Сведение задачи быстродействия к системе дифференциальных уравнений

Формулировка задачи

Решается задача оптимального управления(47), с фиксированными концами и нефиксированным врменем:

$T \to \infty, \ \ |u| \leq 1$

$\dot{x_1} = (u - \frac{2}{3})x_1 + \frac{x_2}{6}, \ \ \dot{x_2} = \frac{2x_1}{3} - \frac{x_2}{6}$

($x_1 > 0, x_2 > 0$ - Задача IA61):

$$\begin{array}{ccccc} & x_1(0) & x_2(0) & x_1(T) & x_2(T) \\ a) & 3.0 & 5.0 & 2.0 & 2.5 \\ b) & 0.1 & 5.0 & 1.0 & 1.5 \\ c) & 0.01 & 7.0 & 2.0 & 3.0 \\ d) & 4 & 7 & 3 & 3.5 \\ at \ \ all) & a_1 & b_1 & a_2 & b_2 \end{array}$$

$(x_1(0), x_2(0), p_1(0), p_2(0))\alpha + (1-\alpha)(x_{10}(0), x_{20}(0), p_{10}(0), p_{20}(0))$

$(x_1(T), x_2(T))\alpha + (1-\alpha)(x_{10}(T), x_{20}(T))$

$(2, 2.5)\alpha + (1-\alpha)(x_{10}(T), x_{20}(T))$

$(p_{10}(0), p_{20}(0))$

$(2, 2.5)0.01 + (1-0.01)(x_{10}(T), x_{20}(T))$

Нужно прислать: отчет и траектории решений(время, посчитанные граничные условия, управление в точке T).

Решение задачи оптимального управления(необходимые условия минимума)

Для сокращения записей все громоздкие вычисления в отчете не приводятся. Также автор позволяет себе неточность обозначений, не всегда выделяя элементы оптимального решения $\hat{x}$ по сравнению с неизвестными x.

Запишем функцию Лагранжа:

$$\mathcal{L} = \int_0^T L dt + l \to \infty$$

$$L = p_1(\dot{x}_1 - ((u - \frac{2}{3})x_1 + \frac{x_2}{6})) + p_2(\dot{x}_2 - (\frac{2x_1}{3} - \frac{x_2}{6}))$$

$$l = \lambda_0 T + \lambda_1(x_1(0) - a_1) + \lambda_2(x_1(T) - a_2) + \lambda_3(x_2(0) - b_1) + \lambda_4(x_2(T) - b_2)$$

И необходимые условия оптимальности:

1) система уравнений Эйлера:

$\dot{p_1} + p_1 u - \frac{2}{3}(p_1 - p_2) = 0$

$\dot{p_2} + \frac{1}{6}(p_1 - p_2) = 0$

2) условия трансверсальности по x_i (не содержательны, но приведены для упрощения некоторых рассуждений):

$p_1(0) = \lambda_1;\;\; p_1(T) = -\lambda_2;\;\; p_2(0) = \lambda_3;\;\; p_2(T) = -\lambda_4;$

3) условие максимума по u(слогаемые не зависящие от u пропущены):

$\min_{u\in[-1,1]}(-p_1 u x_1) = -p_1 \hat{u} x_1 \Rightarrow$

$u = (1)I\{x_1 p_1 > 0\} + (-1)I\{x_1 p_1 < 0\} + (\forall u)I\{x_1 p_1 = 0\}$

4) условия стационарности по времени:

$$\mathcal{L}_T = l_T = \lambda_0 + \lambda_2 u(T) x_1(T) + \frac{1}{6}(x_2(T) - 4x_1(T))(\lambda_2 - \lambda_4) =$$
$$= \lambda_0 - p_1(T)u(T)x_1(T) - \frac{1}{6}(x_2(T) - 4x_1(T))(p_1(T) - p_2(T)) = 0$$

5) условие неотрицательности $\lambda_0 \geq 0$

В силу того что λ_0 ограничен только последними двумя условиями, их можно соединить в одно:

4) ' $p_1(T)u(T)x_1(T) + \frac{1}{6}(x_2(T) - 4x_1(T))(p_1(T) - p_2(T)) \geq 0$

Система дифференциальных уравнений

Итого получена система дифференциальных уравнений:

$$\begin{aligned}
\dot{x_1} &= (u - \tfrac{2}{3})x_1 + \tfrac{x_2}{6}, \\
\dot{x_2} &= \tfrac{2x_1}{3} - \tfrac{x_2}{6} \\
\dot{p_1} &= -p_1 u + \tfrac{2}{3}(p_1 - p_2) \\
\dot{p_2} &= -\tfrac{1}{6}(p_1 - p_2)
\end{aligned}$$

С граничными условиями заданными в таблице в исходной формулировке и дополнительными условиями:

$$p_1(T)u(T)x_1(T) + \frac{1}{6}(x_2(T) - 4x_1(T))(p_1(T) - p_2(T)) \geq 0$$

Где u определено как:

$$u = (1)I\{x_1 p_1 > 0\} + (-1)I\{x_1 p_1 < 0\} + (\forall u)I\{x_1 p_1 = 0\}$$

Дополняя систему недостающими начальными условиями:

$$p_i(0) = \alpha_i, \;\; i \in \{1, 2\}$$

Получаем задачу Коши(отказываясь временно от граничных условий при $t = T$), решение которой в общем случае зависит от параметров α_1, α_2. Итерационно считая решение задачи Коши (методом Рунге-Кутта) $\hat{x}_i^j$(i-ая неизвестная на j-ой итерации) и оптимизируя методом Ньютона функционал потерь(невязку): $X_i^j(\alpha) = x_i^j(T) - \hat{x}_i^j(T, \alpha)$ по T и α находими искомые.

Сведение задачи к однопараметрической

Заметим также несколько моментов, которые помогут упростить задачу:

1) При замене функций $(p_1(t), p_2(t))$ - решений последних двух уравнений, на пропорциональные $(\lambda p_1(t), \lambda p_2(t)), \;\; \lambda > 0$ получаются новые решения, характер которых, в смысле момента переключения управления не меняется. Таким образом, при пристрелке можно фиксировать начальное значение $p_2(0)$ на значениях $1, 0$ и -1 и изменять только значения $p_1(0)$.

