Answers to P-Set # 07, (18.353/12.006/2.050)j
MIT (Fall 2023)

Rodolfo R. Rosales (MIT, Math. Dept., room 2-337, Cambridge, MA 02139)

December 1, 2023

Contents

List of Figures

1 Liapunov exponents for 1-D maps #02

1.1 Statement: Liapunov exponents for 1-D maps #02

Compute the Liapunov exponent, and produce a figure analog to figure 10.5.2 in Strogatz's book (see example 10.5.3 there) **for the 1-D maps $x_{n+1} = f(x_n)$ below.** In all the cases, **given the range of r selected, justify the selected range for x.**

Meaning of "justify". Show that the x-region is such that it may contain an attractor, because either: (a) It is trapping; an orbit starting there stays there; or (b) Orbits starting outside the region diverge to infinity, so that any attractor has to be inside.

1. The **cosine map** $f(x) = r\,\cos(x)$, with $-5 \leq r \leq 5$ and $-5 \leq x \leq 5$.
2. The **quartic map** $f(x) = r\,(1-(2\,x-1)^4)$, with $0 \leq r \leq 1$ and $0 \leq x \leq 1$.
3. The **cusp075 map** $f(x) = r\,(1-z^\mu)$, with $0 \leq r \leq 1$ and $0 \leq x \leq 1$, where $z = |2\,x-1|$ and $\mu = 3/4$.
   Important. $\mathrm{d}f/\mathrm{d}x$ for this map involves $z^{\mu-1}$. To **avoid a potential division by zero,** when calculating use $z = |2\,x-1| + \epsilon$, where ϵ is very small; e.g.: $\epsilon = 10^{-200}$.

In all cases, plot not just the region listed above, but **do a detail of the r-ranges where the exponent transitions from negative to positive. See also the optional task below, after the hints.**

Hints and related.

h1. The **process to follow to calculate the Liapunov exponent** is explained in example 10.5.3 of Strogatz book [also in the lectures]. At any rate, see the *sample/sketch MatLab script* at the end of the problem statement.

h2. If you use MatLab, "vectorize" the operation, so that you do **all** the r's simultaneously.

h3. In MatLab "print -dpng FigureName" added to the script saves the figure as a **small png file** — and it is more reliable than saving the figure using the figure window GUI. **Please be careful with the figure sizes, do not upload monster size answers.** As a reference: in my answer the pictures take about 30kb each.

Optional task. For the case of the **cusp075** map, plot the Liapunov exponent versus r for the region where the Liapunov exponent transitions from negative to positive — this is, roughly $\mathbf{0.52 < r < 0.68}$. Use for this a program following the outline of the "sample/sketch MatLab script" below. Run the program a few times (without changing any parameters) and do a few plots. **What do you see?** (you should be seeing something that "should not" be, remember that you are looking at a deterministic system). **Explain why you see what you see.** Note: *The explanation is simple and clean. If you find yourself making convoluted arguments, you are on the wrong track!* **Hints:** (i) Do an orbit/bifurcation diagram for the map. (ii) Initialize the iterations that compute the Liapunov exponent with $\boldsymbol{x_0 = 0.51}$. (iii) Initialize the iterations that compute the Liapunov exponent with $\boldsymbol{x_0 = 0.01}$.

Sample/sketch MatLab script. These are the parameters that you will need to assign values to:
r1 = Lower value of the parameter r range to explore.
r2 = Upper value of the parameter r range to explore.
N = Number of r-values to use between r1 and r2. Because the Liapunov exponent as a function of r can be very "wiggly", take N large, say: N = 1000, or larger.
x1 = Lower value of x considered.
x2 = Upper value of x considered.
nb = Number of map iterations before the calculation starts. This should be a fairly large number, say nb = 5000, to allow the iterates to settle on the attractor.
np = Number of iterations used to calculate the Liapunov exponent. Again, a fairly large number, to obtain an accurate calculation. Note that the size of the error is, typically, $\boldsymbol{O(1/np)}$**!**
In addition, **you will need two sub-scripts,** y = Fun(r, x) and y = dFun(r, x), which compute the function $\boldsymbol{f(x)}$, and the absolute value of its derivative, $|\mathbf{d}\boldsymbol{f}/\mathbf{d}\boldsymbol{x}|$. **The basic script is then:**

```
r = r1 + (r2 - r1)*(0:N)/N;
x = x1 + (x2 -x1)*rand(size(r)); % This initializes the iteration.
for j=1:nb; x = Fun(r, x); end; % Iterate nb times to approach the attractor.
nf = nb + np;
le = zeros(size(r)); % Will contain the Liapunov exponent for each value in the array r.
for j=(nb+1):nf
    x = Fun(r, x);
    le = le + (1/np)*log(dFun(r, x));
end
```

Now all that remains to do is plot le versus r.

