} := 0.448063076382984$   
 $n := 93$   
 $p_{93} := 0.448063076857712$   
 $n := 94$   
 $p_{94} := 0.448063076476590$   
 $n := 95$   
 $p_{95} := 0.448063076782563$   
 $n := 96$   
 $p_{96} := 0.448063076536921$   
 $n := 97$   
 $p_{97} := 0.448063076734128$   
 $n := 98$   
 $p_{98} := 0.448063076575806$   
 $n := 99$   
 $p_{99} := 0.448063076702910$   
 $n := 100$   
 $p_{100} := 0.448063076600868$   
 $n := 101$   
 $p_{101} := 0.448063076682790$   
 $n := 102$   
 $p_{102} := 0.448063076617021$   
 $n := 103$   
 $p_{103} := 0.448063076669822$   
 $n := 104$   
 $p_{104} := 0.448063076627432$   
 $n := 105$   
 $p_{105} := 0.448063076661464$   
 $n := 106$   
 $p_{106} := 0.448063076634142$   
 $n := 107$

```

p107 := 0.448063076656077
      n := 108
p108 := 0.448063076638467
      n := 109
p109 := 0.448063076652604
      n := 110
p110 := 0.448063076641255
      n := 111
p111 := 0.448063076650366
      n := 112
p112 := 0.448063076643052
      n := 113
p113 := 0.448063076648924
      n := 114
p114 := 0.448063076644209
      n := 115
p115 := 0.448063076647995
      n := 116
p116 := 0.448063076644955
      n := 117
p117 := 0.448063076647396
      n := 118
p118 := 0.448063076645436
      n := 119
p119 := 0.448063076647010
      n := 120
p120 := 0.448063076645746
      n := 121
p121 := 0.448063076646761
      n := 122
p122 := 0.448063076645946

```

Quite a few steps, 122 to be exact. The convergence in this case is certainly no more than linear. We now create the **Aitkin sequence** using the previous sequence.

```

> n:=0:
  while (n<2 or abs(pp[n]-pp[n-1])>=10.0^(-12)) do

```

```
n:=n+1:  
pp[n]:=a(n)  
od;
```

```
n := 1  
pp1 := 0.453290741721909  
n := 2  
pp2 := 0.451361395711290  
n := 3  
pp3 := 0.450202066326030  
n := 4  
pp4 := 0.449425523624652  
n := 5  
pp5 := 0.448945608739211  
n := 6  
pp6 := 0.448628002775637  
n := 7  
pp7 := 0.448428518165384  
n := 8  
pp8 := 0.448297644775226  
n := 9  
pp9 := 0.448214645321998  
n := 10  
pp10 := 0.448160519378368  
n := 11  
pp11 := 0.448125988281332  
n := 12  
pp12 := 0.448103560634296  
n := 13  
pp13 := 0.448089199115509  
n := 14  
pp14 := 0.448079896387558  
n := 15  
pp15 := 0.448073925420985  
n := 16  
pp16 := 0.448070064492589
```

$n := 17$   
 $pp_{17} := 0.448067582655499$   
 $n := 18$   
 $pp_{18} := 0.448065979693353$   
 $n := 19$   
 $pp_{19} := 0.448064948314233$   
 $n := 20$   
 $pp_{20} := 0.448064282664655$   
 $n := 21$   
 $pp_{21} := 0.448063854110429$   
 $n := 22$   
 $pp_{22} := 0.448063577655592$   
 $n := 23$   
 $pp_{23} := 0.448063399600446$   
 $n := 24$   
 $pp_{24} := 0.448063284775064$   
 $n := 25$   
 $pp_{25} := 0.448063210801305$   
 $n := 26$   
 $pp_{26} := 0.448063163106195$   
 $n := 27$   
 $pp_{27} := 0.448063132374672$   
 $n := 28$   
 $pp_{28} := 0.448063112562870$   
 $n := 29$   
 $pp_{29} := 0.448063099796144$   
 $n := 30$   
 $pp_{30} := 0.448063091566460$   
 $n := 31$   
 $pp_{31} := 0.448063086262894$   
 $n := 32$   
 $pp_{32} := 0.448063082844296$   
 $n := 33$   
 $pp_{33} := 0.448063080641106$