2) Заметим, в силу отсутствия фиксации времемни за невязку можно принять минимульный вектор от фазовой кривой x_1, x_2 до краевого значения $x_1(T), x_2(T)$. Обозначим невязку $X(p_1(0), p_2(0))$.

Таким образом, задача практически сводится к однопараметрической (при трех случаях $p_2(0) \in \{-1, 0, 1\}$)- необходимо выбрать $p_1(0)$ так чтобы найти ноль указанной невязки. Заметим также, что невязка - система из двух уравнений, на одну переменную. На деле возможны два подхода - поиск минимума нормы невязки и поиск нуля одного из уравнений, а затем проверка второго. Мы придерживаемся второго подхода.

Решать будем Рунге-Кутта 4ого порядка:

$$\mathbf{k}_1 = \mathbf{f}(t_n, \mathbf{y}_n),$$

$$\mathbf{k}_2 = \mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right),$$

$$\mathbf{k}_3 = \mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right),$$

$$\mathbf{k}_4 = \mathbf{f}(t_n + h, \mathbf{y}_n + h\,\mathbf{k}_3).$$

$$\mathbf{y}_{n+1} = \varphi(t_n, \mathbf{y}_n) = \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$$

Точность метода порядка $o(h^4)$. Для простаты по умолчанию считаем $h = T/1000$

In [1]:

```
#вспомогательный код
import rahmodule
import numpy as np
from matplotlib import pyplot as plt
from tqdm.notebook import tqdm
```

In [2]:

```
# начальные условия:
boundary_conditions = [[[3, 5], [2, 2.5]], [[0.1, 5], [1, 1.5]], [[0.01, 7.0], [2.0,
```

```
#полученная система дифференциальных уравнений:
def u(x1, p1):
        if (x1*p1 >= 0):
            return 1
        else:
            return -1

def f(t, y):
    x1, x2, p1, p2 = y
    u_ = u(x1, p1)
    #print(y, u_)

    xx1 = (u_ - 2/3)*x1 + x2/6
    xx2 = 2/3*x1 - x2/6
    pp1 = -p1*u_ + 2/3*(p1 - p2)
    pp2 = -1/6*(p1 - p2)
    return np.array([xx1, xx2, pp1, pp2])
```

Решение для первых граничных условий

Подробно опишем техническую часть решения для первых граничных условий. Так как задачу удалось свести к однопараметрической, то основная часть решения - подбор инициального значения, для запуска оптимизации. Напомним, что оптимизируемая функция - расстояние от правого краевого значения, до ближайшей точки траектории; параметр оптимизации $p_1(0)$:

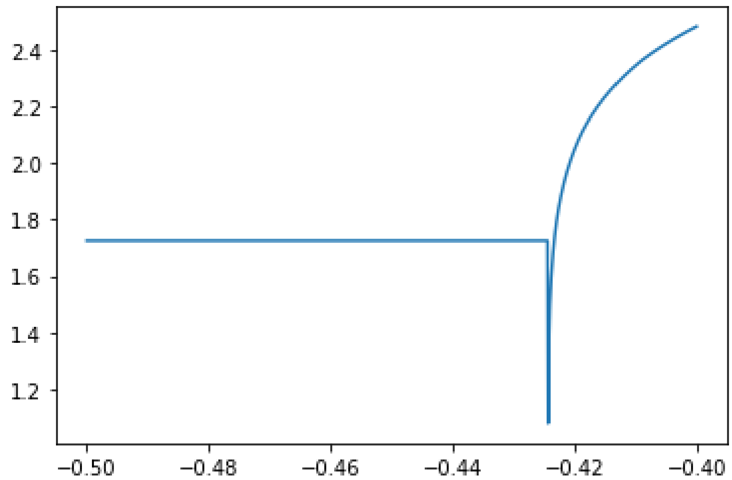
$$X(p_{10}, p_{20}) = |\hat{x}(1, p_{10}, p_{20}) - x(1)|$$

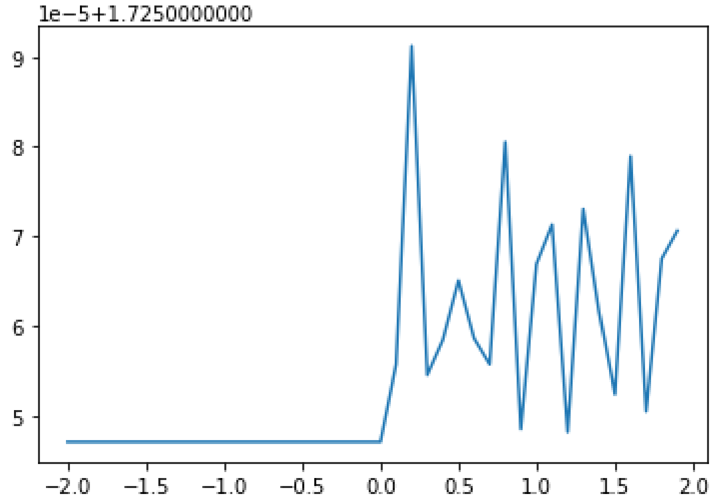
$$(p_{1initial}, p_{2initial})(0) = (p_{10}, p_{20}) = \text{argmin}_{p_{10} \in (-1,1);\ p_{20} \in \{-1,1\}}(X(p_{10}, p_{20}))$$

Ниже приведены графики X от p_{10}, при $p_{20} = -1$ и $p_{20} = 1$ соответственно. Отметим отдельно резкий характер зависимости траекторий решения системы(в смысле оптимизируемой функции) от p_{10} в окрестности изменения характера решений(в момент когда решение при постоянном управлении переходит в решение с переключающимся управлением).