1.2 Answer: Liapunov exponents for 1-D maps #02

The answer to the three parts, and the optional task in § 1.2.1, follows below. Even if you did not attempt to do the optional task, **please read section 1.2.1.**
Remark. The Liapunov exponent is negative within any periodic window (where the attractor is a periodic orbit), and goes to $-\infty$ at super-stable periodic orbits[1] — though lack of numerical resolution cuts these downward "dives"

[1] Orbits including a point with vanishing Map derivative are super-stable: small perturbations to them vanish at super-exponential rate.

to a finite value. *In the chaotic regions the Liapunov exponent is positive.* However *the chaotic regions may contain a fractal structure of periodic windows* (which are very hard to resolve beyond the larger ones). Note also that *at period doubling bifurcations the Liapunov exponent reaches zero, but does not cross to positive.* These features are evident in the plots of the Liapunov exponents below. ♣

1. The Liapunov exponent calculation for the **cosine map** $\boldsymbol{f(x) = r\,\cos(x)}$, with $\boldsymbol{-5 \leq r,\, x \leq 5}$, is shown in **figure** 1.1. Details near the onset of chaos (on the $\boldsymbol{r > 0}$ side) are shown in **figure** 1.2. Here the **left panel**

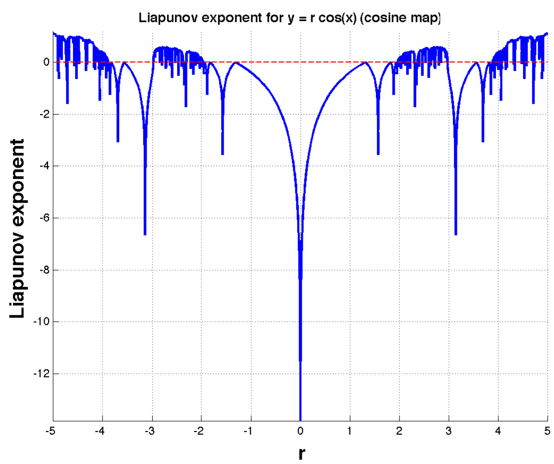

Figure 1.1: Liapunov exponent: cosine map. Liapunov exponent ($\boldsymbol{L_i(r)}$) for the cosine map, in the range: $\boldsymbol{-5 \leq r,\, x \leq 5}$. **Sensitive dependence to initial conditions, and chaos, occurs for $L_i > 0$** (dashed red line). Notice also the multiple "downward spikes", corresponding to super-stable orbits within periodic windows.
Justification of the x-range: *since $|f(x)| \leq r$, the attractor must be contained within $|x| \leq r$.*

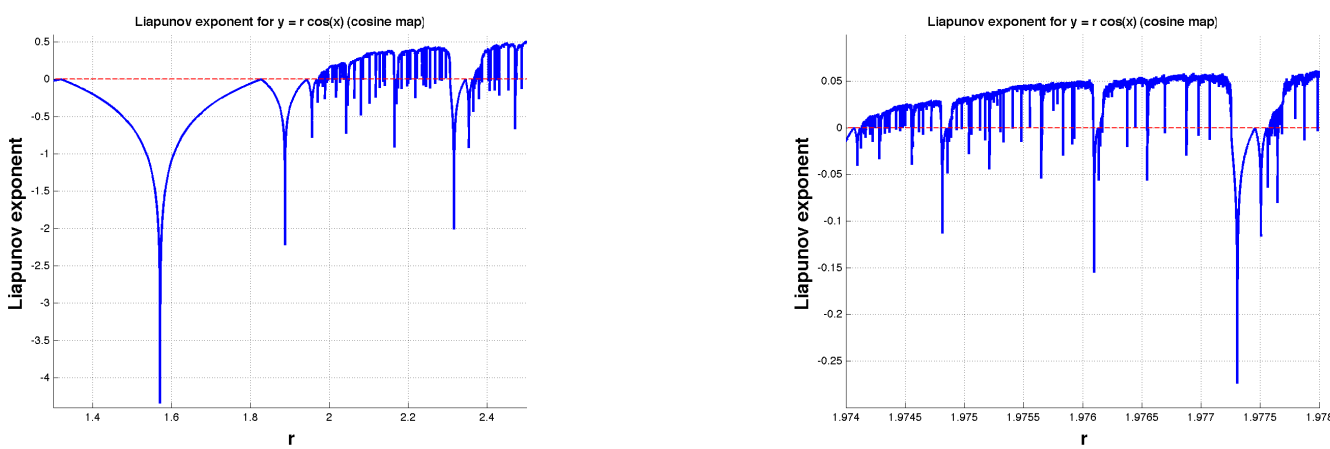

Figure 1.2: Details of the Liapunov exponent near the onset of chaos for the cosine map — see the text for description.

