In [0]:

```python
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import math
```

In [0]:

```python
def new_plot(title='', y_min=-8, y_max=8):
    fig, ax = plt.subplots(figsize=(10,5))

    ax.set(xlabel='alpha', ylabel='x',
           title=title)
    ax.grid()

    return ax


def draw_line(ax, foo, x_min=0, x_max=2, step=0.01, style='b', label=''):
    t = np.arange(x_min, x_max, step)
    s = foo(t)

    ax.plot(t, s, style, label=label)


def draw_asym(point, label=''):
    y = list(range(-8, 8))
    x = [point] * len(y)

    ax.plot(x, y, '--', label=label, linewidth=0.6)



def draw_tree(ax, f, x_min, x_max, style, step=0.01, iter=100, inf=10, inf_n=1
00, eps=10e-6, start=0):

    a = np.arange(x_min, x_max, step)

    a_ = []
    x_ = []


    for i in a:

        x_tmp = start

        for j in range(iter):
            x_tmp = f(x_tmp, i)

    #     print(i, x_tmp)

        if not (-inf < x_tmp < inf):
            continue
```

```
        tmp = []

        for n in range(1, inf_n):
            tmp.append(x_tmp)
            x_tmp = f(x_tmp, i)

        a_.append(i)
        x_.append(tmp)


    ax.plot(a_, x_, style, ms=0.5)


def show_plot(y_min=-10, y_max=10):
    plt.ylim(y_min, y_max)     # set the ylim to bottom, top
    plt.legend()
    plt.show()
```

Лабораторная #1

Даны следующие отображения:

1. $x_{n+1} = 1 - \alpha x_n^2$
2. $x_{n+1} = \alpha x_n(1 - x_n)$

Небходимо выполнить для каждого:

- Найти α, при которых возникает бифуркация: $\mu = 1, \ \mu = -1$;
- Построить дерево бифуркаций;
- Построить графики $x_1^*(a), \ x_2 * (a)$;
- Объяснить за счет чего возникает странный аттрактор и как он разрушается; *Свести отображение 2 к отображению 1;

Теория

Неподвижные точки

Неподвижной точкой отображения $f : X \to X$ называется такая точка $x \in X$, которую заданное отображение переводит в неё же, иными словами, неподвижная точка удовлетворяет уравнению $f(x) = x$.

Мультипликатор

Неподвижные точки могут быть *устойчивыми* и *неустойчивыми*. Данная характеристика описывает поведение точек в окрестности данной неподвижной точки. Характер устойчивости можно определить с помощью специальной характеристики – *мультипликатора*, представляющей собой производную функции, вычисленную в неподвижной точке. Действительно, в малой окрестности неподвижной точки x_0 некоторого отображения f можно считать, что $x_{n+1} = x_0 + \tilde{x}_{n+1}$ и $x_n = x_0 + \tilde{x}_n$, где $\tilde{x}_{n+1}$ и $\tilde{x}_n$ малые добавки к x_0.

Тогда, из уравнения неподвижной точки имеем:

$$x_0 + \tilde{x}_{n+1} = f(x_0 + \tilde{x}_n) \approx f(x_0) + f'(x_0)\tilde{x}_n.$$

Откуда следует, что

$$\tilde{x}_{n+1} = f'(x_0)\tilde{x}_n = \mu\tilde{x}_n.$$

Соответственно, что на каждом шаге дискретного времени (итерации) возмущение умножается на величину, равную мультипликатору. Следовательно, возмущение будет убывать и точка будет *устойчивой*, если $|\mu| < 1$. Иначе, при $|\mu| > 1$ возмущение будет нарастать, и точка окажется *неустойчивой*.

Таким образом, при $|\mu| \neq 1$ и небольшом изменении параметра отображения неподвижная точка сохранится и сохранит свой характер устойчивости - *случай общего положения*.

Существуют вырожденные случаи, при $|\mu| = 1$. Даже если изменить параметр на очень малую величину, то поведение системы изменится существенным образом. Такая ситуация называется бифуркацией.

Бифуркация

Бифуркация – эти качественная перестройка картины движения при малом изменении параметров отображения.

Значения управляющего параметра, при которых происходят бифуркации, называются **критическими** или **бифуркационными значениями**.

Качественная перестройка установившегося режима в одномерных отображениях управляется мультипликатором. Это утверждение с очевидностью носит общий характер. Действительно, пусть имеется некоторое одномерное отображение, зависящее от единственного параметра и имеющее неподвижную точку. Тогда при плавном изменении этого параметра будет изменяться взаимное расположение функции, характеризующей отображение, и биссектрисы на итерационной диаграмме. Следовательно, будет меняться мультипликатор неподвижной точки. При выполнении условия $|\mu| = 1$ точка будет терять устойчивость, а значит, будет иметь место некоторая бифуркация. Для одномерных отображений мультипликатор всегда действительная величина, поэтому бифуркации возможны в двух случаях: $\mu = 1$ и $\mu = -1$.

Бифуркация периодических циклов

Замечательное свойство отображений состоит в том, что бифуркационный анализ периодических циклов может быть построен совершенно аналогично анализу бифуркаций неподвижных точек. Обоснуем это утверждение. Пусть отображение $x_{n+1} = f(x_n)$ имеет цикл периода N. Это означает, что через N итераций последовательность x_n повторяется, то есть $x_{n+N} = x_n$.

Поскольку, $x_{n+1} = f(x_n)$, $x_{n+2} = f(f(x_n))$ и так далее, то условие реализации N-цикла можно переписать так:
$x_n = f^N(x_n)$,
где $f^N(x) = f(f \ldots (f(x_n)))$ - N-кратно проитерированная функция, задающая отображение. Следовательно, если отображение имеет N-цикл, то отображение $f^N(x)$ имеет неподвижную точку, и наоборот.Этот результат позволяет сразу ввести определение мультипликатора N-цикла:
$\mu = (f^N(x))'$.
В частном случае для 2-цикла имеем:
$\mu = f'(x_2)f'(x_1)$.
Если мультипликатор цикла обратится в ноль, то такой цикл будет сверхустойчивым.

Аттрактор

Аттрактор - подмножество A фазового пространства, обладающее следующими свойствами:

- $a \in A \Rightarrow \forall t > 0 : f(t, a) \in A$;
- определена окрестность A, именуемая "бассейном притяжения" (basin of attraction) и обозначаемая $B(A)$, для которой выполняется
  $\forall b \in B(A) : \forall \epsilon > 0 \; \exists T > 0 : f(t, b) \in U_\epsilon(A) \; \forall t > T$;
- не существует непустого подмножества A, удовлетворяющего вышеописанным свойствам.

Аттрактор именуется **странным**, если он имеет фрактальную структуру.

Ситуация, в которой при изменении параметра(-ов) динамической системы происходит появление/исчезновение аттрактора, именуется кризисом (crisis). Согласно классификации Гребоги, Отта, Ромейраса и Йорка (1983), сужествует 3 типа кризисов:

- внешний кризис - аттрактор формируется/разрушается; происходит при изменении параметра(-ов) системы;
- внутренний кризис - аттрактор "увеличивается"/"уменьшается"; происходит, когда при изменении параметра аттрактор "сталкивается" с неподвижной точкой или циклом, появляющимся(-ейся) внутри бассейна притяжения исходного аттрактора;
- кризис слияния - два аттрактора сливаются в один / аттрактор распадается на два; происходит при изменении параметра(-ов) системы.

1. $x_{n+1} = 1 - \alpha x_n^2$

Неподвижные точки

В нашем случае, $f(x) = 1 - \alpha x^2$.

Таким образом, неподвижные точки этого отображения можно найти, следующим образом:

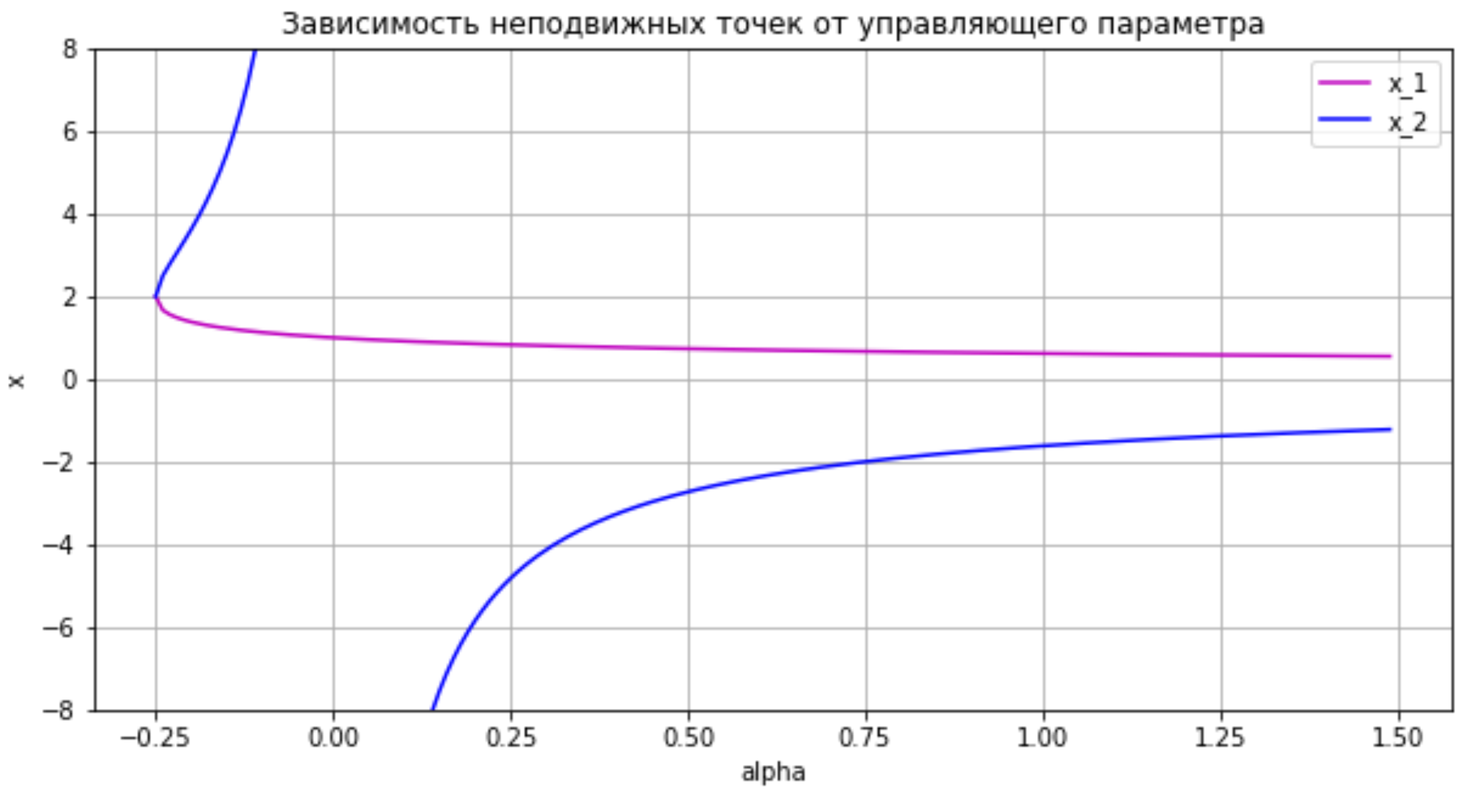
$$x = 1 - \alpha x^2$$
$$\alpha x^2 + x - 1 = 0$$
$$x_{1,2} = \frac{-1 \pm \sqrt{1 + 4\alpha}}{2\alpha} \text{ при } \alpha \geq -\frac{1}{4} \text{ и } \alpha \neq 0$$

Построим график $x_1(\alpha) = \frac{-1+\sqrt{1+4\alpha}}{2\alpha}$, $x_2(\alpha) = \frac{-1-\sqrt{1+4\alpha}}{2\alpha}$.