In [4]:
```
cur_range = np.arange(-0.5, -0.4, 0.0001)
```

In [5]:
```
X_1 = rahmodule.initial_search(boundary_conditions=boundary_conditions[0], cur_range
```

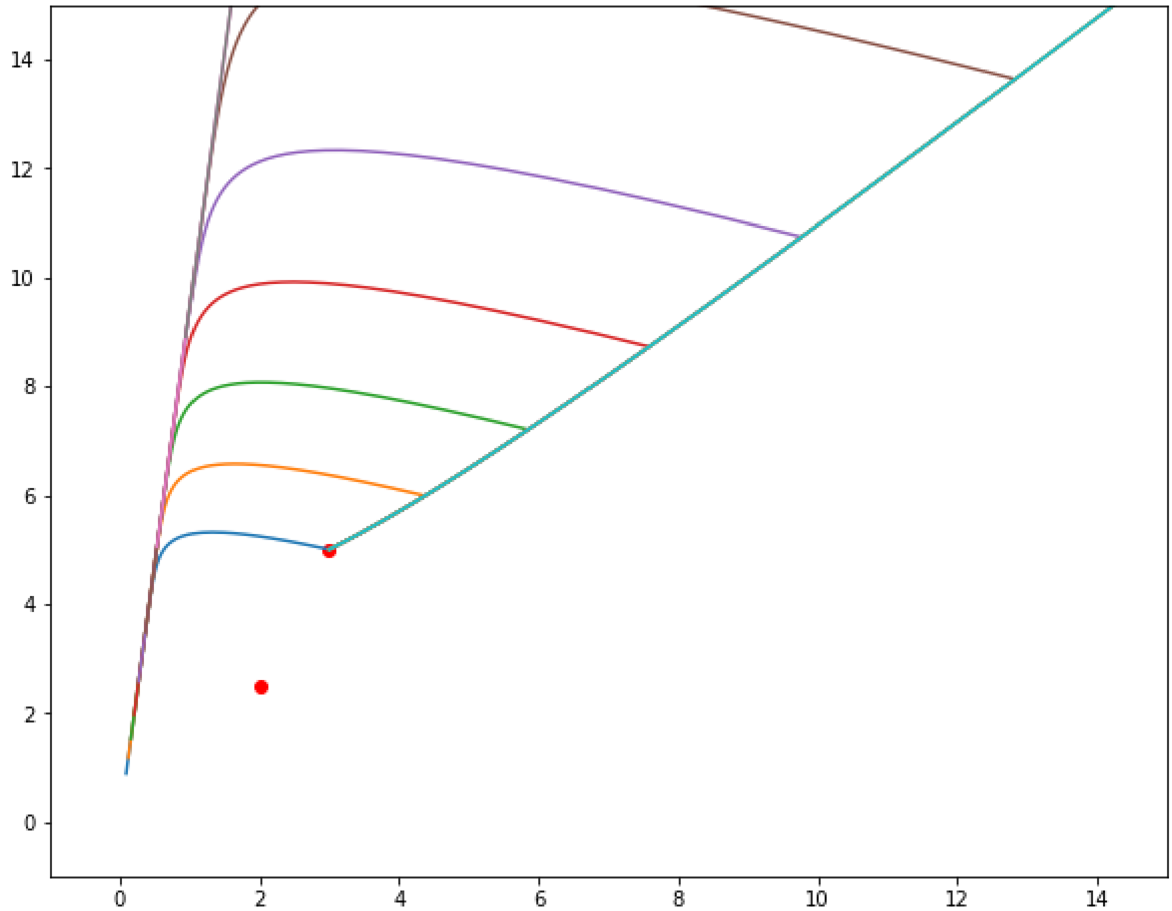




In [6]:
```
p10 = cur_range[np.argmin(X_1[0])]
#p10 = -0.4244000000000193
p20 = -1
```

За отсутствием аналитического доказательства, что при $p_{20} = 1$ решений нет, будем обсчитывать этот случай каждый раз - с меньшим шагом, но на большем интервале. Заметим, что колебания невязки при $p_{20} = 1$ происходят в пятом и далее знаках, точные причины появления этих флуктуаций подробно изучены не были - фактически, в данном конкретном случае при $p_{20} = 1$, $p_{10} > 0$ траектории (x_1, x_2)(первое изображение ниже) имеют асимптотой траекторию при $p_{20} = 1$, $p_{10} < 0$(вторoе изображение ниже) (она единственна, т.к. не происходит переключения управления).

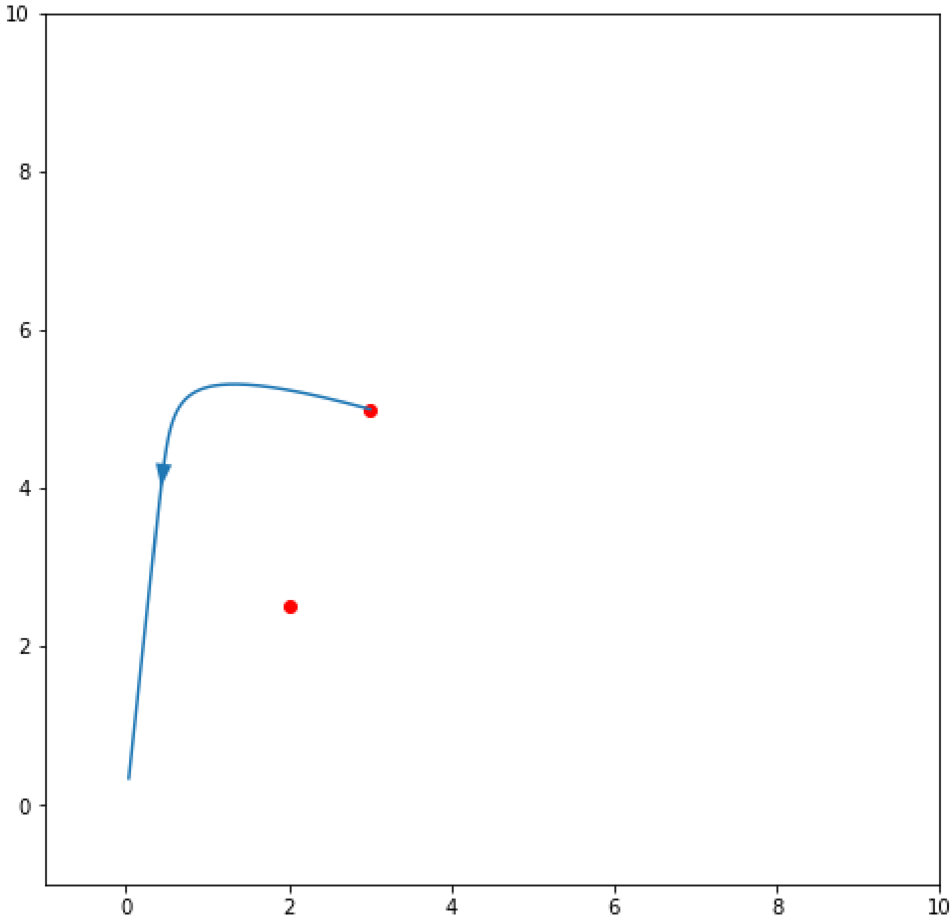
In [7]:
```
rahmodule.random_strike_portrait(np.arange(0, 5, 0.5), boundary_conditions=boundary_
```