shows the range $\boldsymbol{1.3 < r < 2.5}$, with $\boldsymbol{-4.5 < L_i < 0.5}$ (the $\boldsymbol{L_i = 0}$ level is indicated by the dashed red line). The **right panel** shows the range $\boldsymbol{1.974 < r < 1.978}$, with $\boldsymbol{-0.30 < L_i < 0.1}$. Note that, **as the resolution in r is increased, more periodic windows (downward spikes through $L_i = 0$) become visible.**

2. The Liapunov exponent calculation for the **quartic map** $\boldsymbol{f(x) = r\,(1 - (2\,x - 1)^4)}$, with $\boldsymbol{0 \leq r,\, x \leq 1}$, is shown on the left panel in **figure** 1.3. The right panel shows a detail near the onset of chaos, for $\boldsymbol{0.96 \leq r \leq 0.98}$. As in all cases, the dashed red line indicates the instability threshold $\boldsymbol{L_i = 0}$. **Sensitive dependence on initial conditions, and chaos, occur for $L_i > 0$.** **Figure** 1.4 shows finer detail near the onset of chaos, for $\boldsymbol{0.967 \leq r \leq 0.969}$ on the left panel, and $\boldsymbol{0.9686 \leq r \leq 0.9687}$ on the right panel. As usual, **as the resolution in r is increased, more periodic windows (downward spikes through $L_i = 0$) become visible.**

Justification of the x-range: *It is easy to see that, for* $\boldsymbol{0 \leq x \leq 1}$, $\boldsymbol{0 \leq f(x) \leq r}$. *Hence for* $\boldsymbol{0 \leq r \leq 1}$ *the unit interval is mapped onto itself.*

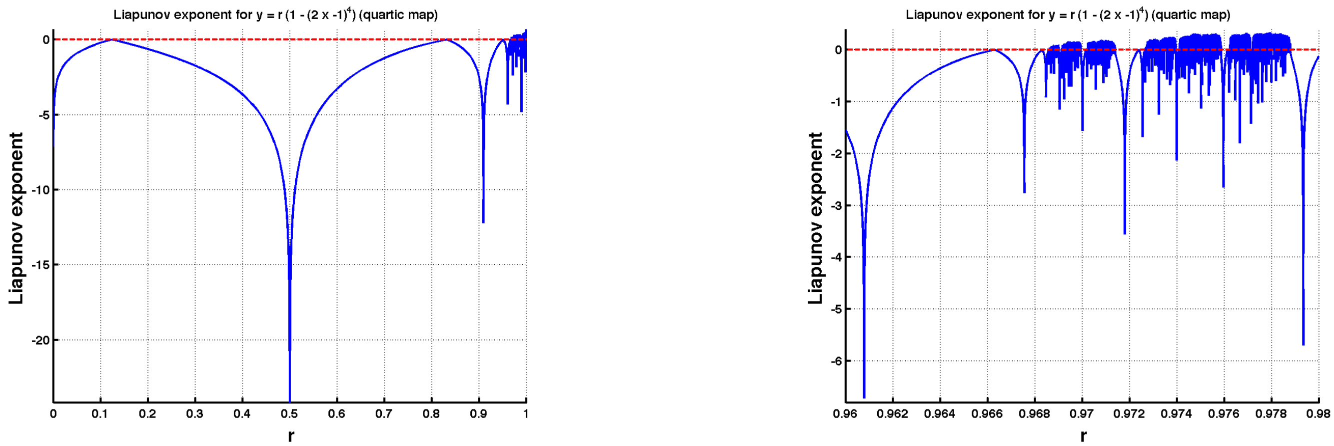

Figure 1.3: Liapunov exponent ($L_i(r)$) for the quartic map and detail — see the text for description.