$n := 34$   
 $pp_{34} := 0.448063079221013$   
 $n := 35$   
 $pp_{35} := 0.448063078305778$   
 $n := 36$   
 $pp_{36} := 0.448063077715865$   
 $n := 37$   
 $pp_{37} := 0.448063077335666$   
 $n := 38$   
 $pp_{38} := 0.448063077090612$   
 $n := 39$   
 $pp_{39} := 0.448063076932673$   
 $n := 40$   
 $pp_{40} := 0.448063076830876$   
 $n := 41$   
 $pp_{41} := 0.448063076765267$   
 $n := 42$   
 $pp_{42} := 0.448063076722980$   
 $n := 43$   
 $pp_{43} := 0.448063076695725$   
 $n := 44$   
 $pp_{44} := 0.448063076678159$   
 $n := 45$   
 $pp_{45} := 0.448063076666837$   
 $n := 46$   
 $pp_{46} := 0.448063076659540$   
 $n := 47$   
 $pp_{47} := 0.448063076654836$   
 $n := 48$   
 $pp_{48} := 0.448063076651805$   
 $n := 49$   
 $pp_{49} := 0.448063076649851$   
 $n := 50$   
 $pp_{50} := 0.448063076648592$

```
n := 51
pp51 := 0.448063076647780
```

We still need 104 steps, 51 here and 53 from the original sequence. In the loop below, we replace every third iteration by Aitkin's formula applied to the three previous iterations. This is **Steffensen's method**.

```
> n:=0:
  p[0]:=0.3:
  while (n<2 or abs(p[n]-p[n-1])>=10.0^(-12)) do
    n:=n+1:
    if n mod 3 = 0 then
      p[n]:=evalf(a(n-3));
    else
      p[n]:=evalf(g(p[n-1]));
    fi;
    print(p[n]);
  od;
```

```
n := 1
0.584190681067866
n := 2
0.351084204776959
n := 3
0.456127560449368
n := 4
0.441635288962987
n := 5
0.453253271752731
n := 6
0.448083750128186
n := 7
0.448046479842922
n := 8
0.448076401111039
n := 9
0.448063076783193
n := 10
0.448063076536415
n := 11
0.448063076734534
n := 12
0.448063076646309
n := 13
0.448063076646309
```

Only 13 steps needed. A considerable savings. Now we apply the class library procedure `steffensen` to

implement this method.

## *nalib*

### *Steffensen's Method*

We first import our class library.

```
> libname:="c:/nalib",libname;
libname := "/nalib", "/Library/Frameworks/Maple.framework/Versions/15/lib",
"/Library/Frameworks/Maple.framework/Versions/15/toolbox/NAG/lib"
> with(numanal);
[SOR, SOR_dir, adaptq, adaptq_dir, bezier, bezier_dir, bisection, bisection_dir, chop, chop_dir,
clamped_spline, clamped_spline_dir, divided_diff, divided_diff_dir, extrapol, extrapol_dir,
falseposition, falseposition_dir, fixedpoint, fixedpoint_dir, gaussseidel, gaussseidel_dir, hermite,
hermite_dd, hermite_dd_dir, hermite_dir, horner, horner_dir, jacobi, jacobi_dir, muller,
muller_dir, natural_spline, natural_spline_dir, newton, newton_dir, romberg, romberg_dir,
secant, secant_dir, steffensen, steffensen_dir]
```

We check the directions for **steffensen**.

```
> steffensen_dir();
steffensen returns a root a function of the form x=g(x).
It allows up to 15 decimal places.
```

The arguments for **steffensen** are:

- (1)function expression in x
- (2)initial approximation
- (3)tolerance
- (4)maximum number of iterations
- (5)variable for returning root

If assigning the result to a variable, have the variable and the 5th argument the same.

If r is the variable for returning the root and has already been given a value, the procedure should be preceded by the statement:  
`r:='r'`

We now apply the procedure **steffensen** to the above equation.

```
> r:=steffensen(g(x),.3,10^(-12),25,r);
k          p[0](k)          p[1](k)          p[2](k)
-----
1  3.000000000000000e-01  5.841906810678660e-01
3.510842047769590e-01
2  4.561275604493680e-01  4.416352889629870e-01
4.532532717527310e-01
3  4.480837501281860e-01  4.480464798429220e-01
4.480764011110390e-01
4  4.480630767831930e-01  4.480630765364150e-01
4.480630767345340e-01
```

Denominator = 0, method fails  
Best possible solution is r = 0.448063076646309  
with g(r) = 0.448063076646309  
r :=

Here the method terminated since one of the denominators in an Aitkin's step was 0. When this happens, we go back to the last iteration in a right-hand column for a solution. When this happens, check the returned  $r$  and  $g(r)$  to see if they are close enough for your uses before proceeding. Here they are the same to 15 digits.

