計算機実習 問題 6.1-周期倍化

早稲田大学先進理工学部物理学科 B4 藤本將太郎

2014/04/28

1 シミュレーションの目的

生物の個体数の時間発展は、その世代交代がある決まった季節に起こるなど、離散的であると見なせるため、これを記述するのには微分方程式よりもむしろ差分方程式の方が適している。世代 (n+1) の個体数を世代 n の個体数に関係づけ、その成長率が密度に依存しない一番簡単なモデルを考えると、世代 n の個体数を P_n、成長率を α(定数) として

$$P_{n+1} = \alpha P_n \tag{1}$$

と書ける。これは大変簡単ではあるが、$\alpha > 1$ のときに個体数が等比級数的に無限に増加してしまうことと、現実には個体数はその環境に制限されていることとから、現象を的確に表したモデルとは言えない。そこで次に、成長率が個体数が増加するにつれて線形に減少する場合

$$P_{n+1} = P_n(a - bP_n) \tag{2}$$

を考える。このときこの式は P_n の 2 乗を含んでいるので、非線形の方程式となる。$P_n = (a/b)x_n$ と置くことで個体数のスケールを変換し、

$$x_{n+1} = ax_n \tag{3}$$

として P_n を x_n に置き換えると、いろいろなパラメータを定義している単位系も変換される。ここでパラメータ $r = a/4$ を導入して、再び書き直すと

$$x_{n+1} = f(x_n) = 4rx_n(1 - x_n) \tag{4}$$

となる。これは単一のパラメータ r のみでその動的な性質が決定されるという意味で望ましいモデルとなっている。ただし、$x_n > 1$ のとき x_{n+1} は負になるので、このような非物理的性質が現れないように、x と r を区間 $0 \leq x \leq 1$、$0 \leq r \leq 1$ に制限することにする。

式 (4) で定義される関数 f は 1 次元の写像であり、この形はロジスティック写像として知られているものである。ロジスティック写像は、系の未来の状態が決定論的に与えられるという意味で、単純な力学系の例である。本シミュレーションでは式 (4) の差分方程式を、世代ごとに計算を繰り返す方法で実際に求め、ロジスティック写像の動的な性質を調べることにする。

2 作成したプログラム

ロジスティック写像の性質を調べるために、Python を使用したプログラムを以下に示す。

```
#! usr/bin/env python
# coding:utf-8
#
""" 計算機実習
問題 6.1　周期倍化
プログラムを実行すると、パラメータの設定ダイアログが開き、
OK ボタンを押すとグラフが描画される。

作成者：藤本將太郎
"""

import array as array
import matplotlib.pylab as plt # 追加で必要なモジュール
from Tkinter import *

class Iterate_map():
   def __init__(self):
      self.show_setting_window()

   def judge(self):
      self.x0=float(self.entry[0].get())
      self.r=float(self.entry[1].get())
      if self.x0 < 0 or self.x0 > 1 or self.r < 0 or self.r > 1:
         if self.x0 < 0 or self.x0 > 1:
            toplevel=Toplevel()
            s="条件: 0 <= x0 <= 1 を満たしていません。"
            label=Label(toplevel, text=s).pack(padx=10, pady=5)
            if self.r < 0 or self.r > 1:
               s="条件: 0 <= r <= 1 を満たしていません。"
               label=Label(toplevel, text=s).pack()
         toplevel.grab_set()
         button=Button(toplevel, text='OK', command=self.close)
         button.pack(pady=5)
      else: self.mapping()
   def close(self):
```

```
36            toplevel.destroy()
37
38        def show_setting_window(self):
39            parameters=['x0', 'r', 'nmax'] # 設定するパラメータ
40            root=Tk()
41            frame=Frame(root, padx=5, pady=5)
42            frame.pack()
43            label=Label(frame, text='setting parameter')
44            label.grid(row=0, column=0, columnspan=2)
45            self.entry=[]
46            for i, parameter in enumerate(parameters):
47                label=Label(frame, text=parameter + ' = ')
48                label.grid(row=i+1, column=0, sticky=E)
49                self.entry.append(Entry(frame, width=10))
50                self.entry[i].grid(row=i+1, column=1)
51            self.entry[0].focus_set()
52            button=Button(frame, text='OK', command=self.judge)
53            button.grid(row=4, column=0, columnspan=2, sticky=E+W)
54            button=Button(frame, text='add attractor', command=self.drawing)
55            button.grid(row=5, column=0, columnspan=2, sticky=E+W)
56            root.mainloop()
57
58        def mapping(self): # グラフの描画
59            self.nmax=int(self.entry[2].get())
60            x=array.array('d')
61            x.append(self.x0)
62            for i in range(0, self.nmax):
63                t = 4.0*self.r*x[i]*(1-x[i])
64                x.append(t)
65            n=range(self.nmax + 1)
66            plt.plot(n, x,
67                     label= r'$x_{0}\ =$' + str(self.x0) + ' : '
68                          + r'$r\ =$' + str(self.r) + ' : '
69                          + r'$n_{\mathrm{max}}\ =$' + str(self.nmax)
70                     )
71            plt.gca().set_xlim(0,self.nmax)
72            plt.gca().set_ylim(-0.05,1.0)
73            plt.xlabel(r'$n$', fontsize=16)
74            plt.ylabel(r'$x$', fontsize=16)
75            plt.title('Iterate_map')
```

```
76          plt.legend(loc="best")
77          plt.show()
78
79      def drawing(self):
80          if self.r > 1.0/4.0 and self.r < 3.0/4.0:
81              xfi=1.0-1.0/(4.0*self.r)
82              plt.plot([0, self.nmax], [xfi]*2,
83                       label=r'$x = 1 - 1 / 4r = $' + str(xfi)
84                       )
85              plt.legend(loc="best")
86              plt.show()
87          else: pass
88
89  if __name__ == '__main__':
90      run=Iterate_map()
91
```