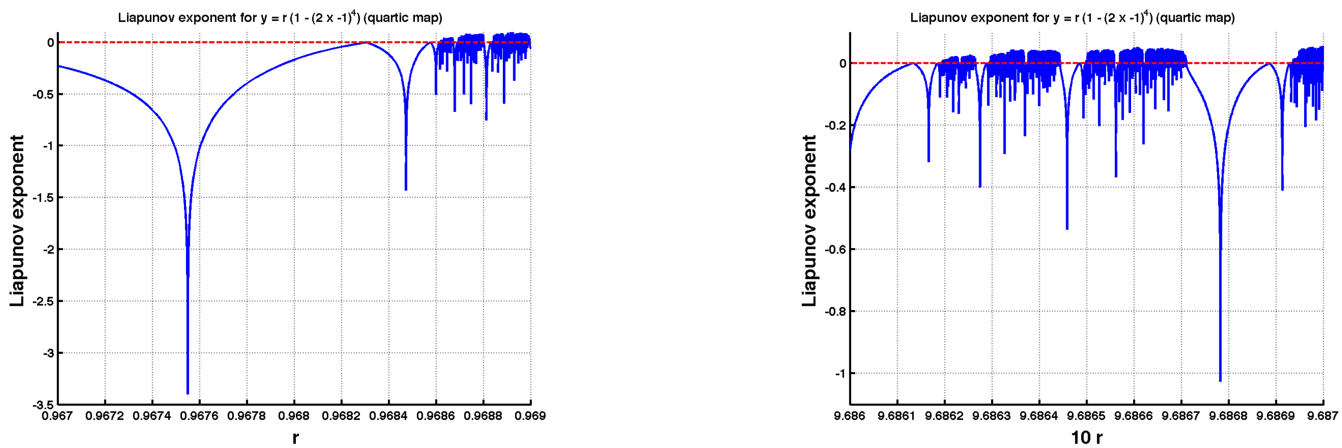

Figure 1.4: Quartic map: Liapunov exponent $L_i(r)$. Details near the onset of chaos — see text for description.

3. The Liapunov exponent calculation for the **cusp075 map** $f(x) = r\,(1 - |2\,x - 1|^{0.75})$, with $0 \le r,\ x \le 1$, is shown on the left panel in **figure** 1.5. The other panels shows a detail near the onset of chaos, for $0.52 \le r \le 0.70$. As in all cases, the dashed red line indicates the instability threshold $L_i = 0$. **Sensitive dependence on initial conditions, and chaos, occur for $L_i > 0$.** Note that the two right panels in **Figure** 1.5 correspond to the same range in r; these are **two runs of exactly the same program, but the pictures differ; why?** This is the **question posed in the optional task,** for the answer see § 1.2.1.

Justification of the x-range: *It is easy to see that, for* $0 \le x \le 1,\ \ 0 \le f(x) \le r$. *Hence for* $0 \le r \le 1$ *the unit interval is mapped onto itself.*

1.2.1 Optional task

An (incorrect) "obvious" answer to why the two right panels in **figure** 1.5 are not the same is: this is because the program computing them[2] starts each sequence with a random x_0. As we will see, there is some truth to this, but *this cannot be the full answer* — if it was, we would see the same phenomena with all the Liapunov exponent calculations, and we do not! The idea is that, **even though we start with a random x_0, we compute the sequence**

[2] See sample/sketch MatLab script in the problem statement.

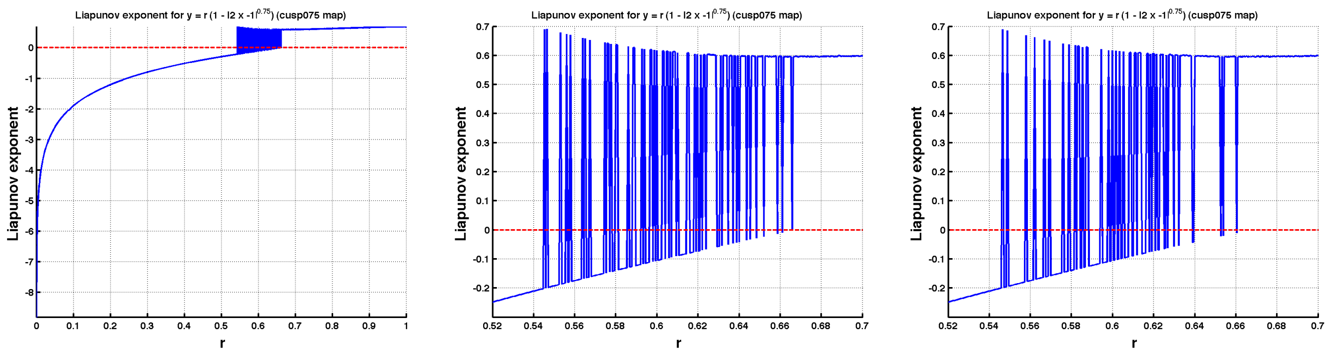

Figure 1.5: Liapunov exponent ($L_i(r)$) for the cusp075 map and details — see the text for description.

for many x_n, so it settles on the attractor; and only then we begin the calculation of the Liapunov exponent. The **key point** here is the phrasing *"the attractor"*, for **what if there is more than one attractor?** *In this case the sequence used to compute the Liapunov exponent would settle, randomly, on one of the possible attractors, giving rise to the behavior observed in* **figure** 1.5. To check this hypothesis, we compute an orbit diagram for the cusp075 map, shown **figure** 1.6. The **left panel** shows the full orbit diagram, for $0 \leq r \leq 1$, while the two **right panels** show a