In [0]:

```
def f(a):  # TODO lambda
  return (-1 + np.sqrt(1 + 4 * a))/(2 * a)
def g(a):  # TODO lambda
  return (-1 - np.sqrt(1 + 4 * a))/(2 * a)

ax = new_plot(title="Зависимость неподвижных точек от управляющего параметра")

draw_line(ax, f, -1, 1.5, style='m', label='x_1')
draw_line(ax, g, -1, 0, style='b', label='x_2')
draw_line(ax, g, 0, 1.5, style='b')

show_plot(-8, 8)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: Ru
ntimeWarning: invalid value encountered in sqrt

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: invalid value encountered in sqrt
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Точки бифуркации

Вернемся к нашему отображению.

$f(x) = 1 - \alpha x^2,$

$f'(x) = -2\alpha x.$

Подставим неподвижную точку x

$\mu = f'(x) = -2\alpha \frac{-1 \pm \sqrt{1+4\alpha}}{2\alpha} = 1 \mp \sqrt{1+4\alpha}.$

Найдем критические значения параметра α, то есть те значения, при которых $\mu = 1,\ \mu = -1$:

- $\mu = 1$

  $1 \mp \sqrt{1+4\alpha} = 1$

  $\sqrt{1+4\alpha} = 0$

  $\alpha = \frac{-1}{4}$

  Найдем x:

  $x_{1,2} = 4\frac{-1 \pm \sqrt{1-1}}{-2} = 2$

Соответственно, $x = 2$ - неподвижная точка.

- $\mu = -1$

  $1 \mp \sqrt{1+4\alpha} = -1$

  $\mp\sqrt{1+4\alpha} = -2$

  $1 + 4\alpha = 4$

  $\alpha = \frac{3}{4}$

  Найдем x:

  $x_{1,2} = 4\frac{-1 \pm \sqrt{1+3}}{6} = \frac{-2 \pm 4}{3}$

  Таким образом, точки $x_1 = -2$ и $x_2 = 2/3$ - две неподвижные точки.

Рассмотрим устойчивость/неустойчивость

Точка будет устойчивой, если $|\mu| < 1$ и неустойчивой, если $|\mu| > 1$.

- $\mu_{x_1} = 1 - \sqrt{1+4\alpha}$

  $-1 < \mu_{x_1} < 1$

  $-1 < 1 - \sqrt{1+4\alpha} < 1$

  $-2 < -\sqrt{1+4\alpha} < 0$

  $0 < 1 + 4\alpha < 4$

  $-1 < 4\alpha < 3$

  Точка x_1 - *неустойчива* при $\alpha > \frac{3}{4}$ и *устойчива* на промежутке $-\frac{1}{4} < \alpha < \frac{3}{4}$.

- $\mu_{x_2} = 1 + \sqrt{1+4\alpha}$

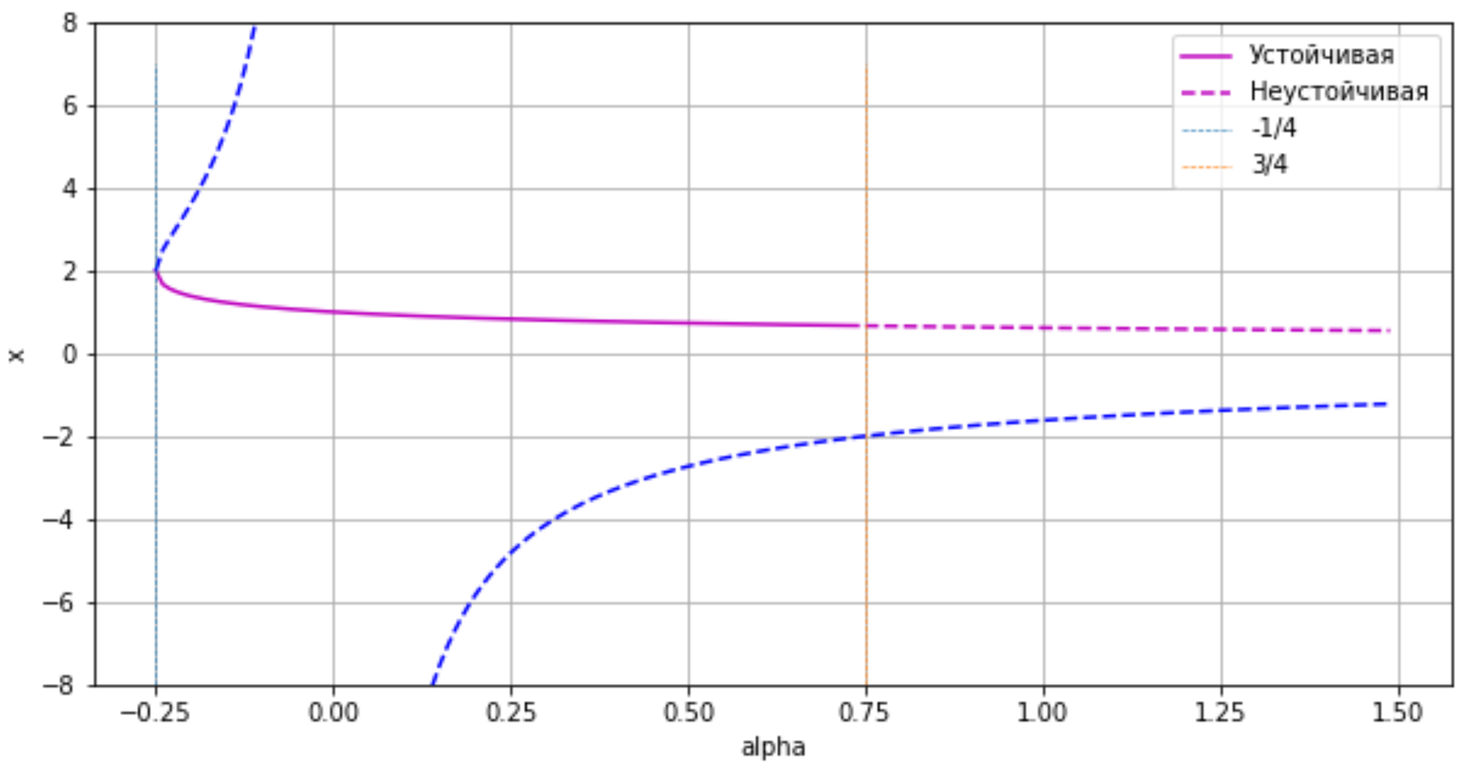
  Заметим, что $\mu_{x_2} = 1 + \sqrt{1+4\alpha} > 1$ при любом значении параметра $\alpha > -\frac{1}{4}$.

  Следовательно, точка x_2 - *неустойчивая* при любом $\alpha > -\frac{1}{4}$.

In [0]:

```
def f(a):  # TODO lambda
  return (-1 + np.sqrt(1 + 4 * a))/(2 * a)
def g(a):  # TODO lambda
  return (-1 - np.sqrt(1 + 4 * a))/(2 * a)

ax = new_plot()

draw_line(ax, f, -1, 3./4, style='m', label='Устойчивая')
draw_line(ax, f, 3./4, 1.5, style='m--', label='Неустойчивая')
draw_line(ax, g, -1, 0, style='b--')
draw_line(ax, g, 0, 1.5, style='b--')

draw_asym(-1/4, label='-1/4')
draw_asym(3./4, label='3/4')

show_plot(-8, 8)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: Ru
ntimeWarning: invalid value encountered in sqrt

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: invalid value encountered in sqrt
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Два-цикл

Условие существования два цикла можно записать в виде:
$x_1 = f(x_2), x_2 = f(x_1).$
В нашем случае, для $f(x_{n+1}) = 1 - \alpha x_n^2$ имеем:
$x_1 = 1 - \alpha x_2^2$, $x_2 = 1 - \alpha x_1^2$
Подставим одно в другое и найдем $x_1,\ x_2$
$x_2 = 1 - \alpha(1 - \alpha x_2^2)^2$
$x_2 = 1 - \alpha(1 - 2\alpha x_2^2 + \alpha^2 x_2^4)$
$x_2 = 1 - \alpha + 2\alpha^2 x_2^2 - \alpha^3 x_2^4$
$\alpha^3 x_2^4 - 2\alpha^2 x_2^2 + x_2 + (\alpha - 1) = 0$

$(\alpha x^2 + x - 1) \cdot (\alpha^2 x^2 - \alpha x + (1 - \alpha)) = 0$

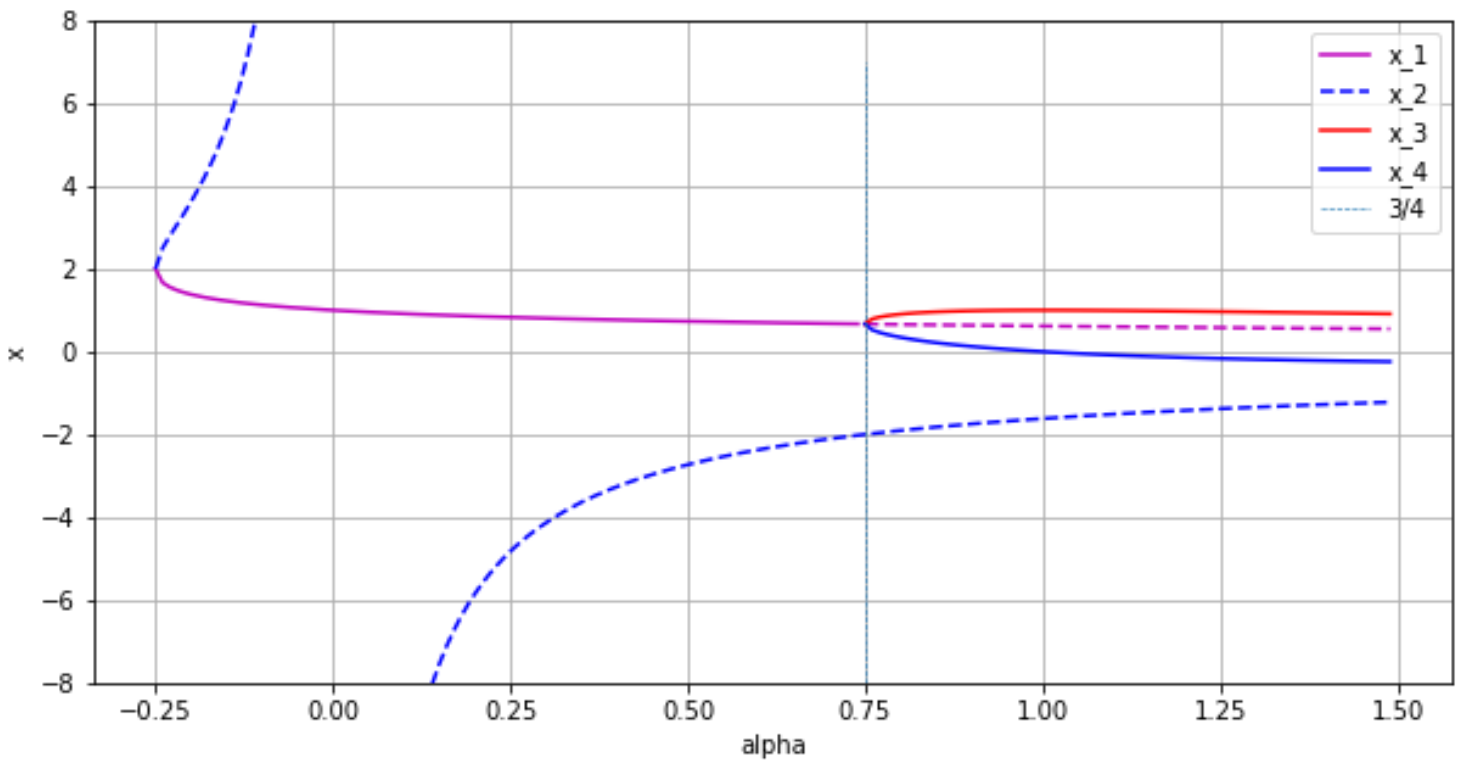
- $\alpha x^2 + x - 1 = 0$
  $x_{3,4} = \frac{-1 \pm \sqrt{1+4\alpha}}{2\alpha}$ при $\alpha \geq \frac{-1}{4}$ Замерит что это в точности неподвижные точки (1-цикл), полученные в первом пункте. Потому что в условие, которые мы задали изначало он входит при $x_1 = x_2$
- $\alpha^2 x^2 - \alpha x + (1 - \alpha) = 0$ $x_{1,2} = \frac{\alpha \pm \sqrt{\alpha^2 - 4\alpha^2(1-\alpha)}}{2\alpha^2}$
  $x_{1,2} = \frac{1 \pm \sqrt{-3+4\alpha}}{2\alpha}$ при $\alpha \geq \frac{3}{4}$