このプログラムは、追加のモジュールとして、グラフ描画に便利な matplotlib を使用している。プログラムを実行すると各パラメータ {x0, r, nmax}(x0: x の初期値 x_0、r: 制御パラメータ、nmax: 演算する世代数) の設定ダイアログが開き、OK ボタンを押すことでその値で計算されたロジスティック写像のグラフが自動的に描画される。

3 実習課題

a. $r = 0.24$ とおいてさまざまな初期値 x_0 について動的な振る舞いを調べよ。$x = 0$ が安定な固定点であることを示せ。r の値が十分に小さいときには、計算を反復して求めた x の値が、初期値 x_0 とは無関係に $x = 0$ に収束する。x が昆虫の個体数を表すと考えて、個体数の定性的な振る舞いについて調べよ。

$r = 0.24$ とおいて $x_0 = 0, 0.2, 0.4, 0.6, 0.8, 1.0$ とした時のグラフを図 1 に示した。このグラフから確かめられるように、r の値が十分小さい時には、初期値によらず x の値は $x = 0$ に単調に収束する。x が昆虫の個体数を表すと見るなら、r の値が小さいということは、個体数の増加率が 1 よりも小さくなる状態を表しており、生まれてくる個体よりも死ぬ個体のほうが多い状態とも言えるだろう。そのような状況では、個体数は減り続け、いずれ 0 になる。

b. 式 (4) の動的な性質を $r = 0.26, 0.5, 0.74, 0.748$ について調べよ。$r > 0.25$ では $x = 0$ が不安定な固定点であることを確かめよ。与えられた r の値では、計算の反復で得られる x の値は初期値の過渡点を過ぎると変化しないこと、つまり、長時間後の動的な振る舞いは周期 1 であることを示せ。$r < 3/4$ について区間 $0 < x_0 < 1$ の x_0 を初期値とするすべての軌跡が $x = 1 - 1/4r$ のアトラクタに近づくことが示されている。軌跡がアトラクタに吸い込まれる初期値の集合をそのアトラクタの吸引域と呼ぶ。ロジスティック写像では、領域 $0 < x_0 < 1$ は $x = 1 - 1/4r$ のアトラクタの吸引域である。

$r = 0.26, 0.5, 0.74, 0.748$ とおいたとき、さまざまな x_0 を初期値として設定しグラフを描いた。これらを図 2、図 3、図 4、図 5 に示す。これらの結果から、この範囲の r では x の値は、振動しな