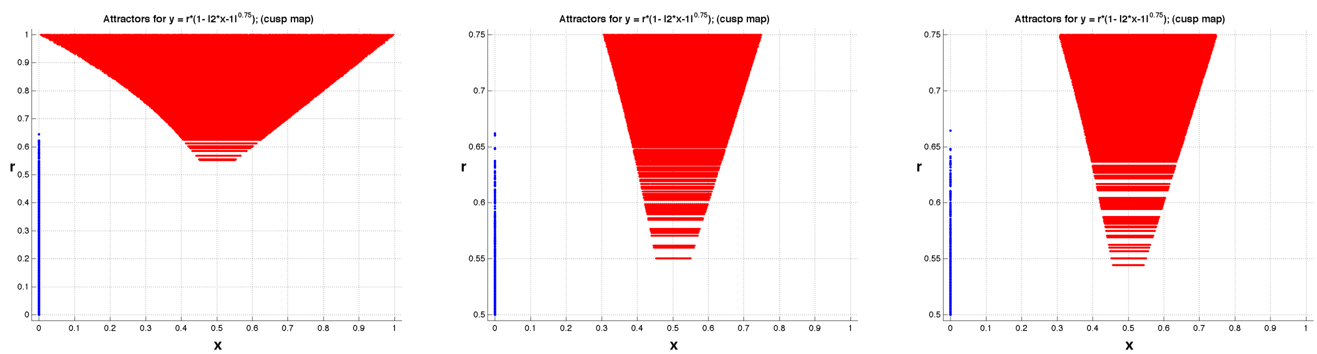

Figure 1.6: Orbit diagram for the cusp075 map, $f(x) = r\,(1 - |2\,x - 1|^{3/4})$, and detail — see text for description.

detail for $0.50 < r < 0.75$ (<u>same</u> detail). From this figure we can see that

a. There are two attractors. In blue: the fixed point $x = 0$, stable for $0 \leq r < 2/3$. In red: a chaotic attractor, which exists for $r_c < r \leq 1$, some $0 < r_c \approx 0.54368897 < 2/3$.

b. The two attractors co-exist for $r_c < r < 2/3$. A sequence starting at a random x_0 may end up at one or the other. This is why the two right panels in **figures** 1.5 **and** 1.6 differ, even though they are for the same parameters.

To double check these conclusions, we recompute the two right panels in **figure** 1.5 (i.e.: Liapunov exponent for $0.52 \leq r \leq 0.70$), but instead of starting the sequences from a random x_0, we start them from: (i) x_0 very close to $1/2$ — left panel in **figure** 1.7**;** and (ii) x_0 very small — right panel in **figure** 1.7**.** As expected, the "oscillations" that appear in **figure** 1.5 vanish. Furthermore, notice that

c. When x_0 is close to $1/2$ (left panel in **figure** 1.7) the sequence is attracted to the fixed point at the origin for $r < r_c$ (negative Liapunov exponent) and to the "red" attractor for $r > r_c$ (positive Liapunov exponent).

b. When x_0 is small (right panel in **figure** 1.7) the sequence is attracted to the fixed point at the origin for $r < 2/3$ (negative Liapunov exponent) and to the "red" attractor for $r > 2/3$ (positive Liapunov exponent).

c. The red attractor is chaotic. This follow because it has a positive Liapunov exponent, so that it exhibits **sensitivity to initial conditions, while being non-trivial.**

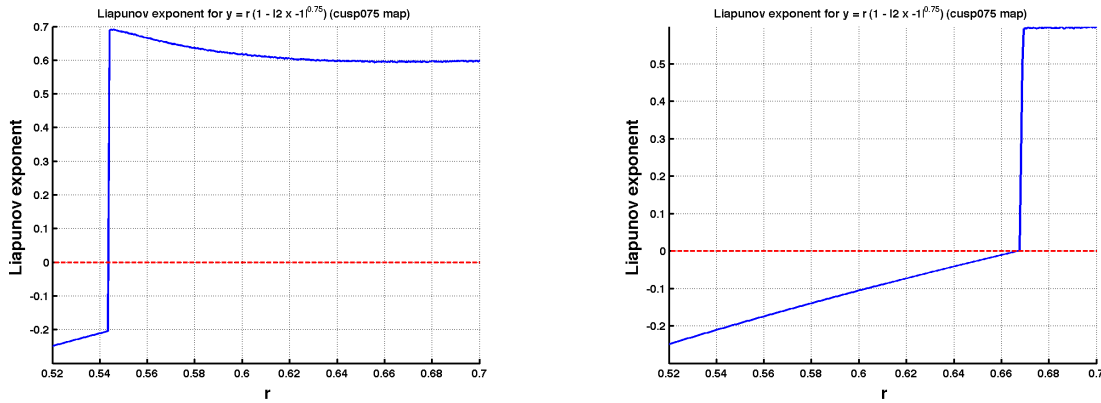

Figure 1.7: Cusp075 map: Liapunov exponent $L_i(r)$. Details near the onset of chaos — see text for description.