Построим графики $x_3 = \frac{1+\sqrt{-3+4\alpha}}{2\alpha},\ x_4 = \frac{1-\sqrt{-3+4\alpha}}{2\alpha}$

In [0]:

```
def f(a):   # TODO lambda
  return (-1 + np.sqrt(1 + 4 * a))/(2 * a)
def g(a):   # TODO lambda
  return (-1 - np.sqrt(1 + 4 * a))/(2 * a)

def h(a):
  return (1 + np.sqrt(-3 + 4 * a)) / (2 * a)
def t(a):
  return (1 - np.sqrt(-3 + 4 * a)) / (2 * a)


ax = new_plot()

draw_line(ax, f, -1, 3./4, style='m', label='x_1')
draw_line(ax, f, 3./4, 1.5, style='m--')
draw_line(ax, g, -1, 0, style='b--', label='x_2')
draw_line(ax, g, 0, 1.5, style='b--')

draw_line(ax, h, 3./4, 1.5, style='r', label="x_3")
draw_line(ax, t, 3./4, 1.5, style='b', label="x_4" )

draw_asym(3./4, label='3/4')

show_plot(-8, 8)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: Ru
ntimeWarning: invalid value encountered in sqrt

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: invalid value encountered in sqrt
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Точки бифункации

$f(x) = 1 - \alpha x^2$
$f'(x) = -2\alpha x$

Подставим $x_3,\ x_4$
$\mu = f'(x_3) \cdot f'(x_4) = 4\alpha^2 x_3 x_4 = 4\alpha \frac{1+\sqrt{-3+4\alpha}}{2\alpha} \frac{1-\sqrt{-3+4\alpha}}{2\alpha} = \frac{1-(-3+4\alpha)}{\alpha} = \frac{4-4\alpha}{\alpha} = 4(1-\alpha)$

Найдем α, где $\mu = 1,\ \mu = -1$:

- $\mu = 1$
  $4(1-\alpha) = 1\ 1 - \alpha = \frac{1}{4}$
  $\alpha = \frac{3}{4}$
  Наедм x:
  $x_{3,4} = \frac{1 \pm \sqrt{-3+4\alpha}}{2\alpha} = \frac{2}{3}$
  $x_3 = x_4 = 2/3$ - одна точка. Бифуркация $\mu = 1$
- $\mu = -1$
  $4(1-\alpha) = -1\ 1 - \alpha = \frac{-1}{4}$
  $\alpha = \frac{5}{4}$
  Наедм x:
  $x_{3,4} = \frac{1 \pm \sqrt{-3+4\alpha}}{2\alpha} = 2\frac{1 \pm \sqrt{2}}{5}$
  $x_3 = 2\frac{1+\sqrt{2}}{5},\ x_4 = 2\frac{1-\sqrt{2}}{5}$ - две точки. Бифуркация $\mu = -1$

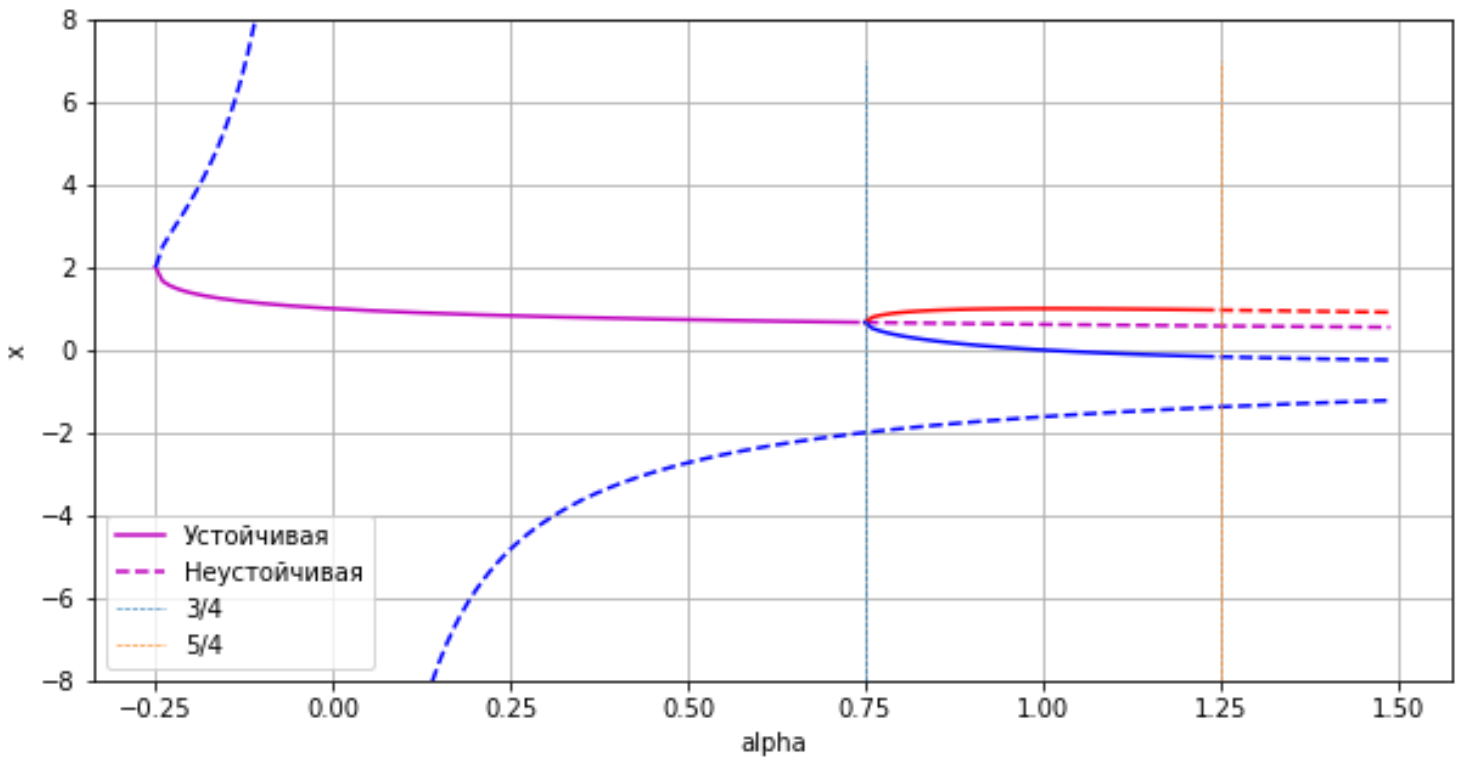
Рассмотрим устойчивость/неустойчивость

- $-1 < \mu_{x_{3,4}} < 1$
  $-1 < 4(1-\alpha) < 1$
  $\frac{-1}{4} < 1 - \alpha < \frac{1}{4}$
  $\frac{-5}{4} < -\alpha < \frac{-3}{4}$
  $\frac{3}{4} < \alpha < \frac{5}{4}$
  $\frac{3}{4} < \alpha < \frac{5}{4}$ - на промежутке - $x_3,\ x_4$ - устойчивые
  для $\alpha > \frac{4}{4}$ - неустойчивые

In [0]:

```
def f(a):  # TODO lambda
  return (-1 + np.sqrt(1 + 4 * a))/(2 * a)
def g(a):  # TODO lambda
  return (-1 - np.sqrt(1 + 4 * a))/(2 * a)

def h(a):
  return (1 + np.sqrt(-3 + 4 * a)) / (2 * a)
def t(a):
  return (1 - np.sqrt(-3 + 4 * a)) / (2 * a)



ax = new_plot()

draw_line(ax, f, -1, 3./4, style='m', label='Устойчивая')
draw_line(ax, f, 3./4, 1.5, style='m--', label='Неустойчивая')
draw_line(ax, g, -1, 0, style='b--')
draw_line(ax, g, 0, 1.5, style='b--')



draw_line(ax, h, 3./4, 5./4, style='r')
draw_line(ax, h, 5./4, 1.5, style='r--')
draw_line(ax, t, 3./4, 5./4, style='b')
draw_line(ax, t, 5./4, 1.5, style='b--')

draw_asym(3./4, label='3/4')
draw_asym(5./4, label='5/4')

show_plot(-8, 8)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: Ru
ntimeWarning: invalid value encountered in sqrt

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: invalid value encountered in sqrt
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



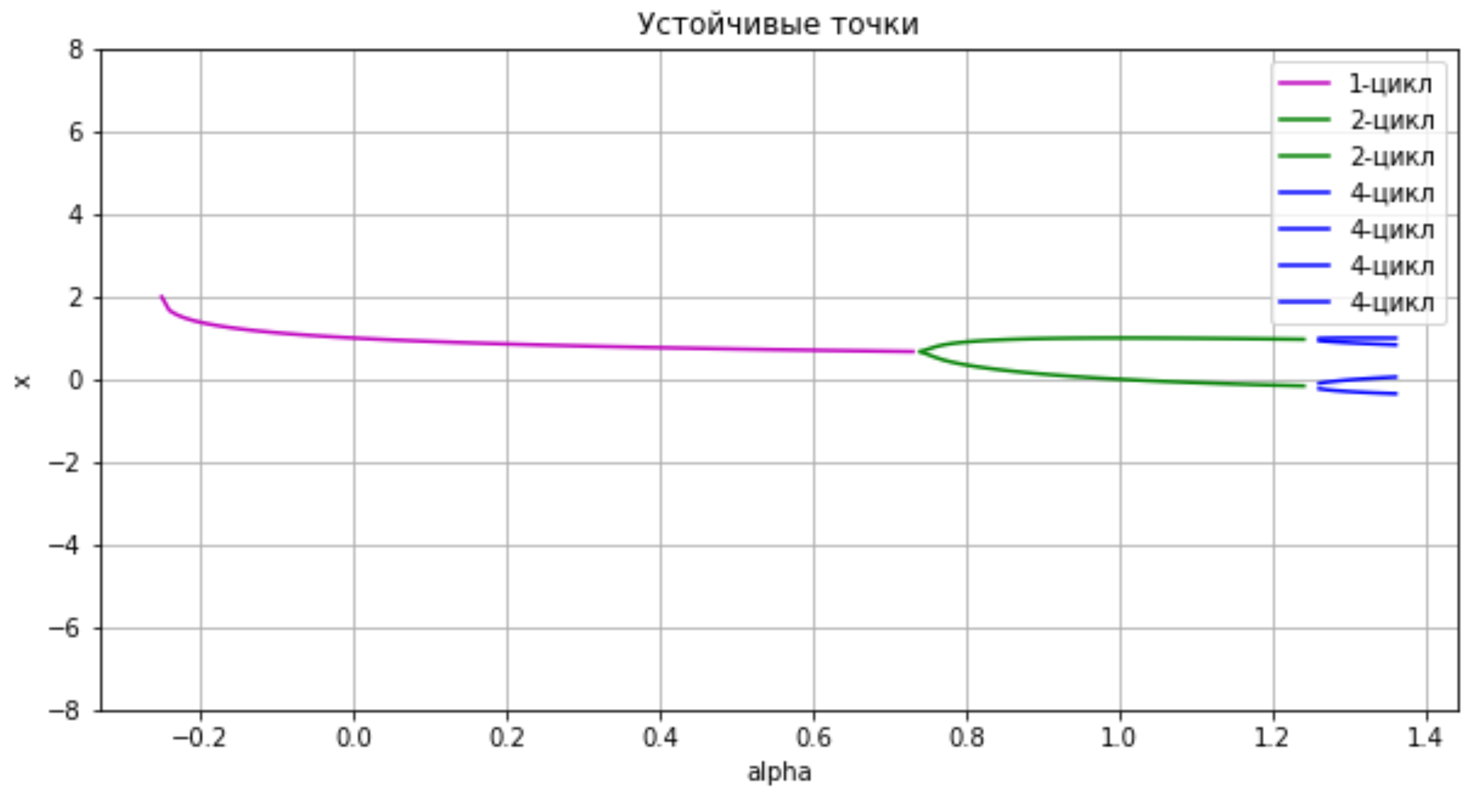
Дерево бифуркаций

In [0]:

```python
def f(x, a, t=1):
    if t > 1:
        x = f(x, a, t-1)

    return 1 - a * (np.square(x))


inf = 100
eps = 10e-6
a = np.arange(-1, 4, 0.01)

a_1 = []
x_1 = []

a_2 = []
x_2 = []

a_4 = []
x_4 = []

x_tmp = 0
for i in a:
    x_tmp = 0
```

```python
    for j in range(1000):
        x_tmp = f(x_tmp, i)

#     print(i, x_tmp)

    if (-inf < x_tmp < inf and -eps < x_tmp - f(x_tmp, i) < eps):
        x_1.append(x_tmp)
        a_1.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 2) < eps:
        x_2.append([x_tmp, f(x_tmp, i)])
        a_2.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 4) < eps:
        x_4.append([x_tmp, f(x_tmp, i), f(x_tmp, i, 2), f(x_tmp, i, 3)])
        a_4.append(i)




fig, ax = plt.subplots(figsize=(10,5))
ax.plot(a_1, x_1, 'm', label='1-цикл')
ax.plot(a_2, x_2, 'g', label='2-цикл')
ax.plot(a_4, x_4, 'b', label='4-цикл')


ax.set(xlabel='alpha', ylabel='x',
       title='Устойчивые точки')
ax.grid()


show_plot(-8, 8)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:5: Ru
ntimeWarning: overflow encountered in square
  """
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:35: R
untimeWarning: invalid value encountered in double_scalars
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:40: R
untimeWarning: invalid value encountered in double_scalars
```