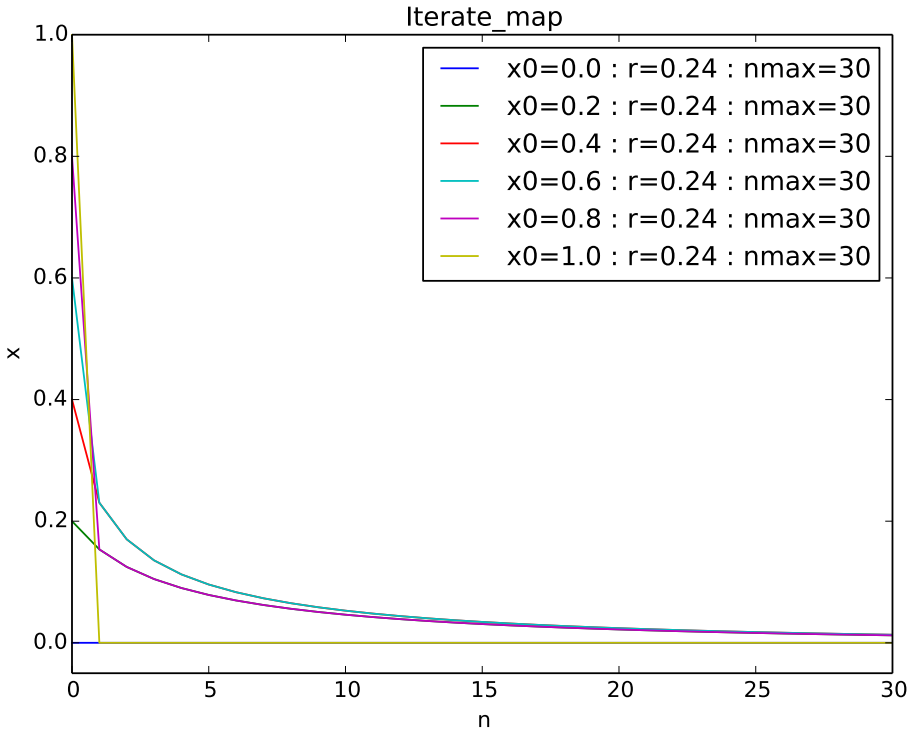

図 1　$r = 0.24$ とおいて $x_0 = 0, 0.2, 0.4, 0.6, 0.8, 1.0$ とした時のグラフ

がら一定の値に収束していき、長時間後にはその動的な振る舞いが周期 1 であることがわかる。また、その時収束する値は $1 - 1/4r$ と等しくなっていることがわかる。