b. Note that **we observe neither period doubling, nor any periodic windows, for the cusp075 map.** The Liapunov exponent for the red attractor behaves pretty smoothly for $r > r_c$. **The transition to chaos for the cusp075 map seems to be very different from the one that occurs for the Logistic map.** The red attractor shows up, suddenly, "fully formed" and with $O(1)$ size for $r > r_c$. Interestingly, **this attractor does not seem to be fractal, and "fills in" a full interval**[3] **in x for each value $r > r_c$**,† **In particular: it should have box and correlation dimension = 1.**

† I only have partial analysis and numerical evidence for this. I cannot prove it, and I have not found a reference where it is proved.

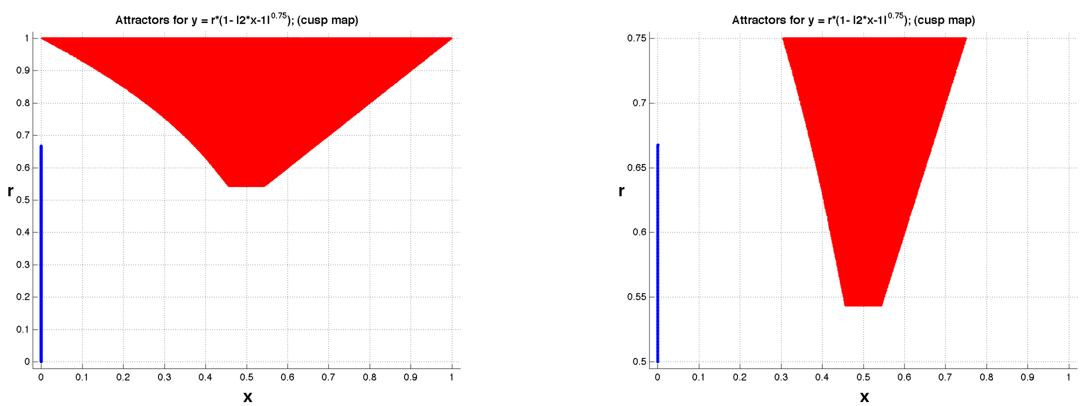

Figure 1.8: Orbit diagram for the cusp075 map, and detail — see text for description.

Finally we recompute the orbit diagram for the cusp075 map, for the same parameter ranges, $0 < r < 1$ and $0.5 < r < 0.75$, as in **figure** 1.6. However, instead of a random initial x_0 for each r, we do two computations: one starting near zero and another starting near $1/2$. The result is shown in **figure** 1.8. Notice that the "gaps" in **figure** 1.6 are gone. **The pictures illustrate how the red attractor pops up "out of nowhere" for $r \geq r_c \approx 0.54368897$.**

---

[3] Except for, at most, a discrete set of points.

2 Newton's method in the complex plane #01

2.1 Statement: Newton's method in the complex plane #01

Suppose that you want to solve an equation, $g(x) = 0$. Then you can use *Newton's method*, which is as follows: Assume that you have a "reasonable" guess, x_0, for the value of a root. Then the sequence $x_{n+1} = f(x_n)$, $n \geq 0$, where

$$f(x) = x - \frac{g(x)}{g'(x)}, \tag{2.1}$$

converges (very fast) to the root.

Remark 2.1 (The idea). Assume an approximate solution $g(x_a) \approx 0$. Then write $x_b = x_a + \delta x$ to improve it, where δx is small. Then $0 = g(x_a + \delta x) \approx g(x_a) + g'(x_a)\,\delta x \;\Rightarrow\; \delta\,x \approx -\frac{g(x_a)}{g'(x_a)}$, and (2.1) follows.
Of course, if x_0 is not close to a root, the method may not converge. Even if it converges, it may converge to a root that is far away from x_0, not necessarily the closest root. In this problem **we investigate the behavior of Newton's method in the complex plane, for arbitrary starting points.** ♣

Consider iterations of the map generated by Newton's method for the roots of $z^3 - 1 = 0$. i.e.:

$$z_{n+1} = f(z_n) = \left(\frac{2}{3} + \frac{1}{3\,z_n^3}\right) z_n, \quad n \geq 0, \tag{2.2}$$

where $0 < |z_0| < \infty$ is arbitrary, and the z_n are **complex numbers.**
Note that

$$\zeta_1 = 1, \quad \zeta_2 = e^{i\,2\,\pi/3} = \frac{1}{2}(-1 + i\,\sqrt{3}), \quad \text{and} \quad \zeta_3 = e^{i\,4\,\pi/3} = \frac{1}{2}(-1 - i\,\sqrt{3}), \tag{2.3}$$

are the roots of $z^3 = 1$.