In [8]:

```
rahmodule.checksol(p10=-0.5, p20=1, boundary_conditions=boundary_conditions[0], T=30
```

```
Данные невязки , невязка, правый конец решения):
Модуль невязки: 1.7250470828953866
Вектор невязки: [ 1.71545128 -0.18169847]
Правый конец решения: [ 2.84548717e-01  2.68169847e+00 -1.92718419e+06  2.04490287e+
05]
Plase plot of x1, x2
```



Далее, используется однопараметрический градиентный спуск по p_{10}, с динамически уменьшающимся шагом. Шаг уменьшается до 0.5e-16 - причина тому резкая зависимость траекторий решений от параметра p_{10} в окрестности изменения характера решений.

In [9]:

```
ans = rahmodule.idiot_optimizer(rahmodule.LOSS, [p10], others=[boundary_conditions[0
print(f"Итоговое значение параметра p10: {ans[0]}")
#ans = [-0.4244289]
```

```
Step 0, point [-0.42441]: 1.016262967901592
Step 1, point [-0.42442]: 0.9106894435272284
Step 2, alpha changed 5e-06
Step 2, point [-0.424425]: 0.803983128445499
Step 3, alpha changed 2.5e-06
Step 3, point [-0.4244275]: 0.6839477496257191
Step 4, alpha changed 1.25e-06
Step 4, point [-0.42442875]: 0.46463940652463237
Step 5, alpha changed 6.25e-07
Step 5, alpha changed 3.125e-07
Step 5, alpha changed 1.5625e-07
Step 5, alpha changed 7.8125e-08
Step 5, point [-0.42442883]: 0.4037253373030261
Step 6, alpha changed 3.90625e-08
Step 6, point [-0.42442887]: 0.34470340777301994
Step 7, alpha changed 1.953125e-08
Step 7, point [-0.42442889]: 0.28378822230315975
Step 8, alpha changed 9.765625e-09
Step 8, point [-0.4244289]: 0.21052799363511487
Step 9, alpha changed 4.8828125e-09
Step 9, alpha changed 2.44140625e-09
Step 9, point [-0.4244289]: 0.16537302446619234
Step 10, alpha changed 1.220703125e-09
Step 10, point [-0.4244289]: 0.11626064388332995
```

```
 Step 11, alpha changed 6.103515625e-10
 Step 11, point [-0.4244289]: 0.043069556633494835
 Step 12, alpha changed 3.0517578125e-10
 Step 12, alpha changed 1.52587890625e-10
 Step 12, alpha changed 7.62939453125e-11
 Step 12, point [-0.4244289]: 0.014777618953562355
 Step 13, alpha changed 3.814697265625e-11
 Step 13, alpha changed 1.9073486328125e-11
 Step 13, point [-0.4244289]: 0.0023795837409014446
 Step 14, alpha changed 9.5367431640625e-12
 Step 14, alpha changed 4.76837158203125e-12
 Step 14, alpha changed 2.384185791015625e-12
 Step 14, alpha changed 1.1920928955078126e-12
 Step 14, alpha changed 5.960464477539063e-13
 Step 14, point [-0.4244289]: 0.0016847832234781074
 Step 15, alpha changed 2.9802322387695315e-13
 Step 15, point [-0.4244289]: 0.0016598519769005263
 Step 16, alpha changed 1.4901161193847657e-13
 Step 16, point [-0.4244289]: 0.0015554739901131472
 Step 17, alpha changed 7.450580596923829e-14
 Step 17, alpha changed 3.7252902984619144e-14
 Step 17, alpha changed 1.8626451492309572e-14
 Step 17, alpha changed 9.313225746154786e-15
 Step 17, point [-0.4244289]: 0.0015551575679511273
 Step 18, alpha changed 4.656612873077393e-15
 Step 18, alpha changed 2.3283064365386965e-15
 Step 18, point [-0.4244289]: 0.0015551458409863062
 Step 19, alpha changed 1.1641532182693482e-15
 Step 19, point [-0.4244289]: 0.0015551421840265074
 Step 20, alpha changed 5.820766091346741e-16
 Step 20, point [-0.4244289]: 0.0015551416840664677
 Step 21, alpha changed 2.9103830456733706e-16
 Step 21, alpha changed 1.4551915228366853e-16
 Step 21, point [-0.4244289]: 0.0015551416542219688
 Step 22, alpha changed 7.275957614183426e-17
 Step 22, point [-0.4244289]: 0.0015551416196893037
 Step 23, alpha changed 3.637978807091713e-17
 Step 23, alpha changed 1.8189894035458566e-17
 Step 23, alpha changed 9.094947017729283e-18
 Step 23, alpha changed 4.547473508864616e-18
 Step 23, alpha changed 2.2737367544323208e-18
 Step 23, alpha changed 1.1368683772161604e-18
 Step 23, alpha changed 5.684341886080802e-19
Interrupted! Step less than 1e-18!
Итоговое значение параметра p10: -0.424428900852651
```