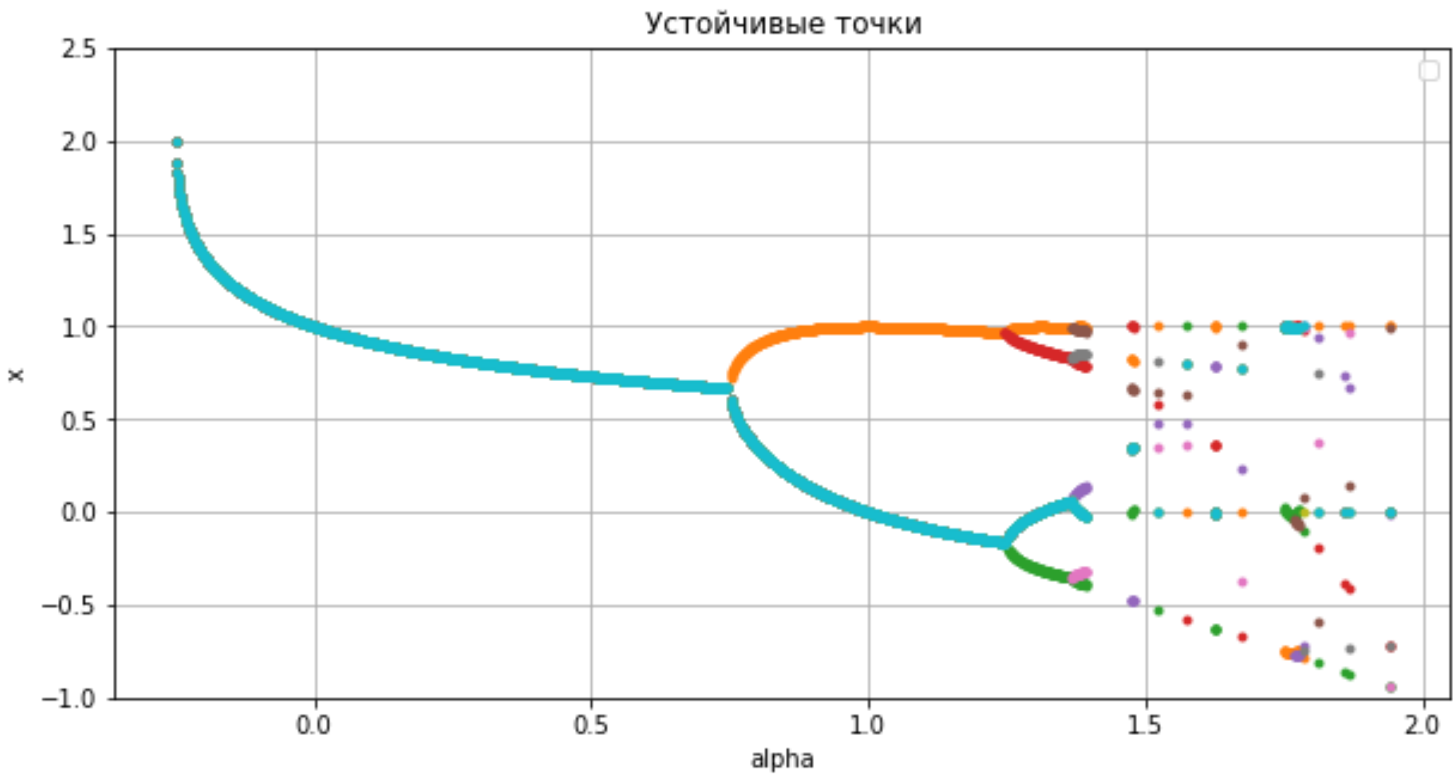


In [0]:

```python
def f(x, a, t=1):
    if t == 0:
        return x

    if t > 1:
        x = f(x, a, t-1)

    return 1 - a * (np.square(x))


inf = 100
inf_n = 10
eps = 10e-6
a = np.arange(-1, 4, 0.001)

a_ = []
x_ = []

x_tmp = 0
for i in a:

    if x_tmp < -eps or x_tmp > eps:
        x_tmp = 0

    for j in range(1000):
        x_tmp = f(x_tmp, i)

#     print(i, x_tmp)
```

```
    if not (-inf < x_tmp < inf):
        continue

    for n in range(1, inf_n):
        if -eps < x_tmp - f(x_tmp, i, n) < eps:

            tmp = []
            for j in range(n):
                tmp.append(f(x_tmp, i, j))

#             if n > 0:
#                 print(i, n)

#             print(tmp, n)
            for j in range(inf_n - n):
                tmp.append(tmp[0])

            x_.append(tmp)
            a_.append(i)

            break


# print(len(a_))
# print(len(x_))

fig, ax = plt.subplots(figsize=(10,5))
ax.plot(a_, x_, '.')


ax.set(xlabel='alpha', ylabel='x',
       title='Устойчивые точки')
ax.grid()


plt.ylim(-1., 2.5)     # set the ylim to bottom, top
plt.legend()
plt.show()
```

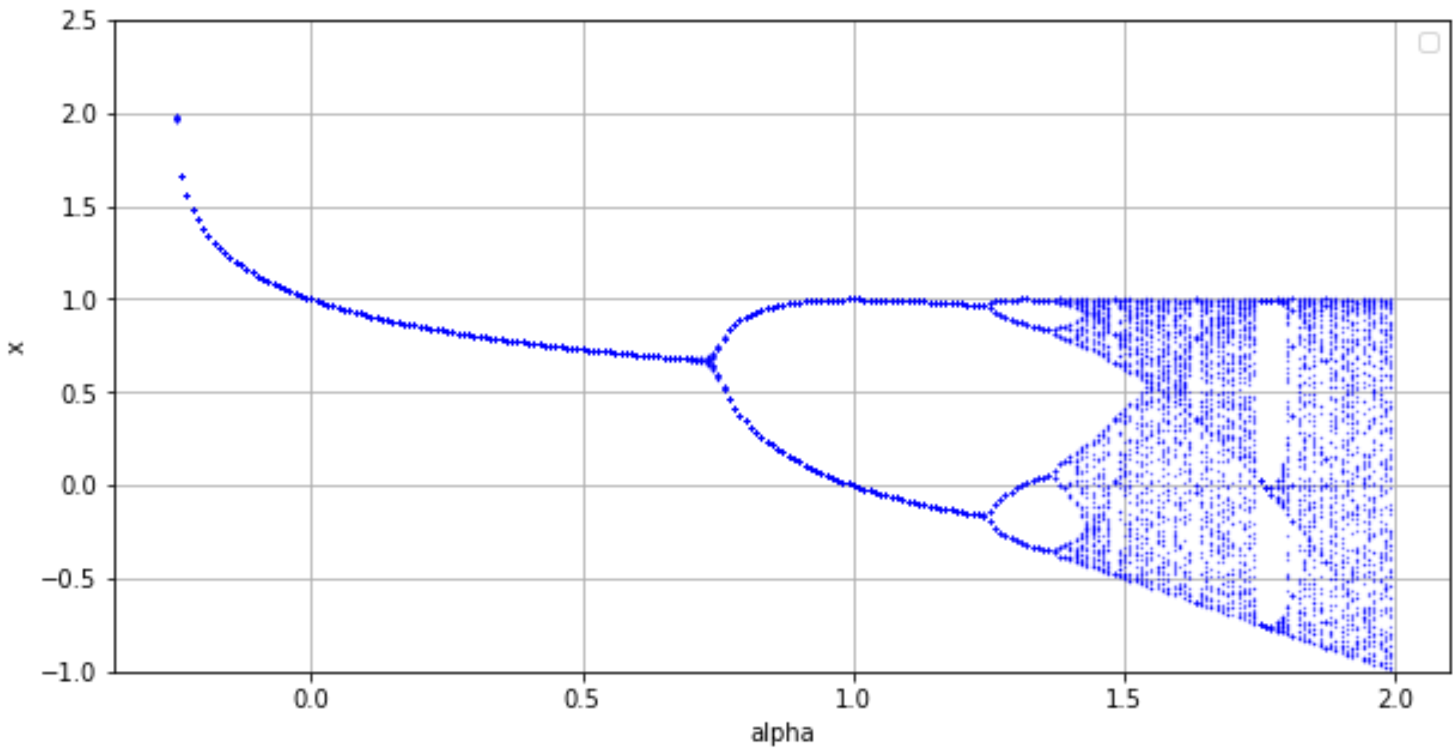
```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: overflow encountered in square

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: overflow encountered in double_scalars

No handles with labels found to put in legend.
```



In [0]:

```
def f(x, a, t=1):
    if t == 0:
        return x

    if t > 1:
        x = f(x, a, t-1)

    return 1 - a * (np.square(x))






ax = new_plot()

draw_tree(ax, f, -1, 4, style='.b')

show_plot(-1, 2.5)
```