Your tasks: Write a computer program to calculate the orbits $\{z_n\}_{n=0}^{\infty}$. Then, for every initial point z_0, draw a colored dot at the position of z_0, where **the colors are picked as follows:**

$$z_n \to \zeta_1,\ \textbf{green.} \qquad z_n \to \zeta_2,\ \textbf{red.} \qquad z_n \to \zeta_3,\ \textbf{blue.} \qquad \textbf{No convergence, black.} \tag{2.4}$$

What do you see? Do blow ups of the limit regions between zones.

Hints and practical numerical considerations.

h1. Divide the region where the initial data z_0 will be picked [I suggest the square $-2 \leq \text{Re}(z_0),\ \text{Im}(z_0) \leq 2$] into pixels, then pick a z_0 at the center of each pixel, and color the pixel according to (2.4).

h2. If you use MatLab, **do not plot points.** As suggested in item **h1** plot pixels — use the command image(x, y, C) to plot, where $x = \text{Re}(z_0)$ and $y = \text{Im}(z_0)$. **Why?** Because using points leaves a lot of unpainted space in the figure, and **gives huge file sizes** if you use enough pixels to get a good picture.

h3. Deciding convergence. Deciding that the sequence converges is easy: once z_n gets "close enough" to one of the roots, then the very design of Newton's method guarantees convergence. Thus, given a z_0, compute z_N for some large N, and check if $|z_N - \zeta_j| < \delta$ for one of the roots and some "small" tolerance δ — which does not have to be very small, in fact $\delta = 0.25$ is good enough. If this criteria is not satisfied for any of the roots, then classify the sequence starting at z_0 as "non-convergent".

You can get reasonable pictures with $N = 50$ iterations on a 150×150 grid — a larger N is needed when refining near the boundary between zones. For the answer I used a 500×500 grid and $N = 100$ iterations — which I increased to $N = 200$ and $N = 300$ for the blow ups of details.

h4. Compute in parallel. If you use MatLab, make sure to do all the sequences (one for each pixel) in parallel, using vector/matrix operations. This is much faster than a "for loop".

h5. Avoid division by zero. Note that (2.2) ceases to make sense if $z_n = 0$ — classify this as non-convergence. This can cause a problem if you are computing all the sequences in parallel, because this requires all of them to be computed from z_0 to z_N. One way to get around this (in MatLab) is as follows: Place all the iterates in a complex matrix **Zn**, where the entry $(p,\ q)$ corresponds to z_n for the sequence starting in the $(p,\ q)$ pixel. Then, before computing the next iterate, execute: **Zn = Zn + del*(Zn == 0)**, where **del = 1e-30**.[†] After this sequences with $\boldsymbol{z_n} = \mathbf{0}$ will produce a very large $\boldsymbol{z_{n+1}}$, which is guaranteed not return to the vicinity of the roots $\boldsymbol{\zeta_j}$ for many iterations (more than 300), resulting in "effective" non-convergence. ‡

† This replaces zero entries in **Zn** by **del**, because the logical operator **(Zn == 0)** yields zero for all non-zero entries in **Zn**, and one for zero entries.

‡ The result will be $\boldsymbol{z_{n+1}} \approx \mathbf{(1/3)10^{60}}$, while for $\boldsymbol{z_n}$ large (2.2) reduces to $\boldsymbol{z_{n+1}} \approx \mathbf{(2/3)}\,\boldsymbol{z_n}$. Hence returning to $\boldsymbol{z_{n+M}} = \boldsymbol{O}\mathbf{(1)}$ requires, roughly, $\mathbf{(2/3)}^{\boldsymbol{M}}\,\mathbf{10^{60}} = \boldsymbol{O}\mathbf{(1)}$.

2.2 Answer: Newton's method in the complex plane #01

Figure 2.1 show the results of our calculations. Note the *fractal nature of the boundary between the basins of attraction*