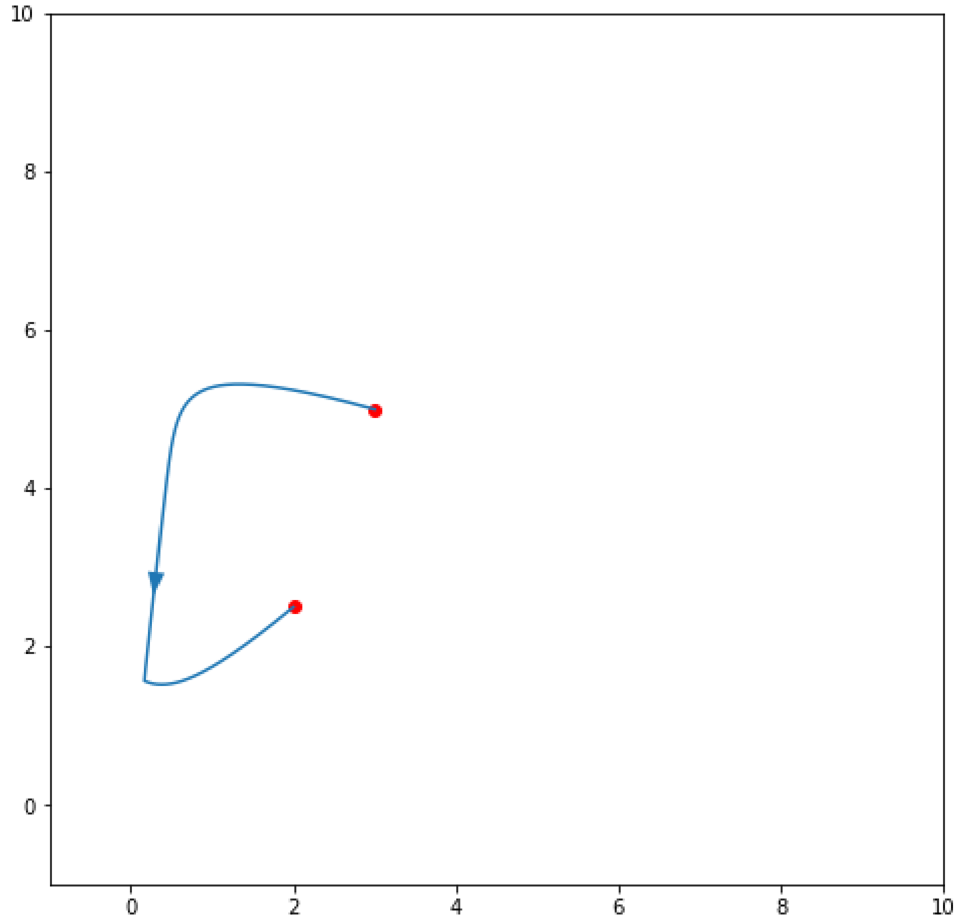
Итоговые характеристики найденного решения и графики траекторий приведены ниже.

Далее приводятся аналогичные построения для прочих краевых условий. Сами траектории решений содержатся в файлах bc{i}.txt, где $i \in \{0, \dots, 3\}$ - номер пункта.

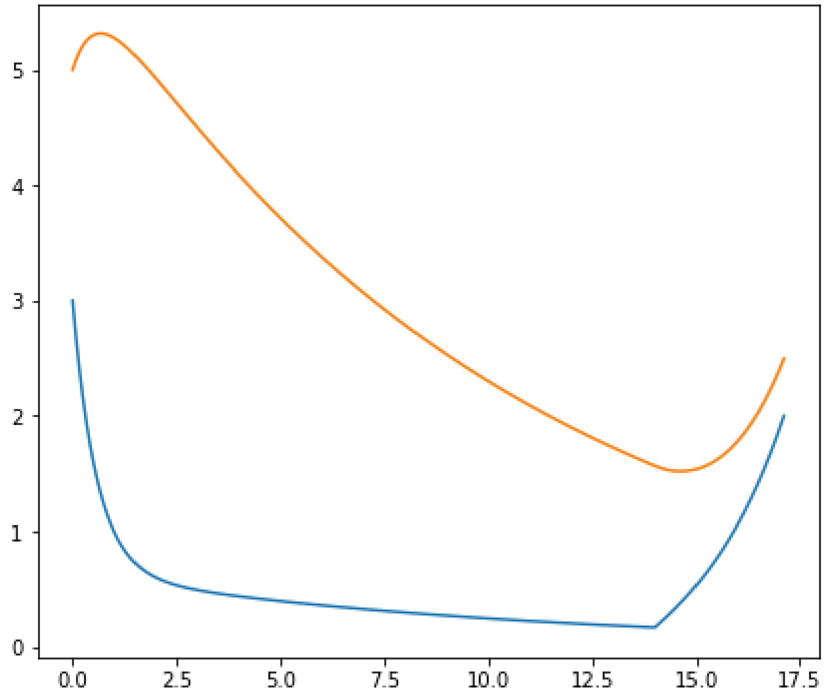
In [13]:

```
rahmodule.checksol(p10=ans[0], p20=-1, boundary_conditions=boundary_conditions[0], s
```

```
Данные невязки , невязка, правый конец решения):
Модуль невязки: 0.0015551466632049364
Вектор невязки: [-0.0012114   0.00097518]
Правый конец решения: [ 2.0012114   2.49902482  8.33106667 -9.1817865 ]
Plase plot of x1, x2
```



Plot of x1, x2



In [14]:

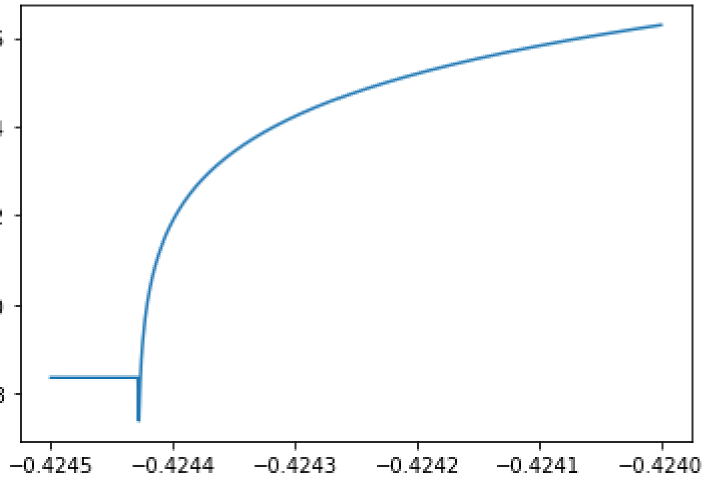
```
def pipesearch(i, cur_range = np.arange(-0.5, -0.4, 0.00005)):
    #подобрано, вычислением с таким же шагом, но на интервале (-2, 2),
    #но результаты и графики были утеряны и не воспроизодились

    X = rahmodule.initial_search(boundary_conditions=boundary_conditions[i], cur_ran
    p10 = cur_range[np.argmin(X[0])]
    print(f"Initial p10: {p10}")
    p20 = -1
    ans = rahmodule.idiot_optimizer(rahmodule.LOSS, [p10], others=[boundary_conditio
```

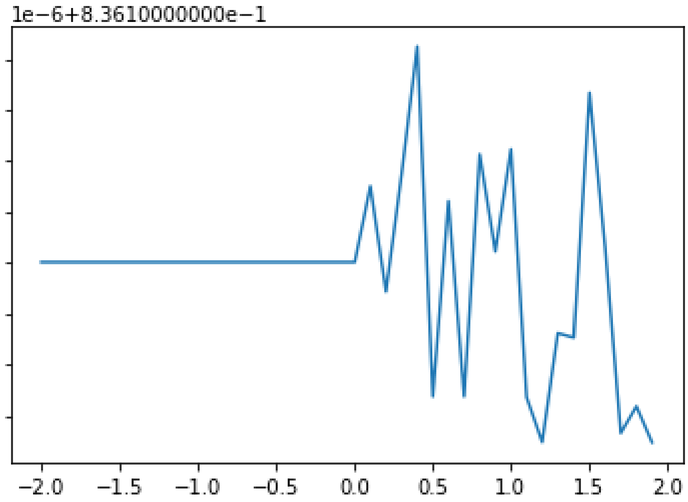
```
    print(f"Optimized p10: {ans[0]}")
    rahmodule.checksol(p10=ans[0], p20=-1, boundary_conditions=boundary_conditions[i
```

Решение для вторых граничных условий

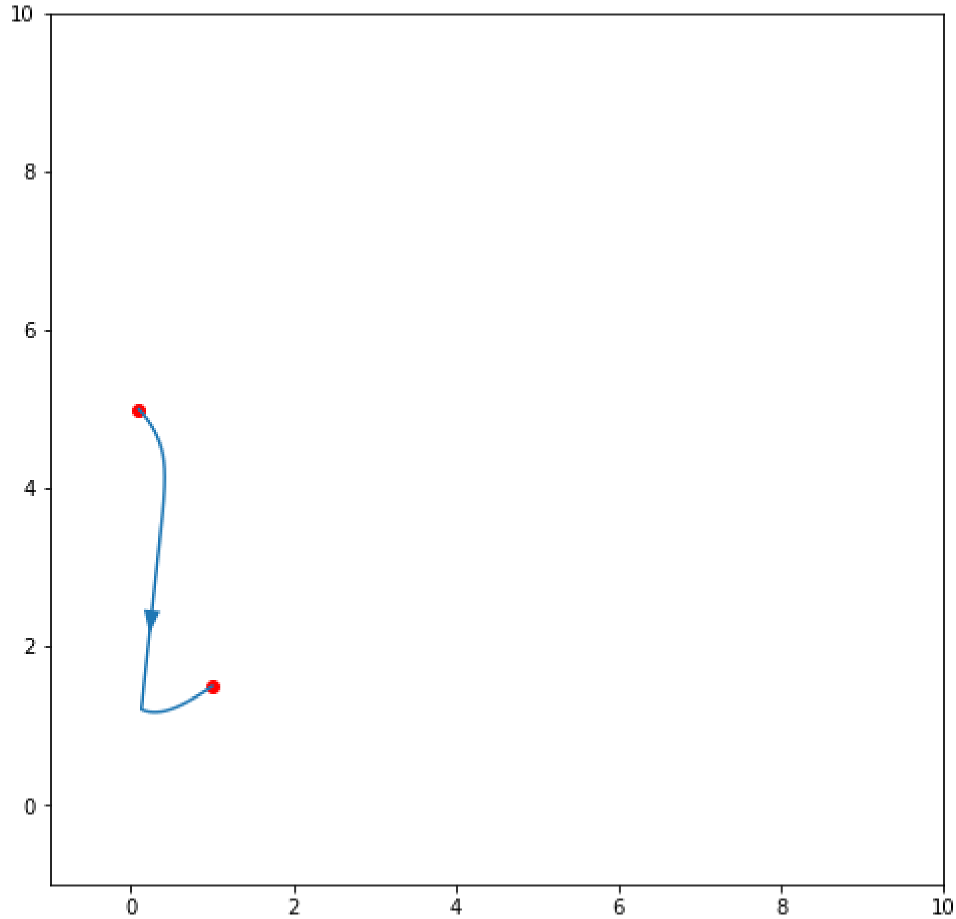
In [15]:

```
pipesearch(1, cur_range=np.arange(-0.4245, -0.424, 1e-6))
```