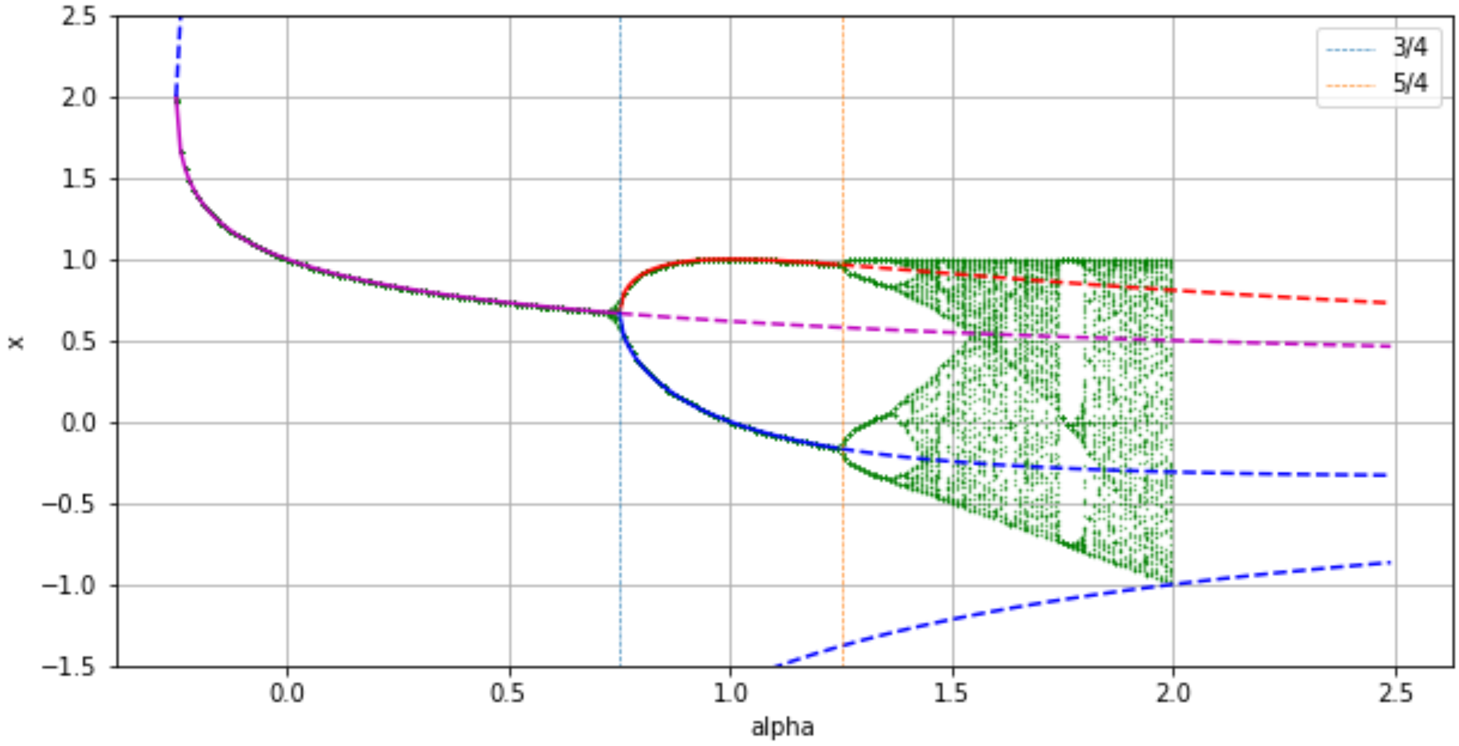
```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: overflow encountered in square

No handles with labels found to put in legend.
```



In [0]:

```
def f(a):  # TODO lambda
  return (-1 + np.sqrt(1 + 4 * a))/(2 * a)
def g(a):  # TODO lambda
  return (-1 - np.sqrt(1 + 4 * a))/(2 * a)

def h(a):
  return (1 + np.sqrt(-3 + 4 * a)) / (2 * a)
def t(a):
  return (1 - np.sqrt(-3 + 4 * a)) / (2 * a)


def u(x, a, t=1):
    if t == 0:
        return x

    if t > 1:
        x = f(x, a, t-1)

    return 1 - a * (np.square(x))



ax = new_plot()



draw_tree(ax, u, -1, 2.5, style='.g')


draw_line(ax, f, -1, 3./4, style='m')
draw_line(ax, f, 3./4, 2.5, style='m--')
draw_line(ax, g, -1, 0, style='b--')
draw_line(ax, g, 0, 2.5, style='b--')


draw_line(ax, h, 3./4, 5./4, style='r')
draw_line(ax, h, 5./4, 2.5, style='r--')
draw_line(ax, t, 3./4, 5./4, style='b')
draw_line(ax, t, 5./4, 2.5, style='b--')

draw_asym(3./4, label='3/4')
draw_asym(5./4, label='5/4')


show_plot(-1.5, 2.5)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:19: R
untimeWarning: overflow encountered in square
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: Ru
ntimeWarning: invalid value encountered in sqrt

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: invalid value encountered in sqrt
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Сценарий Фейгенбаума

Как показано на бифуркационной диаграмме, изменение управляющего параметра влечет каскад бифуркаций удвоения периода. Это означает, что при увеличении управляющего параметра и достаточно большом числе итераций отображение переходит сначала в устойчивую точку, затем в цикл с периодом два, потом в цикл с периодом 4 и так далее. Такой процесс называют переходом к хаосу через бесконечную последовательность бифуркаций удвоения периода. Этот сценарий перехода к хаосу называют сценарием Фейгенбаума

2. $x_{n+1} = \alpha x_n(1 - x_n)$

Неподвижные точки

$x = \alpha x(1 - x)$

$x = \alpha x - \alpha x^2$

$\alpha x^2 + (1 - \alpha)x = 0$

$x(\alpha x + 1 - \alpha) = 0$

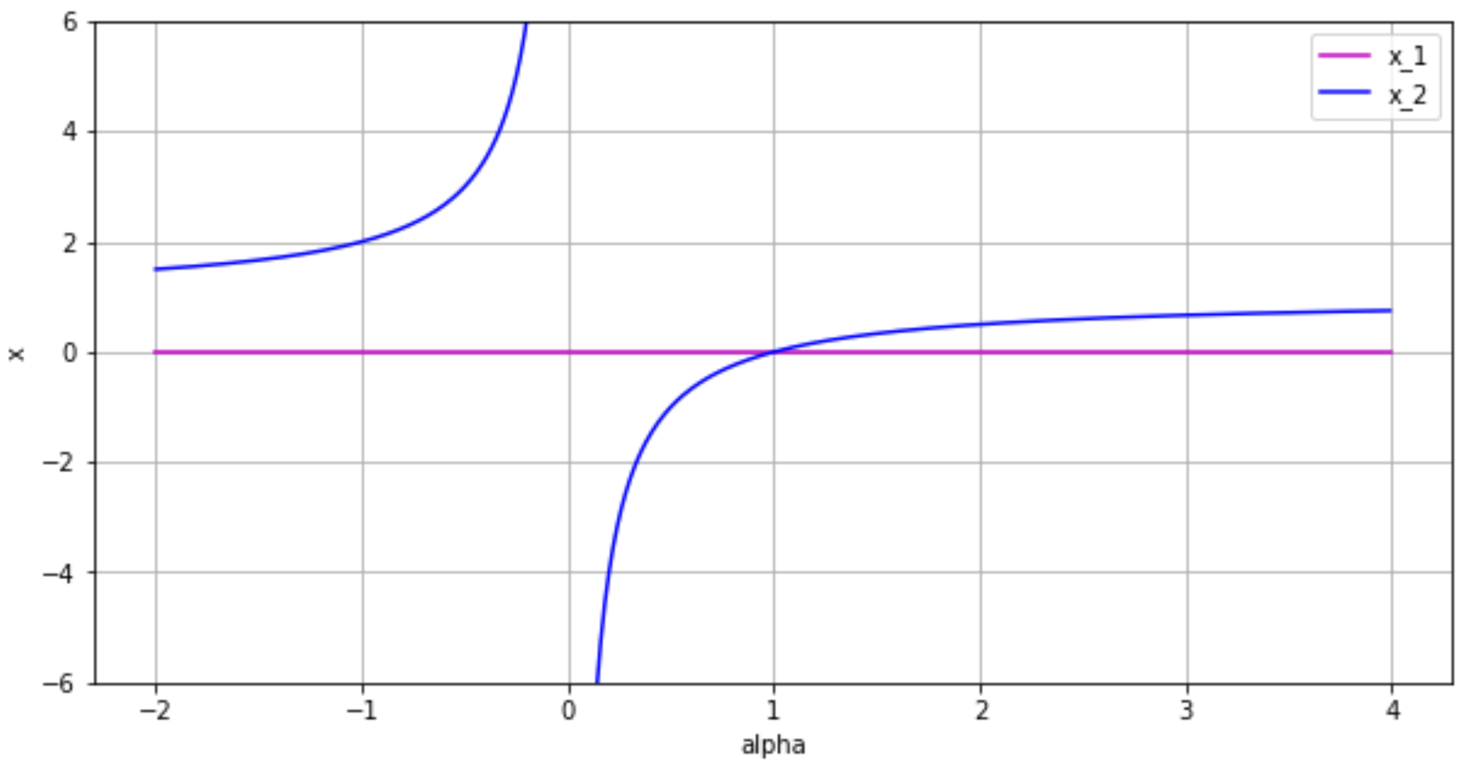
$x_1 = 0 \qquad x_2 = \frac{\alpha-1}{\alpha} \ \alpha \neq 0$

Построим график $x_1(\alpha) = 0$ и $x_2(\alpha) = \frac{\alpha-1}{\alpha}$

In [0]:

```
def f(a):  # TODO lambda
  return (0*a)
def g(a):  # TODO lambda
  return (a-1)/a

ax = new_plot()
draw_line(ax, f, -2, 4, style='m', label='x_1')
draw_line(ax, g, -2, 0, style='b', label='x_2')
draw_line(ax, g, 0, 4, style='b')

show_plot(-6, 6)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Точки бифуркации

$f(x) = \alpha x - \alpha x^2$
$f'(x) = \alpha - 2\alpha x$

Подставим x_1 и x_2

$1. \mu_{x_1} = f'(x_1) = \alpha$

Найдем α, при которых $\mu = 1,\ \mu = -1$:

- $\mu = 1$
  $\alpha = 1$

  Найдем x:
  $x = 0$ - неподвижная точка. Бифуркация $\mu = 1$
- $\mu = -1$
  $\alpha = -1$

  Найдем x:
  $x = 0$ - неподвижная точка. Бифуркация $\mu = -1$

$2. \mu_{x_2} = f'(x_2) = -\alpha + 2$

Найдем α, при которых $\mu = 1,\ \mu = -1$:

- $\mu = 1$
  $-\alpha + 2 = 1$

  $\alpha = 1$

  Найдем x:
  $x = 0$ - неподвижная точка. Бифуркация $\mu = 1$
- $\mu = -1$
  $-\alpha + 2 = -1$

  $\alpha = 3$

  Найдем x:
  $x_2 = 2/3$ - неподвижная точка. Бифуркация $\mu = -1$

Рассмотрим устойчивость/неустойчивость

- $\mu_{x_1} = \alpha$
  $-1 < \mu_{x_1} < 1$
  $-1 < \alpha < 1$ - на промежутке x_1 устойчивая, для $\alpha > 1$ и $\alpha < -1$ неустойчивая

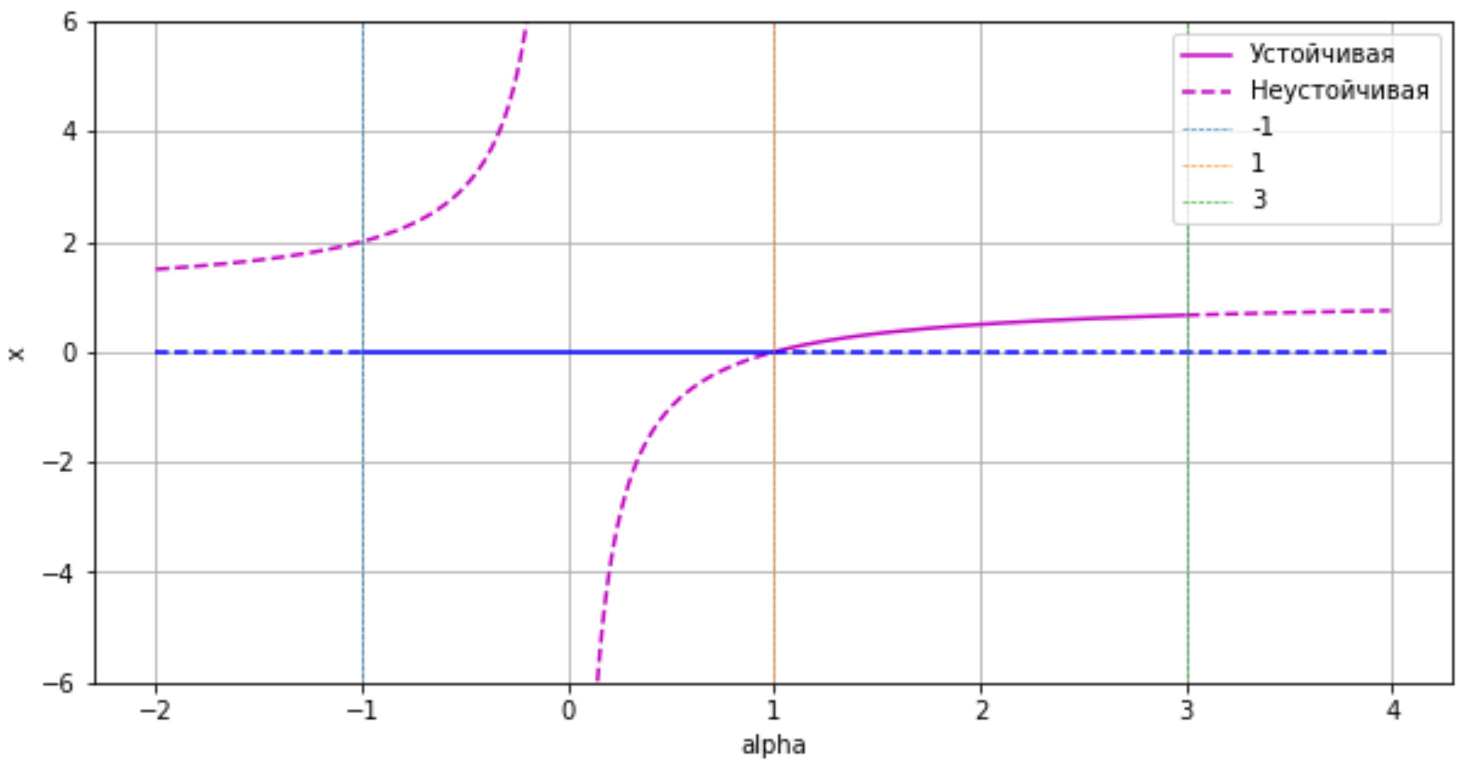
- $\mu_{x_2} = -\alpha + 2$
  $-1 < 2 - \alpha < -1$

