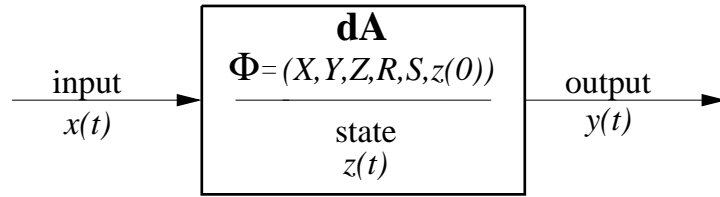


## Modul 72

An Algorithm for calculation of final state output parameters of the Feigenbaum automaton

The Feigenbaum automaton



with

$$X = Z = \mathbb{R}[0, 2]; Y = \mathbb{R}[0, 1] \quad (1)$$

$$y = r(x, z) = \frac{z}{x} \quad (2)$$

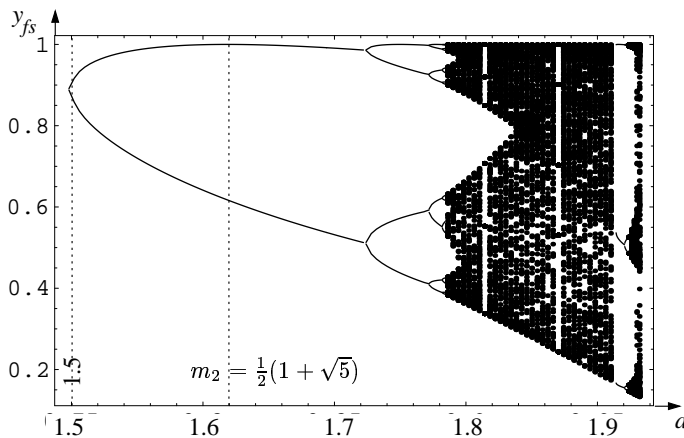
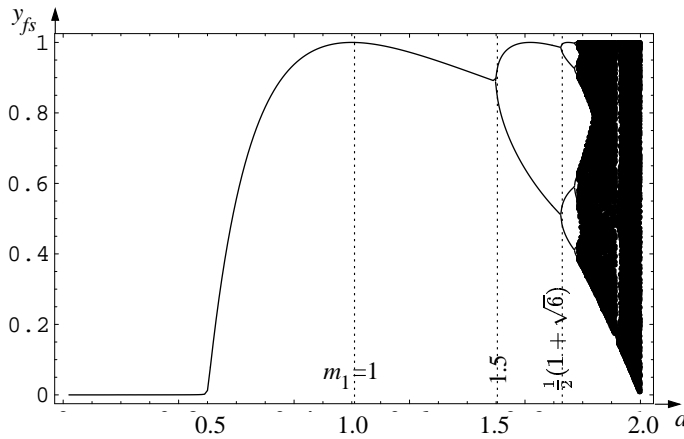
$$z' = s(x, z) = xz(2 - z) \quad (3)$$

(see also [1])

shows for constant input values

$$x(t) = a \quad (4)$$

after a transient response the following final state output characteristic  $y_{fs}(a)$  : Fig. 1.



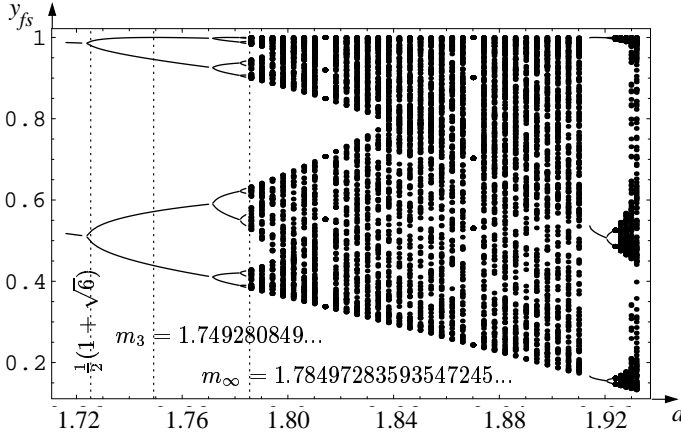


Figure 1: Final state output characteristic  $y_{fs}(a)$  of the Feigenbaum automaton