Next we apply Steffensen's method to solve the equation  $x = g(x)$  where  $g(x) = \sqrt{\frac{10}{x+4}}$ . This equation is known to have a root between 1 and 2, so we use 1.5 as an initial approximation.

```
> r:='r';
                                     r := r
> r:=numanal[steffensen](sqrt(10/(x+4)),1.5,.000005,25,r);
k          p[0](k)                   p[1](k)                   p[2](k)
-----
1  1.5000000000000000e+00  1.348399724926480e+00
1.367376371991280e+00
2  1.365265223957260e+00  1.365225533619790e+00
1.365230583376010e+00
3  1.365230013416590e+00  1.365230013413780e+00
1.365230013414140e+00
4  1.365230013414100e+00

The approximate solution is r =      1.365230013414100
with g(r) =      1.365230013414100
                                     r := 1.36523001341410
```

Looks pretty good.

### *NumericalAnalysis*

There is a procedure called [Steffensen](#) in the **NumericalAnalysis** subpackage of the **Student** package.

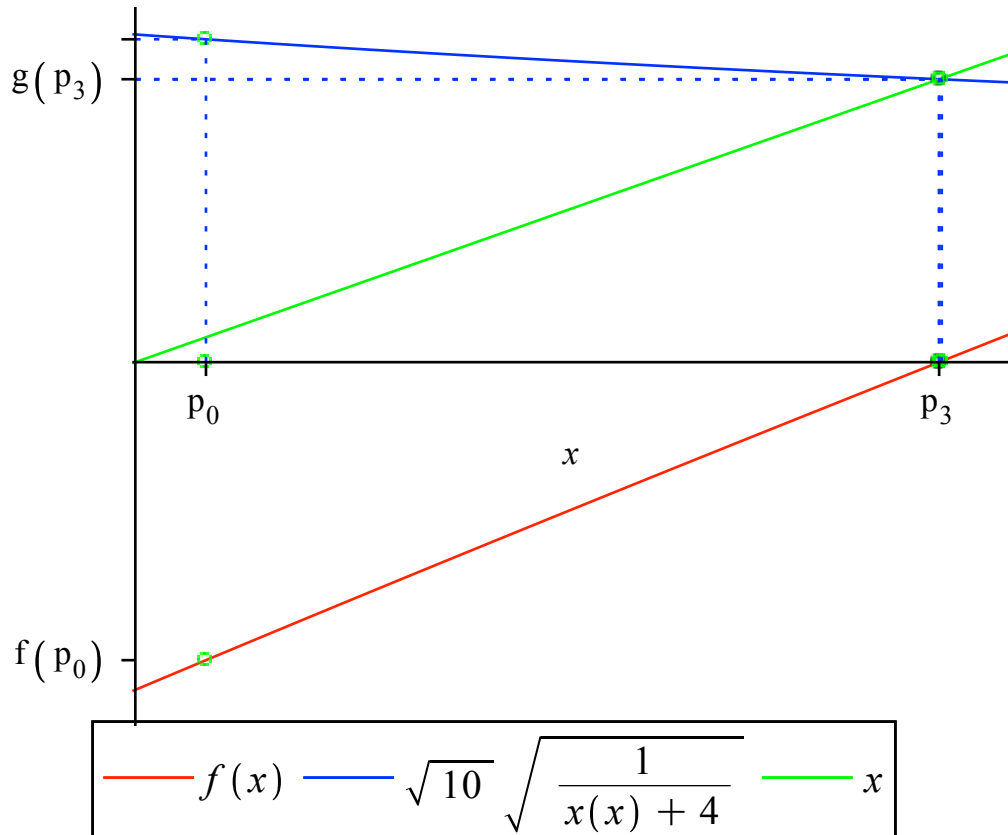
We use it to find the fixed point of  $g(x) = \sqrt{\frac{10}{x+4}}$ .

```
> g:=sqrt(10/(x+4));
                                     g := sqrt(10) sqrt(1/(x+4))
> with(Student[NumericalAnalysis]):
> Steffensen(fixedpointiterator=g,x=.12, stoppingcriterion=absolute,
tolerance=10^(-12),output=information,maxiterations=100);
[ n          Pn,0          Pn,1          Pn,2          absolute error ]
[ 0          0.12          1.55794238212439  1.34135267558335  - ]
[ 1  1.36970576785694  1.36466092188612  1.36530242434948  1.24970576785694 ]
[ 2  1.36523005358021  1.36523000830379  1.36523001406428  0.00447571427673 ]
> Steffensen(fixedpointiterator=g,x=.12, stoppingcriterion=absolute,
tolerance=10^(-6),output=plot,maxiterations=100);
```

3 iterations of Steffensen's method applied to

$$f(x) = x - \sqrt{10} \sqrt{\frac{1}{x+4}}$$

with initial point  $p_0 = 0.12$



Now let's try **Steffensen** with  $g(x) = 6^{-x}$ .