  $-1 < \alpha < 3$ на промежутке - x_2 - устойчивая
  для $\alpha > 3$ и $\alpha < 1$ - неустойчивая

In [0]:

```python
def f(a):  # TODO lambda
  return (0*a)
def g(a):  # TODO lambda
  return (a-1)/a

ax = new_plot()

draw_line(ax, g, 1, 3, style='m', label='Устойчивая')
draw_line(ax, g, -2, 0, style='m--')
draw_line(ax, g, 0, 1, style='m--')
draw_line(ax, g, 3, 4, style='m--', label='Неустойчивая')
draw_line(ax, f, -1, 1, style='b')
draw_line(ax, f, -2, -1, style='b--')
draw_line(ax, f, 1, 4, style='b--')

draw_asym(-1, label='-1')
draw_asym(1, label='1')
draw_asym(3, label='3')

show_plot(-6, 6)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: Ru
ntimeWarning: divide by zero encountered in true_divide
  after removing the cwd from sys.path.
```



Два цикл

$x_1 = f(x_2)$
$x_2 = f(x_1) \Rightarrow$

$x_1 = \alpha x_2(1 - x_2)$
$x_2 = \alpha x_1(1 - x_1) \Rightarrow$

Подставим одно в другое и найдем $x_1,\ x_2$
$x_2 = \alpha(\alpha x_2(1 - x_2))(1 - \alpha x_2(1 - x_2)) \quad x_2 \neq 0$

$1 = (\alpha^2 - \alpha^2 x_2)(1 - \alpha x_2 + \alpha x_2^2)$

$(1 - \alpha + \alpha x)(1 + \alpha - \alpha x - \alpha^2 x + \alpha^2 x^2) = 0$

- $(1 - \alpha + \alpha x) = 0 \quad x_1 = 0 \quad x_2 = \frac{\alpha - 1}{\alpha}$ при $\alpha \neq 0$ Заметим, что это в точности неподвижные точки (1-цикл), полученные в первом пункте. Потому что в условие, которые мы задали изначало он входит при $x_1 = x_2$
- $1 + \alpha - \alpha x - \alpha^2 x + \alpha^2 x^2 = 0$

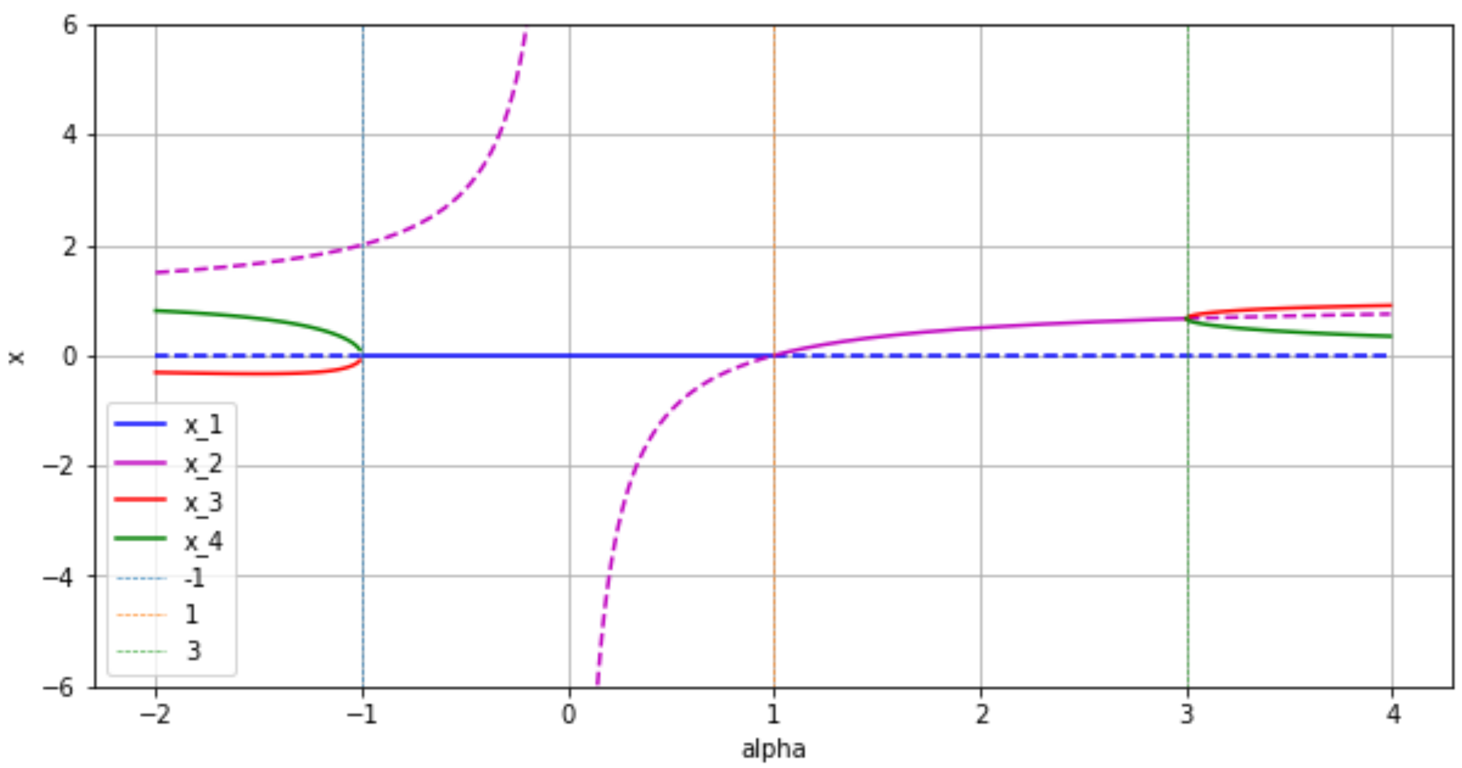
  $x_{3,4} = \frac{1}{2} + \frac{1 \pm \sqrt{\alpha^2 - 2\alpha - 3}}{2\alpha}$

Построим графики $x_3 = \frac{1}{2} + \frac{1 + \sqrt{\alpha^2 - 2\alpha - 3}}{2\alpha},\ x_4 = \frac{1}{2} + \frac{1 - \sqrt{\alpha^2 - 2\alpha - 3}}{2\alpha}$

In [0]:

```
def h(a):  # TODO lambda
  return (1/2+(1+np.sqrt(a*a-2*a-3))/(2*a))
def t(a):  # TODO lambda
  return (1/2+(1-np.sqrt(a*a-2*a-3))/(2*a))
def f(a):  # TODO lambda
  return (0*a)
def g(a):  # TODO lambda
  return (a-1)/a



ax = new_plot()
draw_line(ax, f, -1, 1, style='b',label='x_1')
draw_line(ax, f, -2, -1, style='b--')
draw_line(ax, f, 1, 4, style='b--')
draw_line(ax, g, 1, 3, style='m',label='x_2')
draw_line(ax, g, -2, 0, style='m--')
draw_line(ax, g, 0, 1, style='m--')
draw_line(ax, g, 3, 4, style='m--')
draw_line(ax,h,-2,-1,style='r',label='x_3')
draw_line(ax,h,3,4,style='r')
draw_line(ax,t,-2,-1,style='g',label='x_4')
draw_line(ax,t,3,4,style='g')
draw_asym(-1, label='-1')
draw_asym(1, label='1')
draw_asym(3, label='3')

show_plot(-6, 6)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: divide by zero encountered in true_divide
```



Точки бифуркации

$f(x) = \alpha x - \alpha x^2$
$f'(x) = \alpha - 2\alpha x$
Подставим $x_3,\ x_4$
$\mu = f'(x_3) \cdot f'(x_4) = (\alpha - 2\alpha x_3)(\alpha - 2\alpha x_4) = -\alpha^2 + 2\alpha + 4$

Найдем α, где $\mu = 1,\ \mu = -1$:

- $\mu = 1$
  $-\alpha^2 + 2\alpha + 4 = 1$

  $\alpha = -1 \quad \alpha = 3$
  Найдем x:

  $x_{3,4} = \frac{1}{2} + \frac{1 \pm \sqrt{\alpha^2 - 2\alpha - 3}}{2\alpha}$
  $x = \frac{2}{3} \quad x = 0$ Бифуркация $\mu = 1$
- $\mu = -1$
  $\alpha^2 - 2\alpha - 5 = 0$
  $\alpha = 1 \pm \sqrt{6}$
  Найдем x:
  $x_{3,4} = \frac{1}{2} + \frac{1 \pm \sqrt{\alpha^2 - 2\alpha - 3}}{2\alpha}$
  $x = \frac{3+\sqrt{6}}{2+2\sqrt{6}} \quad x = \frac{1+\sqrt{6}}{2+2\sqrt{6}} \quad x = \frac{3-\sqrt{6}}{2-2\sqrt{6}} \quad x = \frac{1-\sqrt{6}}{2-2\sqrt{6}}$ - четыре точки. Бифуркация $\mu = -1$