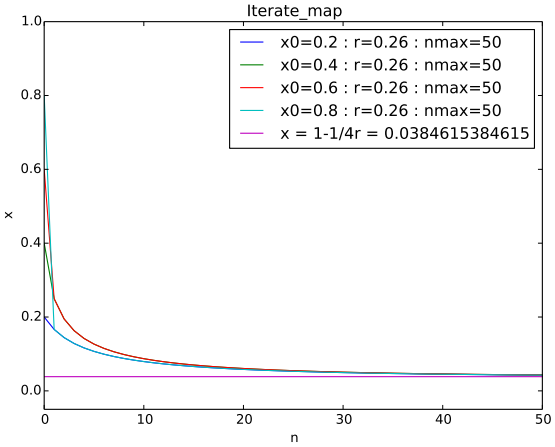

図 2　$r = 0.26$ で $x_0 = 0.2, 0.4, 0.6, 0.8$ とした時のグラフ

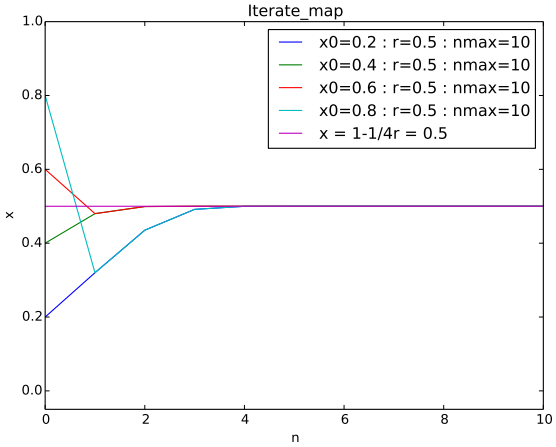

図 3　$r = 0.5$ で $x_0 = 0.2, 0.4, 0.6, 0.8$ とした時のグラフ

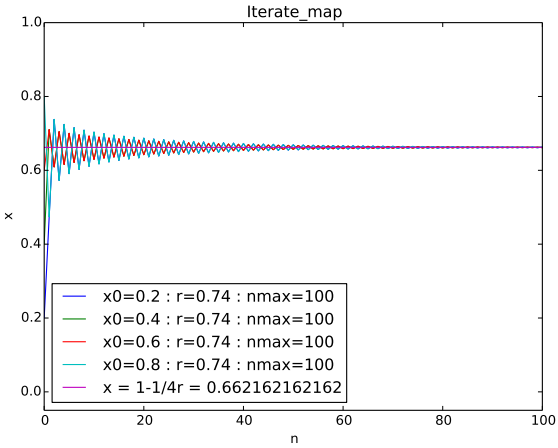

図 4　$r = 0.74$ で $x_0 = 0.2, 0.4, 0.6, 0.8$ とした時のグラフ

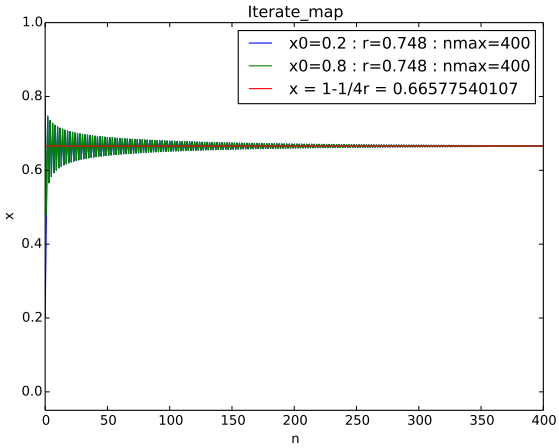

図 5　$r = 0.748$ で $x_0 = 0.2, 0.8$ とした時のグラフ

c. $r = 0.752, 0.76, 0.8, 0.862$ について式 (4) の動的な性質を調べよ。r を 0.75 よりほんの少し大きくすると、初期値の過渡的な振る舞いの後に、2 つの数値の間を振動することを示せ。つまり、単一の固定点に対応する周期 1 の安定なサイクルの代わりに、系は周期 2 の安定なサイクルを持つようになる。1 つの固定点 x^* が 2 つの x_1^* と x_2^* に分かれる。つまり、分岐する r の値は $r = b_1 = 3/4$ である。1 対の $x(x_1^*$ と $x_2^*)$ は 1 つの、周期 2 の安定なアトラクタを形成する。

$r = 0.752, 0.76, 0.8, 0.862$ について x_0 をさまざまに変えたときのグラフを図 6、図 7、図 8、図 9 に示す。これらのグラフから見て取れるように、十分時間が経過した後も、x の値は一つの値に収束することなく、二つの値の間を周期 2 で振動していることがわかる。