> **g:=6<sup>-x</sup>;**

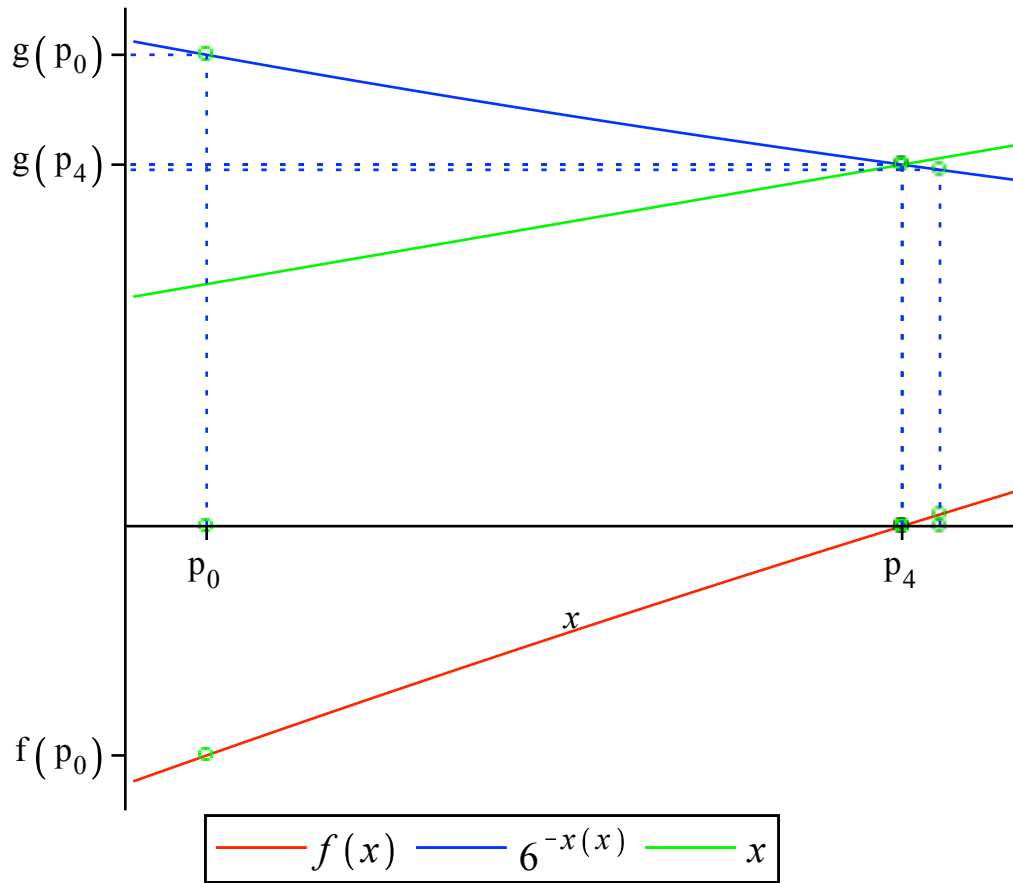
$$g := 6^{-x}$$

> **Steffensen(fixedpointiterator=g,x=.3, stoppingcriterion=absolute, tolerance=10<sup>-12</sup>, output=information, maxiterations=100);**

$n$	$p_{n,0}$	$p_{n,1}$	$p_{n,2}$	absolute error
0	0.3	0.584190681067866	0.351084204776959	-
1	0.456127560449368	0.441635288962987	0.453253271752731	0.156127560449368
2	0.448083750128186	0.448046479842922	0.448076401111039	0.008043810321182
3	0.448063076783193	0.448063076536415	0.448063076734534	0.000020673344993

> **Steffensen(fixedpointiterator=g,x=.3, stoppingcriterion=absolute, tolerance=10<sup>-6</sup>, output=plot, maxiterations=100);**

4 iterations of Steffensen's method applied to  
 $f(x) = x - 6^{-x}$   
 with initial point  $p_0 = 0.3$



OK, so what has Steffensen's method accomplished here?