The algorithm (A1) allows to calculate efficiently some parameters of this final state output characteristic:

- maximum coordinates  $m_n$ ;  $n = 3, 4, 5, \dots$  in the bifurcation area  $1.5 \leq a \leq m_\infty$
- coordinate  $m_\infty$  of the entrance from bifurcation into the deterministic chaos

$$m_\infty = \lim_{n \rightarrow \infty} m_n \quad (5)$$

- Feigenbaum constant = limit of difference quotient of  $m_n$

$$\delta_n = \frac{m_{n-1} - m_{n-2}}{m_n - m_{n-1}} \quad (6)$$

$$\delta = \lim_{n \rightarrow \infty} \frac{m_{n-1} - m_{n-2}}{m_n - m_{n-1}} \quad (7)$$

The calculation of  $m_\infty$  and of  $\delta$  can be accelerated by linear tuning:

$$(m_{\infty,n})_{tuned} = m_n + \frac{(m_n - m_{n-1})^2}{2m_{n-1} - m_{n-2} - m_n} \quad (8)$$

$$m_\infty = \lim_{n \rightarrow \infty} (m_{\infty,n})_{tuned} \quad (9)$$

$$(\delta_n)_{tuned} = \delta_n + \frac{(\delta_n - \delta_{n-1})^2}{2\delta_{n-1} - \delta_{n-2} - \delta_n} \quad (10)$$

$$\delta = \lim_{n \rightarrow \infty} (\delta_n)_{tuned}. \quad (11)$$

The following tables 1, 2 and 3 show the value pattern of  $m_n$ ,  $(m_{\infty,n})_{tuned}$ ,  $\delta_n$ , and  $(\delta_n)_{tuned}$ , respectively, calculated by means of (A1).

The tuning of  $m_n$  according equ. (8) doubles the number of valid digits for  $m_\infty$  !

The tuning gain for  $\delta$  according equ. (10) is about 30%.

$-3.5 < m_0 < -0.3; m_1 := 1; m_2 = \frac{1}{2}(1 + \sqrt{5})$
$n = 2(1) \infty$ <u>and</u> $ m_n - m_{n-1}  \geq \varepsilon_1$
$u := m_n; v := m_n + \frac{(m_n - m_{n-1})^2}{m_{n-1} - m_{n-2}}$
$ u - v  \geq \varepsilon_2$
$u := v; z := 0; y := 0; x := 0$
$i = 1(1)2^n$
$g := 2ux; h := 1 - x^2$
$z := -(uy + 2x)y - gz$
$y := h - gy$
$x := uh - 1$
$v := u - \frac{x}{y - \frac{xz}{y}}$
$m_{n+1} := v$
$\delta_{n+1} := \frac{m_n - m_{n-1}}{m_{n+1} - m_n}$
$(m_{\infty, n+1})_{tuned} := v + \frac{(v - m_n)^2}{2m_n - m_{n-1} - v}$
$(\delta_{n+1})_{tuned} := \delta_{n+1} + \frac{(\delta_{n+1} - \delta_n)^2}{2\delta_n - \delta_{n-1} - \delta_{n+1}}$

(A1)

## References

- [1] <http://www.DresdenAlgorithmicsChannel.de>  
 Modul 24, Modul 48

$n$	$m_n$
3	1.74928084966385075999947269097
4	1.777320431384411243268304092597
5	1.78333368992813425698631557873
6	1.78462176581855516890412475546
7	1.78489764687497231025767626480
8	1.78495673271117425742048675983
9	1.78496938711665274389672303378
10	1.78497209730403246662181667194
11	1.78497267774323429044640119115
12	1.78497280205553921906705863069
13	1.78497282867942824986480380728
14	1.78497283438144998417354883490
15	1.78497283560264842726445704705
16	1.78497283586419173710253450960
17	1.78497283592020630484591723504
18	1.78497283593220290992595787159
19	1.78497283593477221536281405745
20	1.78497283593532248190770844775
21	1.78497283593544033215084672044
22	1.78497283593546557206400628812
23	1.78497283593547097768048477996
24	1.78497283593547213539799861640