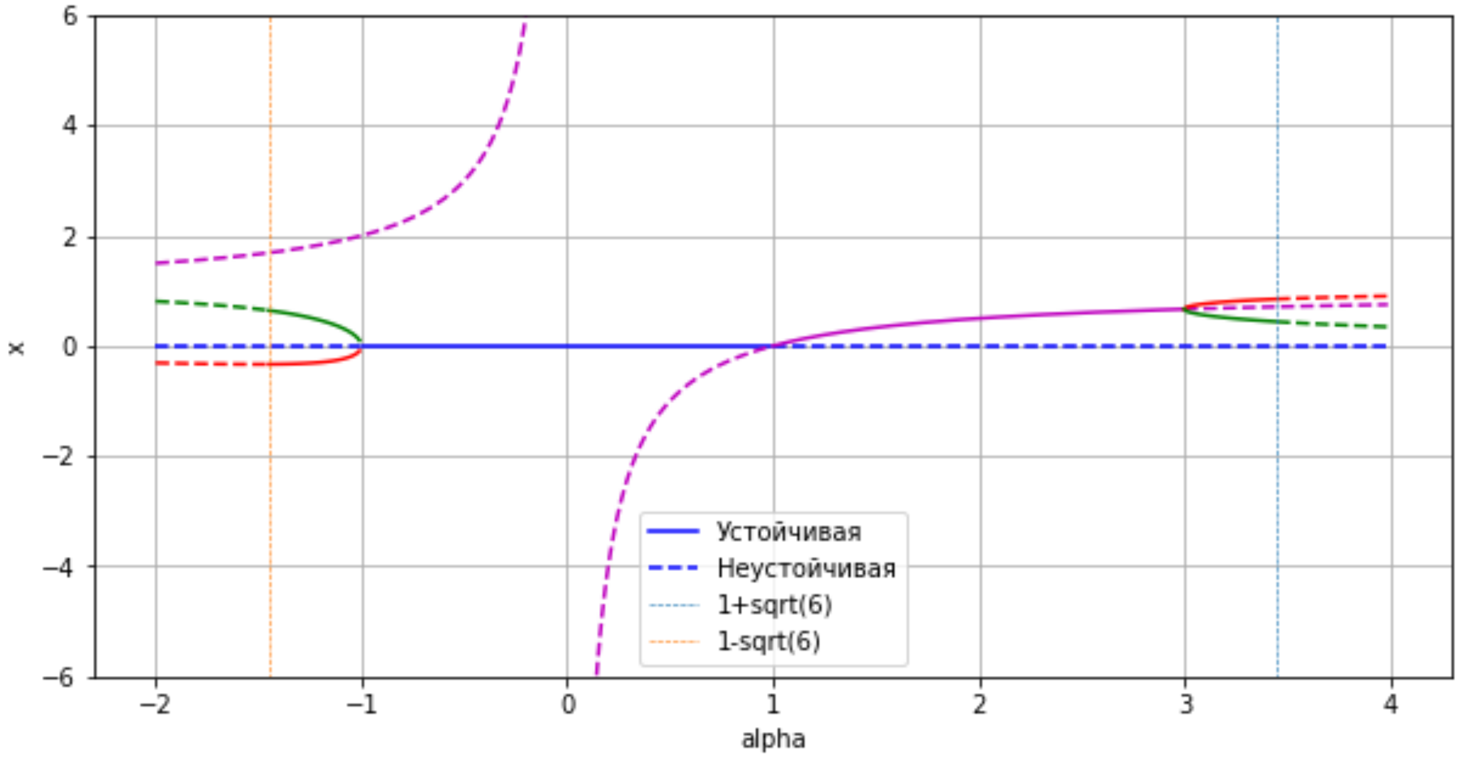
Рассмотрим устойчивость/неустойчивость

- $-1 < \mu_{x_{3,4}} < 1$
  $-1 < -\alpha^2 + 2\alpha + 4 < 1$
  $3 < x < 1 + \sqrt{6}; \quad 1 - \sqrt{6} < x < -1$ - на промежутке - $x_3,\ x_4$ - устойчивые

In [0]:

```
def h(a):  # TODO lambda
  return (1/2+(1+np.sqrt(a*a-2*a-3))/(2*a))
def t(a):  # TODO lambda
  return (1/2+(1-np.sqrt(a*a-2*a-3))/(2*a))
def f(a):  # TODO lambda
  return (0*a)
def g(a):  # TODO lambda
  return (a-1)/a




ax = new_plot()
draw_line(ax, f, -1, 1, style='b',label='Устойчивая')
draw_line(ax, f, -2, -1, style='b--',label='Неустойчивая')
draw_line(ax, f, 1, 4, style='b--')
draw_line(ax, g, 1, 3, style='m')
draw_line(ax, g, -2, 0, style='m--')
draw_line(ax, g, 0, 1, style='m--')
draw_line(ax, g, 3, 4, style='m--')

draw_line(ax,h,-2,1-np.sqrt(6),style='r--')
draw_line(ax,h,1-np.sqrt(6),-1,style='r')
draw_line(ax,h,3,1+np.sqrt(6),style='r')
draw_line(ax,h,1+np.sqrt(6),4,style='r--')
draw_line(ax,t,-2,1-np.sqrt(6),style='g--')
draw_line(ax,t,1-np.sqrt(6),-1,style='g')
draw_line(ax,t,3,1+np.sqrt(6),style='g')
draw_line(ax,t,1+np.sqrt(6),4,style='g--')

draw_asym(1+np.sqrt(6),label='1+sqrt(6)')
draw_asym(1-np.sqrt(6),label='1-sqrt(6)')



show_plot(-6, 6)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: divide by zero encountered in true_divide
```



Дерево бифуркаций

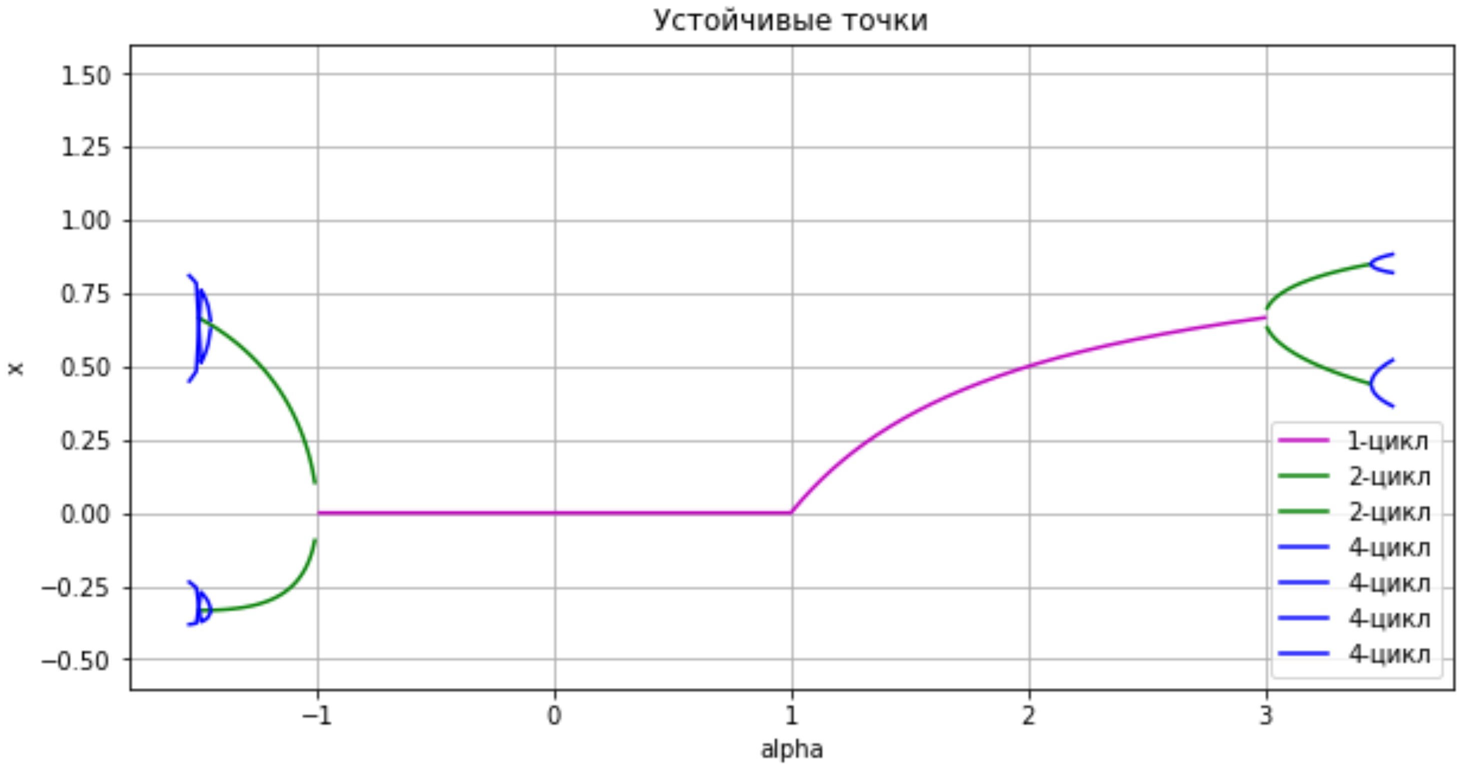
In [0]:

```python
def f(x, a, t=1):
    if t > 1:
        x = f(x, a, t-1)
    return (a*x-a*x*x)


inf = 100
eps = 10e-6
a = np.arange(-2,1, 0.01)

a_1 = []
x_1 = []

a_2 = []
x_2 = []

a_4 = []
x_4 = []

x_tmp =2
for i in a:
    x_tmp = 2/3
    for j in range(1000):
        x_tmp = f(x_tmp, i)

    #print(i, x_tmp)

    if (-inf < x_tmp < inf and -eps < x_tmp - f(x_tmp, i,1) < eps):
        x_1.append(x_tmp)
        a_1.append(i)
```

```
        a_1.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 2) < eps:
        x_2.append([x_tmp, f(x_tmp, i)])
        a_2.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 4) < eps:
        x_4.append([x_tmp, f(x_tmp, i), f(x_tmp, i, 2), f(x_tmp, i, 3)])
        a_4.append(i)


fig, ax = plt.subplots(figsize=(10,5))
ax.plot(a_1, x_1, 'm', label='1-цикл')
ax.plot(a_2, x_2, 'g', label='2-цикл')
ax.plot(a_4, x_4, 'b', label='4-цикл')

a = np.arange(1,4, 0.005)

a_1 = []
x_1 = []

a_2 = []
x_2 = []

a_4 = []
x_4 = []

x_tmp =2
for i in a:
    x_tmp = 2/3

    for j in range(1000):
        x_tmp = f(x_tmp, i)

    #print(i, x_tmp)

    if (-inf < x_tmp < inf and -eps < x_tmp - f(x_tmp, i,1) < eps):
        x_1.append(x_tmp)
        a_1.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 2) < eps:
        x_2.append([x_tmp, f(x_tmp, i)])
        a_2.append(i)
        continue

    if -eps < x_tmp - f(x_tmp, i, 4) < eps:
        x_4.append([x_tmp, f(x_tmp, i), f(x_tmp, i, 2), f(x_tmp, i, 3)])
        a_4.append(i)


ax.plot(a_1, x_1, 'm')
ax.plot(a_2, x_2, 'g')
ax.plot(a_4, x_4, 'b')
```

```
ax.set(xlabel='alpha', ylabel='x',
       title='Устойчивые точки')
ax.grid()
```