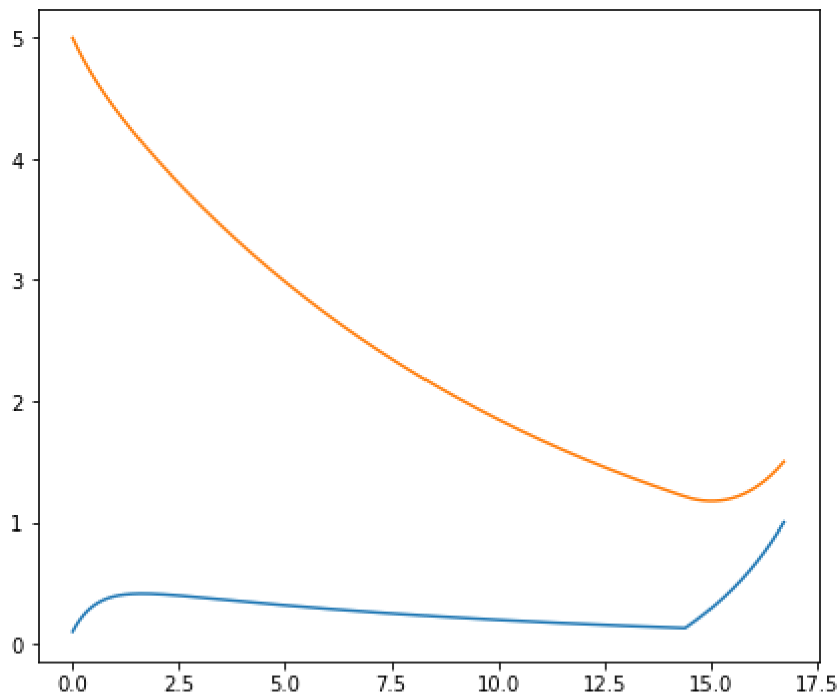





```
Initial p10: -0.4244280000000019

Interrupted! Step less than 1e-18!
Optimized p10: -0.4244289008745079
Данные невязки , невязка, правый конец решения):
Модуль невязки: 0.00042035818775340355
Вектор невязки: [-0.00034847  0.0002351 ]
Правый конец решения: [ 1.00034847  1.4997649   6.17575027 -7.48140271]
Plase plot of x1, x2
```