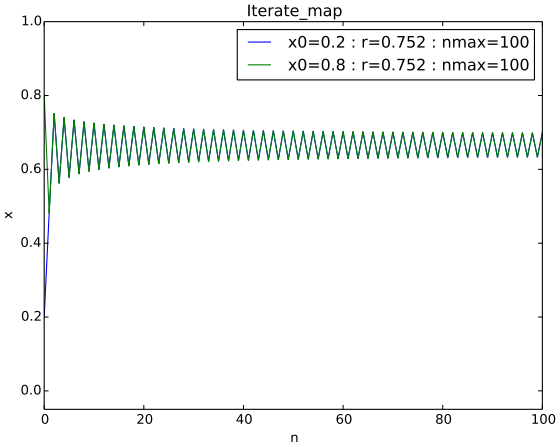

図 6　$r = 0.752$ で $x_0 = 0.2, 0.8$ とした時のグラフ

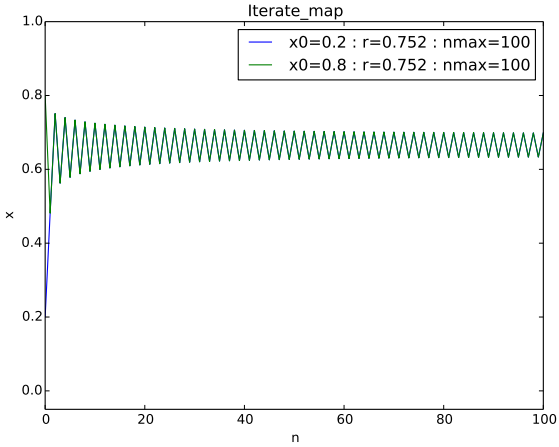

図 7　$r = 0.76$ で $x_0 = 0.2, 0.8$ とした時のグラフ

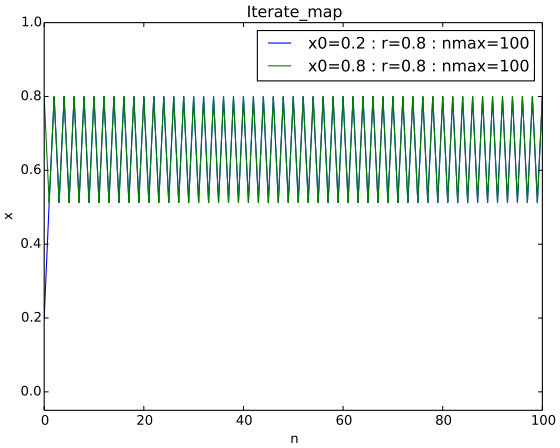

図 8　$r = 0.8$ で $x_0 = 0.2, 0.8$ とした時のグラフ

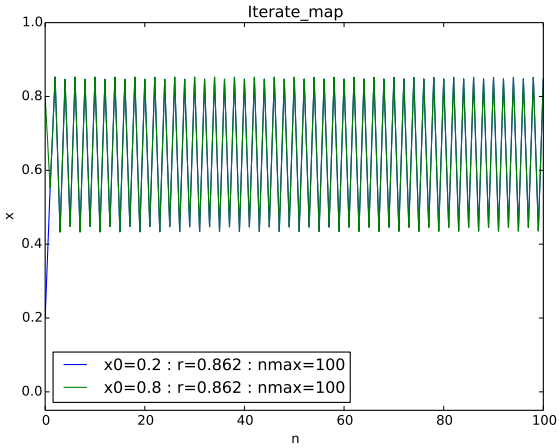

図 9　$r = 0.862$ で $x_0 = 0.2, 0.8$ とした時のグラフ

d. 設問 c で観測したものと同様な動的振る舞いを示す、昆虫の個体数の生態学的なシナリオについて述べよ。

設問 c で観測されたような動的な振る舞いは、ある昆虫が、限られた環境の中で成長率が十分にあるために個体数が増えていき、しかし一方で個体数の増加による各種資源の枯渇など、成長率への負の影響のために、個体数を増やし続けることができず、次の世代ではその数が減少してしまうような場合であると考えることができる。

e. $r = 0.863, 0.88$ では、式 (4) の安定なアトラクタとその周期はどうなるか。

$r = 0.863, 0.88$ で $x_0 = 0.2$ とした時のグラフを図 10、図 11 に示す。これらのグラフを注意深く見ると、設問 c のように周期 2 で振動しているのではなく、その二つのアトラクタがさらに二つのアトラクタに分岐していることがわかる。つまり、十分時間が経ったのちに、x は周期 4 で振動している。