Table 1: Value pattern of  $m_n$

$n$	$(m_{\infty,n})tuned$
3	1.78466744375138556905709021675
4	1.78493828691949350887103904030
5	1.78497532925319710410004864773
6	1.78497289287992380942021584231
7	1.78497284026139652353213056052
8	1.78497283609221844269299303296
9	1.78497283594378382423819551256
10	1.78497283593582435395614663654
11	1.78497283593548937022594371581
12	1.78497283593547320642156412835
13	1.78497283593547248611932519868
14	1.78497283593547245252107922606
15	1.78497283593547245099477756010
16	1.78497283593547245092437641383
17	1.78497283593547245092115759644
18	1.78497283593547245092100967917
19	1.78497283593547245092100290171
20	1.78497283593547245092100259064
21	.78497283593547245092100257638
22	1.78497283593547245092100257572
23	1.78497283593547245092100257569
24	1.78497283593547245092100257569

Table 2: Value pattern of  $(m_{\infty,n})tuned$

$n$	$\delta_n$
20	4.66920160910254658397856194996
21	4.66920160910289554808527091833
22	4.66920160910297030095467173380
23	4.66920160910298630882039845817
24	4.66920160910298973745117232519

$n$	$(\delta_n)tuned$
20	4.66920160910299116389386677544
21	4.66920160910299061106576265973
22	4.66920160910299067937365452591
23	4.66920160910299067092354082045
24	4.66920160910299067196817813386

Table 3: Value pattern of  $\delta_n$  and  $(\delta_n)tuned$

Open problems

1. The simple repetition of the linear tuning by means of equ. (8) to

$$\begin{aligned}
(m_{\infty,n})tuned2 &= (m_{\infty,n})tuned + \\
&\quad \frac{((m_{\infty,n})tuned - (m_{\infty,n-1})tuned)^2}{2(m_{\infty,n-1})tuned - (m_{\infty,n-2})tuned - (m_{\infty,n})tuned}
\end{aligned} \tag{12}$$

doesn't provide any significant extension of the number of valid digits for  $m_\infty$ .

Why?

Find a better approach.

2. From the initialization and the loop body of the algorithm (A1) follows: all  $m_n$  can be represented by power series in

$$\Phi = \frac{1}{2}(1 + \sqrt{5}) \quad (13)$$

$$m_n = \sum_{i=0}^{\infty} a_{n,i} \Phi^{-i} \quad (14)$$

Find an algorithm for calculation of the  $a_{n,i}$ .

3. For numerical algorithms

- which output sequence

$$\langle x(i) \rangle ; i = 0, 1, 2, \dots \quad (15)$$

converges linearly:

$$\lim_{i \rightarrow \infty} x(i) = p \quad (16)$$

$$\lim_{i \rightarrow \infty} \frac{x(i-2) - x(i-1)}{x(i-1) - x(i)} = q, \quad (17)$$

- whereas the values of the  $x(i)$  are calculated with a sufficient large number of digits,

the number of valid digits for a given  $i$  can be extended significantly by repeated application of the linear tuning formula

(separation of algorithmic noise and signal  $p$ )

$$x(k+1, i) = x(k, i) - \frac{1}{q^{k+1} - 1} (x(k, i-1) - x(k, i)) \quad (18)$$

$k = 0, 1, 2, \dots$  - correction depth

$x(0, i) = x(i)$  - initialization of correction

Compare the tuning performance of equ. (18) regarding the algorithms (A1) ( $m_n \hat{=} x(i)$  ;  $q = \delta$ ), (A2) ( $q = 4$ ), and (A3) ( $q = 4$ ).