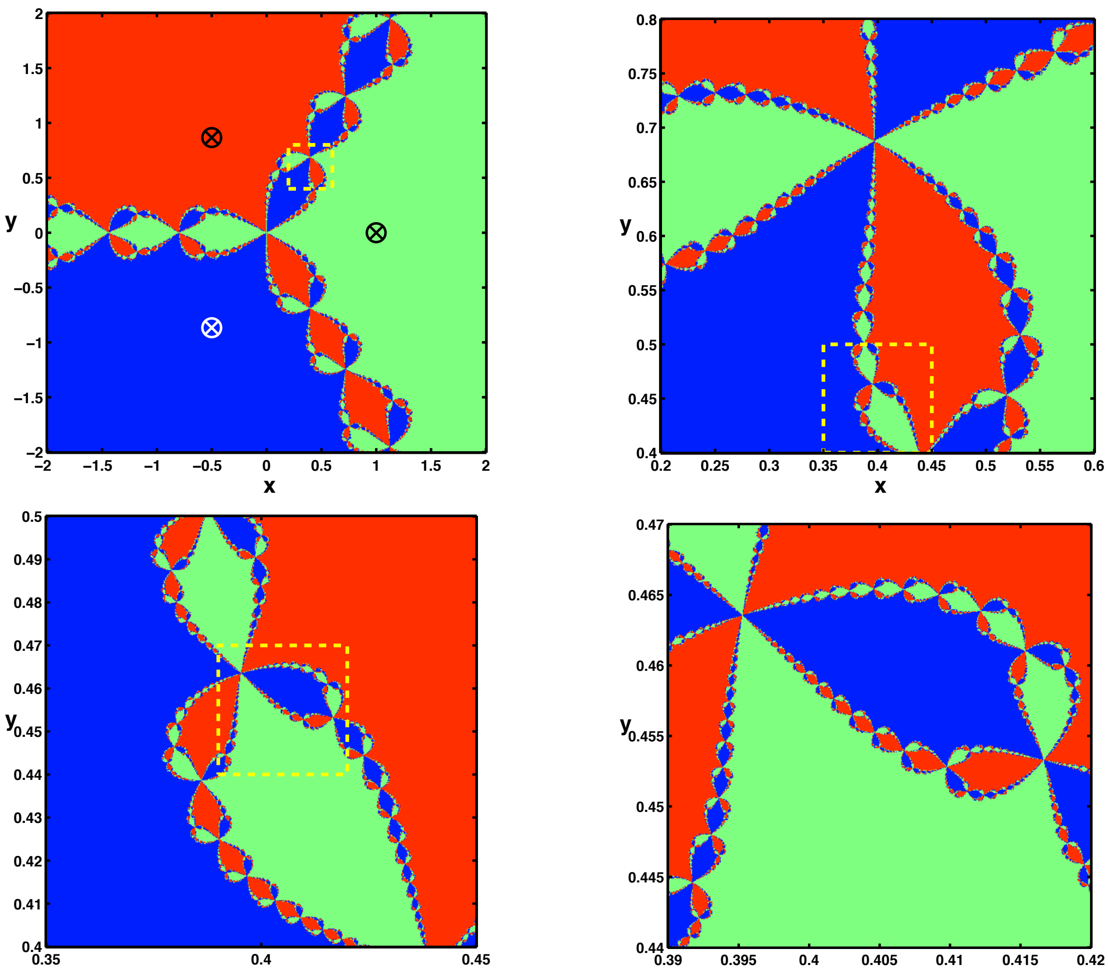

Figure 2.1: (Newton's method in the complex plane #01). Convergence zones for the $z^3 = 1$ Newton's map iterates, using the color scheme in (2.4), with a 500×500 pixel grid. Left to right and top to bottom: (a) $N = 100$ iterations, for $-2 < x,\ y < 2$. The crosses are the roots ζ_j. (b) $N = 200$ iterations, for $0.2 < x < 0.6$ and $0.4 < y < 0.8$. (c) $\boldsymbol{N} = \mathbf{300}$ iterations, for $0.35 < x < 0.45$ and $0.4 < y < 0.5$. (d) $\boldsymbol{N} = \mathbf{300}$ iterations, for $0.39 < x < 0.42$ and $0.44 < y < 0.47$. In (a–c) **the square with a yellow boundary** indicates the region displayed in the next picture.

for each root: as we zoom in, the object appears as a smaller (but distorted) copy of itself. Non-trivial self-similarity [4]

[4] A line in the plane is also self-similar, but it has trivial structure.

is the hallmark of a fractal. Sets like this (boundaries between convergence regions of complex analytic iterations) are called **Julia sets**.

> The attracting basins are **Fatou sets**. The sets are named after Gaston Julia and Pierre Fatou, two mathematicians that pioneered the study of complex dynamics — e.g., see: G. Julia, *Mémoire sur l'iteration des fonctions rationnelles,* Journal de Mathématiques Pures et Appliquées, **8:** 47–245, 1918, and P. Fatou, *Sur les substitutions rationnelles*, Comptes Rendus de l'Académie des Sciences de Paris, **164:** 806–808 and vol. 165, pp. 992–995, (1917).

The orbits within the Julia set are chaotic. These orbits are, generally, not periodic (but recurrent), and small differences in z_n grow exponentially with n (sensitive dependence on initial conditions). However, **computing these orbits is extremely hard,** as perturbations out of the Julia set make the resulting orbit convergent.

THE END.