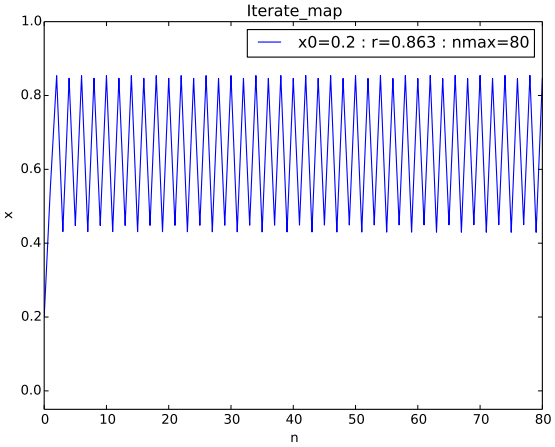

図 10　$r = 0.863$ で $x_0 = 0.2$ とした時のグラフ

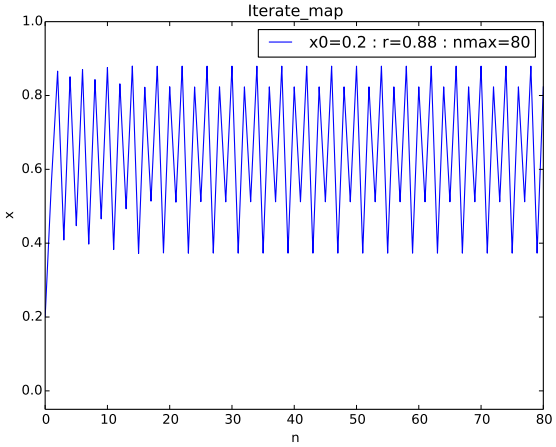

図 11　$r = 0.88$ で $x_0 = 0.2$ とした時のグラフ

f. $r = 0.89, 0.891, 0.8922$ ではアトラクタとその周期はどうなるか。

$r = 0.89, 0.891, 0.8922$ で $x_0 = 0.2$ とした時のグラフを図 12、図 13、図 14 に示す。これらのグラフは、設問 e でのアトラクタがそれぞれさらに二つのアトラクタに分岐しているものであることがわかる。つまり、十分時間が経ったのちに、x は周期 8 で振動している。以上の結果から、特に観測した r の領域については、r の値が増加するにつれて、アトラクタの数、または振動の周期が倍化していることがわかる。

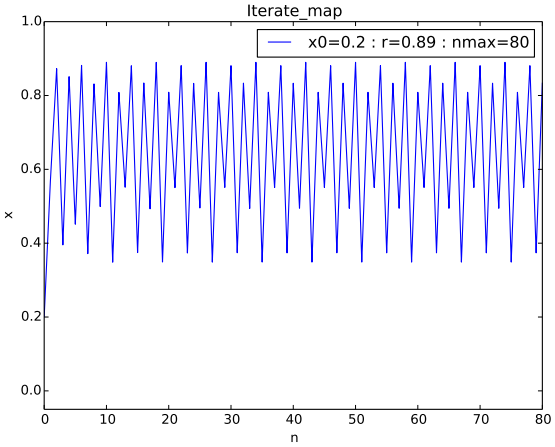

図 12　$r = 0.89$ で $x_0 = 0.2$ とした時のグラフ

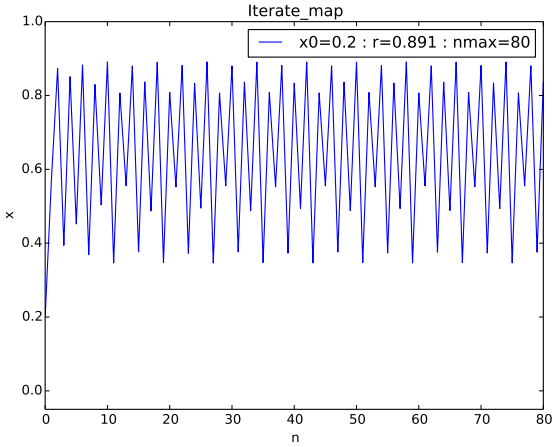

図 13　$r = 0.891$ で $x_0 = 0.2$ とした時のグラフ

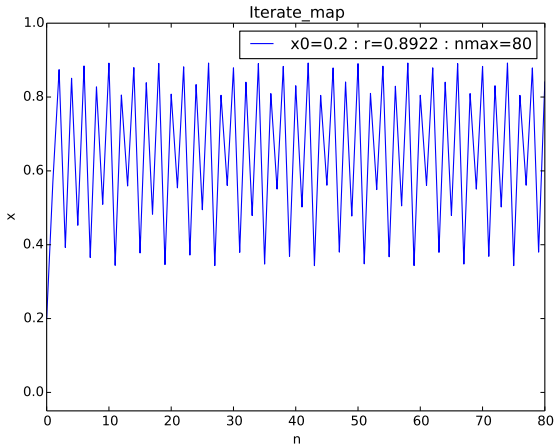

図 14　$r = 0.5$ で $x_0 = 0.2, 0.4, 0.6, 0.8$ とした時のグラフ

4　まとめ

ロジスティック写像の性質である周期倍化を、コンピューターシミュレーションによって確認することができた。簡単なモデルから作られた古典的な方程式からカオス的な挙動が生まれることは興味深い。

5　参考文献

- ハーベイ・ゴールド, ジャン・トボチニク, 石川正勝・宮島佐介訳『計算物理学入門』, ピアソン・エデュケーション, 2000.