$i = 0(1)\infty$ <u>and</u> $(x(i-1) - x(i)) \geq \varepsilon$	
$u := 2^i$	
$x(i) := \left(\frac{2u+1}{2u-1}\right)^u$	
$e(\varepsilon) := x(i)$	(A2)



$i$	$x(2, i)$
3	0.3395762409723822737879227296852642974624318501677344970387676980276914
4	0.3183098467857121705482397239674459152263244017561803000858606112663698
5	0.3183098855775447976896634955647687204748555934494690209331137938263833
6	0.3183098861743540329797308847119581168641441171401956140053242848507087
7	0.3183098861836433639577703800707812276795188973488403951775682595606762
8	0.3183098861837883704027746186248158065200395693033198223189183664591881
9	0.3183098861837906355846653414154525348408917800714512333507028256523069
10	0.3183098861837906709760086330275130253118960016899437260802689133779713
11	0.3183098861837906715289900765611693765302963642864025463608255755118166
12	0.3183098861837906715376303792129776155354433323911737458764214331110920
13	0.3183098861837906715377653838153367988317977877294535866695572289981783

$i$	$x(3, i)$
3	0.3449663400354359606734453126961415085332641017576985366743036932344802
4	0.3179722849732253435126892000671630837622592041623461064834335146511107
5	0.3183098861932881727236543490821865427803878345874577307878320983114629
6	0.3183098861838271954446525893015960437909582206908420678636133402637932
7	0.3183098861837908136558344990447308008670645287807236456723657829687710
8	0.3183098861837906720923778605066258792000478339375179084640191618067835
9	0.3183098861837906715399334481264150225920164183376120493988263884966422
10	0.3183098861837906715377759868626250965892135290172213846950239306434581
11	0.3183098861837906715377675597918623344861439890895209403335328241171157
12	0.3183098861837906715377675268741174288529853477579161458687324784698107
13	0.3183098861837906715377675267455329763444383346395850127138927178217828

$i$	$x(4, i)$
3	0.3463191492120455134604000001969106909196690590194934328965558645804977
4	0.3178664259337656940336274113901866977827650673090309989140575531664701
5	0.3183112101196413602891875457449905563451648096087326783341238574630721
6	0.3183098861837900935729702294593192183047251633814435750678320902322338
7	0.3183098861837906709821528594750960352085394554791153381343608905872218
8	0.3183098861837906715372270501594176246052752194479367094945746652532071
9	0.3183098861837906715377669994504141957033574716097692813240609266405240
10	0.3183098861837906715377675262302180772715554784708669115001070582597201
11	0.3183098861837906715377675267445260099288770497172554483948603099738752
12	0.3183098861837906715377675267450282331446200197526941662825960065260957
13	0.3183098861837906715377675267450287235895028561567680278779913854270847

$i$	$x(5, i)$
3	0.3466576821047259098567444772254511705784370639647715300939132016915246
4	0.3178386129100332601925455222515681992557495208850716025954040846313635
5	0.3183116449037917763852340365899121632450303030029650357920124853265096
6	0.3183098848896201509857598508510243294112349975493543384077477385536112
7	0.3183098861837906715465802129746226987636751488242743525655109676159168
8	0.3183098861837906715377696446762155929233560950725396629367645125402706
9	0.3183098861837906715377675272600828140524269557272295282173448819302476
10	0.318309886183790671537767526745154346950507736877399297114228413795825
11	0.3183098861837906715377675267450287547310836201681316835237896875415332
12	0.3183098861837906715377675267450287240764047147380660516373249759264400
13	0.3183098861837906715377675267450287240689211189489518732949956819851502

$i$	$x(6, i)$
3	0.3467423359953497745477961852784976788008005406592684217984538398360158
4	0.3178315752862982925125298448510861765521298169673476221491285024293952
5	0.3183117604183201667432786491697261666550398953795506019539652590507331
6	0.3183098844598242604789467924418084837863671598679248852459933393480501
7	0.3183098861841067083037379224647627716255438765779434417020574299429162
8	0.3183098861837906715377674931332855789780959438994623162482104841019127
9	0.3183098861837906715377675267430092773542899766944223706752888576004283
10	0.3183098861837906715377675267450286012930958276115398101513481351425737
11	0.3183098861837906715377675267450287240614329671391891662475533912890697
12	0.3183098861837906715377675267450287240689188346634799550239216878405756
13	<u>0.3183098861837906715377675267450287240689192914530326756934468836227713</u>

Table 4: Value patterns of  $x(i)$  and  $x(k, i)$ . The valid digits are underlined.

The tuning (filtering) gain is in this case about 7 digits per correcting step.

e.p.stoschek	30.01.2006
stoschek@tcs.inf.tu-dresden.de	