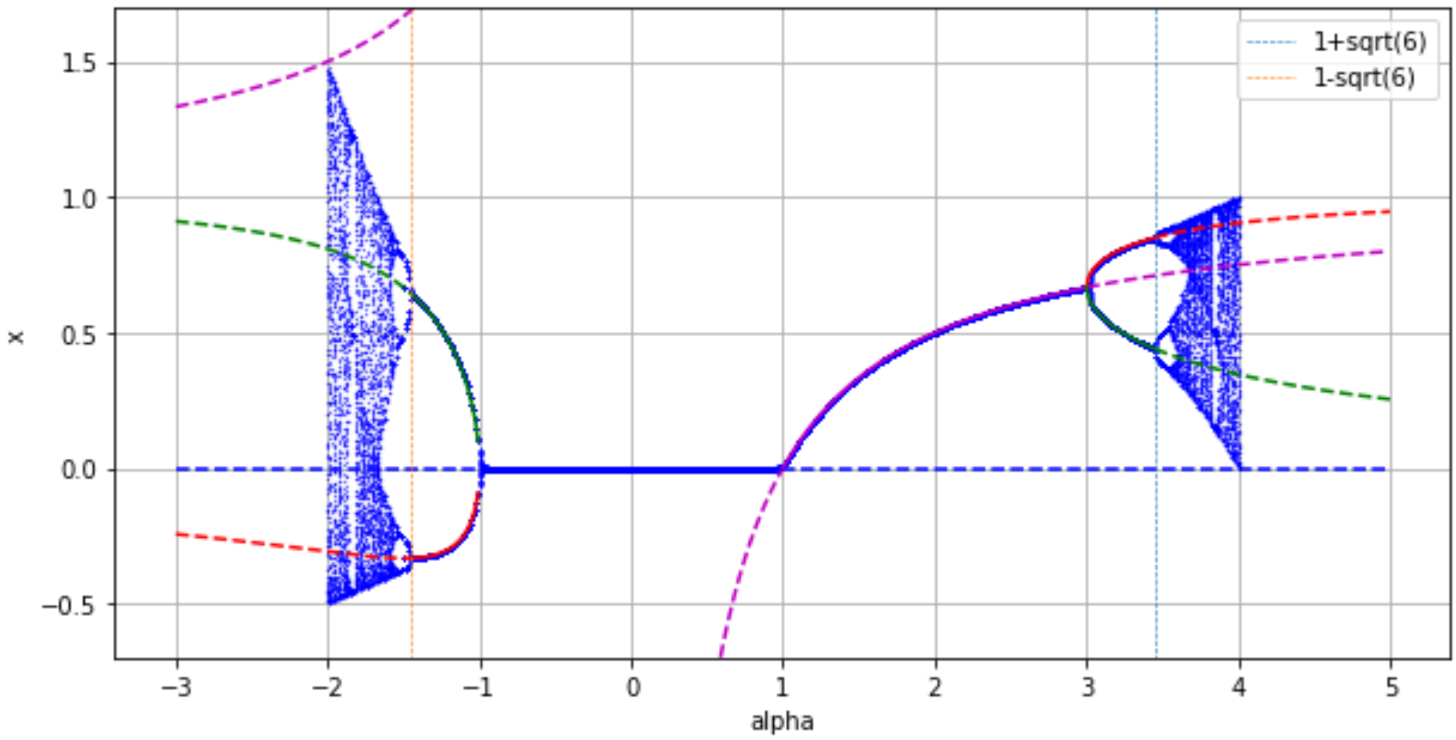
```
show_plot(-0.6, 1.6)
```



In [0]:

```
def h(a):  # TODO lambda
  return (1/2+(1+np.sqrt(a*a-2*a-3))/(2*a))
def t(a):  # TODO lambda
  return (1/2+(1-np.sqrt(a*a-2*a-3))/(2*a))
def f(a):  # TODO lambda
  return (0*a)
def g(a):  # TODO lambda
  return (a-1)/a




def u(x, a, t=1):
    if t > 1:
        x = f(x, a, t-1)
    return (a*x-a*x*x)





ax = new_plot()


draw_tree(ax, u, -3, 1, style='.b', start=2./3)
draw_tree(ax, u, 1, 5, style='.b', start=2./3)


draw_line(ax, f, -1, 1, style='b')
draw_line(ax, f, -3, -1, style='b--')
draw_line(ax, f, 1, 5, style='b--')
draw_line(ax, g, 1, 3, style='m')
draw_line(ax, g, -3, 0, style='m--')
draw_line(ax, g, 0, 1, style='m--')
draw_line(ax, g, 3, 5, style='m--')

draw_line(ax,h,-3,1-np.sqrt(6),style='r--')
draw_line(ax,h,1-np.sqrt(6),-1,style='r')
draw_line(ax,h,3,1+np.sqrt(6),style='r')
draw_line(ax,h,1+np.sqrt(6), 5,style='r--')
draw_line(ax,t,-3,1-np.sqrt(6),style='g--')
draw_line(ax,t,1-np.sqrt(6),-1,style='g')
draw_line(ax,t,3,1+np.sqrt(6),style='g')
draw_line(ax,t,1+np.sqrt(6), 5,style='g--')

draw_asym(1+np.sqrt(6),label='1+sqrt(6)')
draw_asym(1-np.sqrt(6),label='1-sqrt(6)')





show_plot(-0.7, 1.7)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:15: R
untimeWarning: overflow encountered in double_scalars
  from ipykernel import kernelapp as app
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:15: R
untimeWarning: invalid value encountered in double_scalars
  from ipykernel import kernelapp as app
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: divide by zero encountered in true_divide
```



In [0]:

```
def f(x, a, t=1):
    if t > 1:
        x = f(x, a, t-1)
    return (a*x-a*x*x)




ax = new_plot()

draw_tree(ax, f, -3, 1, style='.b', start=2./3)
draw_tree(ax, f, 1, 5, style='.b', start=2./3)

show_plot(-0.7, 1.7)
```

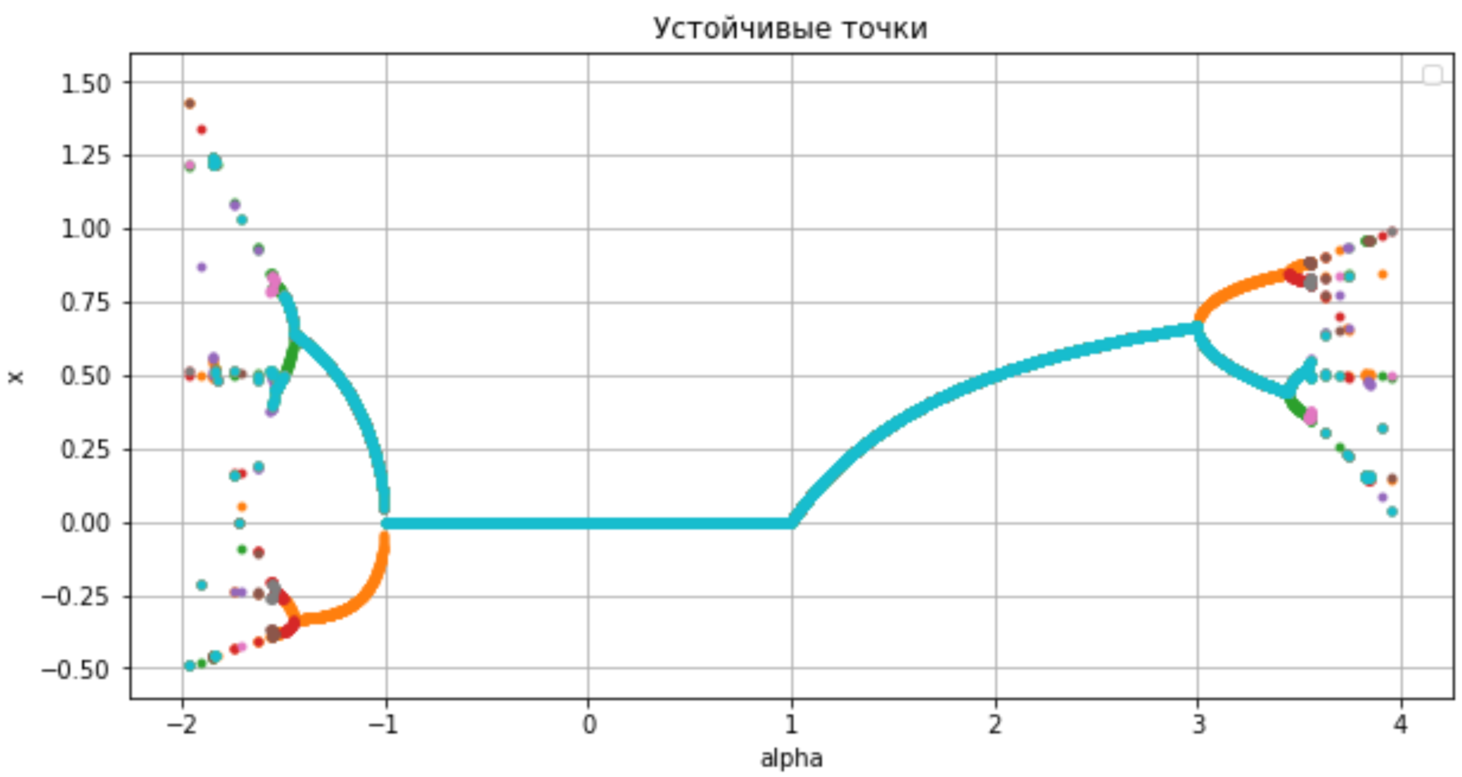
In [0]:

```
def f(x, a, t=1):
    if t == 0:
        return x

    if t > 1:
        x = f(x, a, t-1)

    return a*x-a*x*x
```

```
inf = 100

inf_n = 10
eps = 10e-6
a = np.arange(-2, 1, 0.001)

a_ = []
x_ = []

x_tmp = 2/3
for i in a:
    if x_tmp < -eps or x_tmp > eps:
        x_tmp = 2/3

    for j in range(1000):
        x_tmp = f(x_tmp, i)

#     print(i, x_tmp)

    if not (-inf < x_tmp < inf):
        continue

    for n in range(1, inf_n):
        if -eps < x_tmp - f(x_tmp, i, n) < eps:

            tmp = []
            for j in range(n):
                tmp.append(f(x_tmp, i, j))

#             if n > 0:
#                 print(i, n)

#             print(tmp, n)
            for j in range(inf_n - n):
                tmp.append(tmp[0])

            x_.append(tmp)
            a_.append(i)

            break


# print(len(a_))
# print(len(x_))
x_tmp = 2/3
a = np.arange(1, 5, 0.001)
for i in a:
    if x_tmp < -eps or x_tmp > eps:
        x_tmp = 2/3

    for j in range(1000):
        x_tmp = f(x_tmp, i)

#     print(i, x_tmp)

    if not (-inf < x_tmp < inf):
        continue
```

```
    for n in range(1, inf_n):
        if -eps < x_tmp - f(x_tmp, i, n) < eps:

            tmp = []
            for j in range(n):
                tmp.append(f(x_tmp, i, j))

#               if n > 0:
#                   print(i, n)

#               print(tmp, n)
            for j in range(inf_n - n):
                tmp.append(tmp[0])

            x_.append(tmp)
            a_.append(i)

            break


fig, ax = plt.subplots(figsize=(10,5))
ax.plot(a_, x_, '.')


ax.set(xlabel='alpha', ylabel='x',
       title='Устойчивые точки')
ax.grid()



show_plot(-0.6, 1.6)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: Ru
ntimeWarning: overflow encountered in double_scalars

No handles with labels found to put in legend.
```



Сведение

Сведение одного уравнения к другому можно описать как преобразование координат таким образом, чтобы в обновлённом базисе второе уравнение принимало бы вид первого уравнения в изначальном базисе. В данной работе необходимо свести уравнение параболы к уравнению параболы. Из вида уравнений видно, что поворот базисных векторов не требуется, так что общий вид необходимых преобразований:

$$\begin{cases} x' = kx + l \\ y' = my + n \end{cases}$$

В силу того, что "острота" параболы регулируется коэффициентом при старшей степени, который в уравнении #2 и так является изменяемым параметром, коэффициенты, отвечающие за растяжение/сжатие параболы по осям можно принять за единицу:

$$\begin{cases} x' = x + l \\ y' = y + n \end{cases}$$

применим данную замену :

$$rx'(1 - x') = 1 - ax^2,$$

$$-rx'^2 + rx' = -ax^2 + 1,$$

$$-r(x + l)^2 + r(x + l) - n = -ax^2 + 1,$$

$$-r(x^2 + 2xl + l^2) + r(x + l) - n = -ax^2 + 1,$$

$$-rx^2 - 2rlx - rl^2 + rx + rl - n = -ax^2 + 1,$$

$$\begin{cases} -r = -a \\ -(2l - 1)r = 0 \\ rl - rl^2 - n = 1 \end{cases},$$

$$\begin{cases} r = a \\ l = 0.5 \\ n = 0.25r - 1 \end{cases},$$

таким образом,

$$\begin{cases} x' = x + 0.5 \\ y' = y + 0.25r - 1 \end{cases},$$

$$-rx'(1 - x') = y' \Rightarrow -r(x + 0.5)(1 - (x + 0.5)) - 0.25r + 1 = y$$

Замечание: абсолютно естественно возникновение условия $r = a$, так как исключительно в этом случае гарантируется одинаковая "кривизна" полученных парабол.

Аттрактор, его возникновение и разрушение

Анализ рассматриваемого отображения

Заметим, что бифуркационная диаграмма рассматриваемого в работе отображения имеет фрактальную структуру, что вызвано тем, что все бифуркации, происходящие в системе - последовательно происходящие бифуркации удвоения периода. Таким образом, рассматриваемое отображение содержит странный аттрактор.

Что касается разрушения аттрактора, анализ мультипликатора показывает, что функция #2 имеет экстремум в точке $x = \frac{1}{2}$. При значении $r > 4$ любая точка $x \in (\frac{r-\sqrt{r^2-4r}}{2r}; \frac{r+\sqrt{r^2-4r}}{2r})$ за две итерации попадает в зону $x < 0$, характеризующуюся при $r > 4$ тем, что точка $x = 0$ является неустойчивой. Таким образом, при существовании хаотического аттрактора в системе при $r < 4$ гарантируется его разрушение при $r > 4$.

```
In [0]:
```