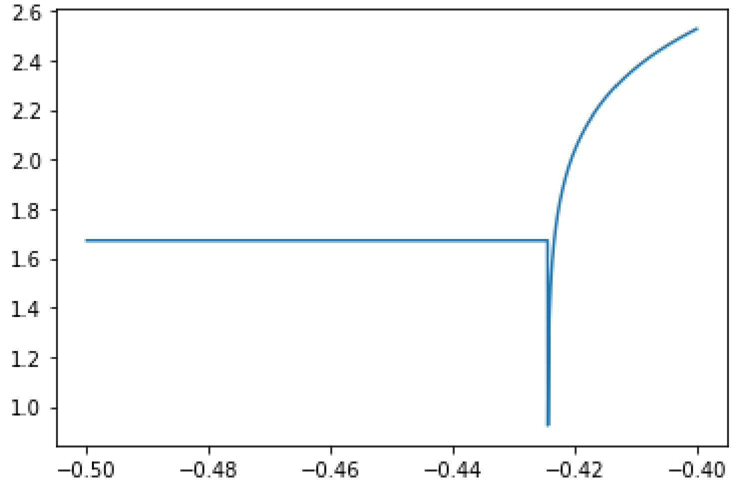
Plot of x1, x2

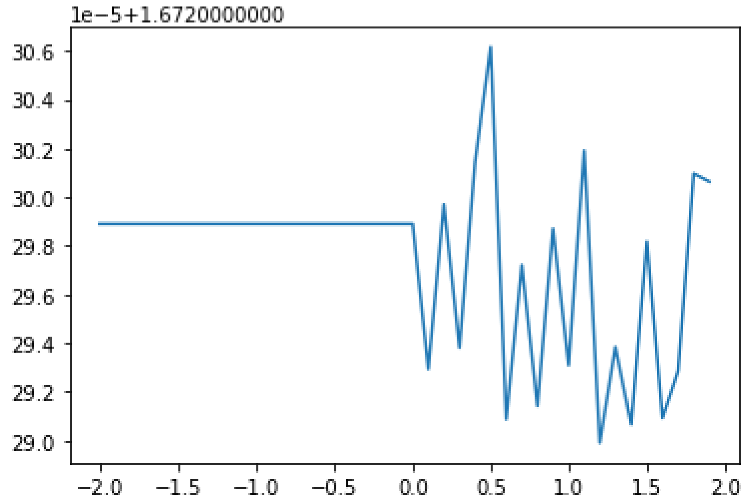


Решение для третьих граничных условий

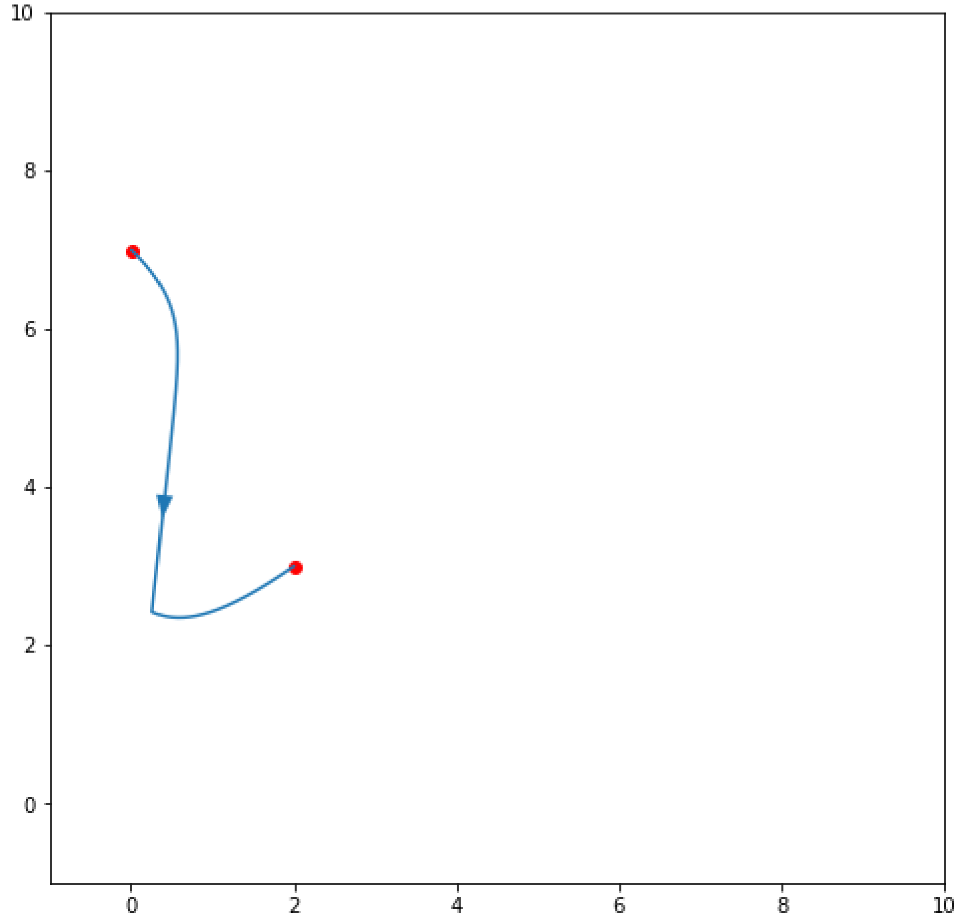
In [16]:

```
pipesearch(2)
```

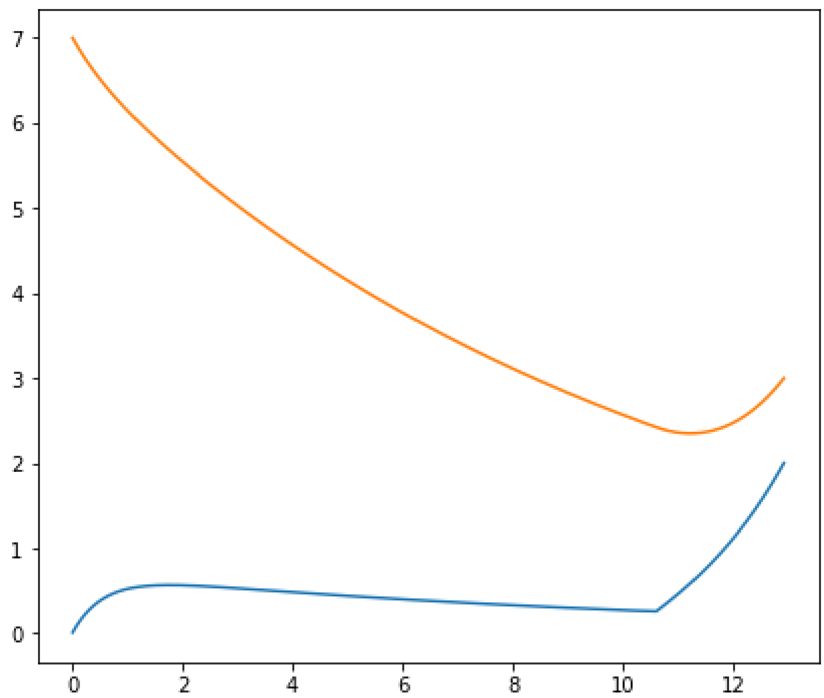





```
Initial p10: -0.4244000000000083

Interrupted! Step less than 1e-18!
Optimized p10: -0.4244288888532254
Данные невязки , невязка, правый конец решения):
Модуль невязки: 0.0008406971035597048
Вектор невязки: [-0.00070043  0.00046493]
Правый конец решения: [ 2.00070043  2.99953507  4.28922928 -5.196041   ]
Plase plot of x1, x2
```

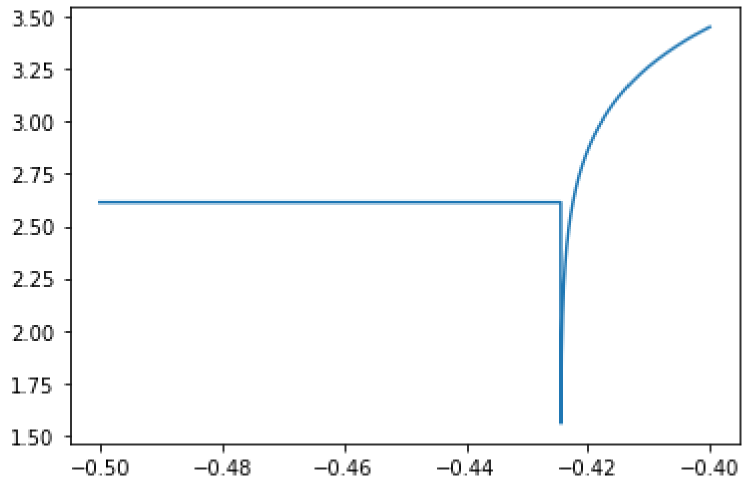


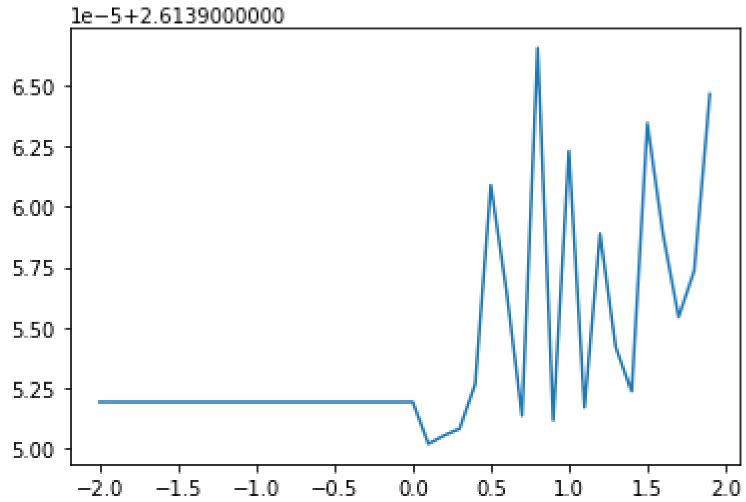

```
Plot of x1, x2
```



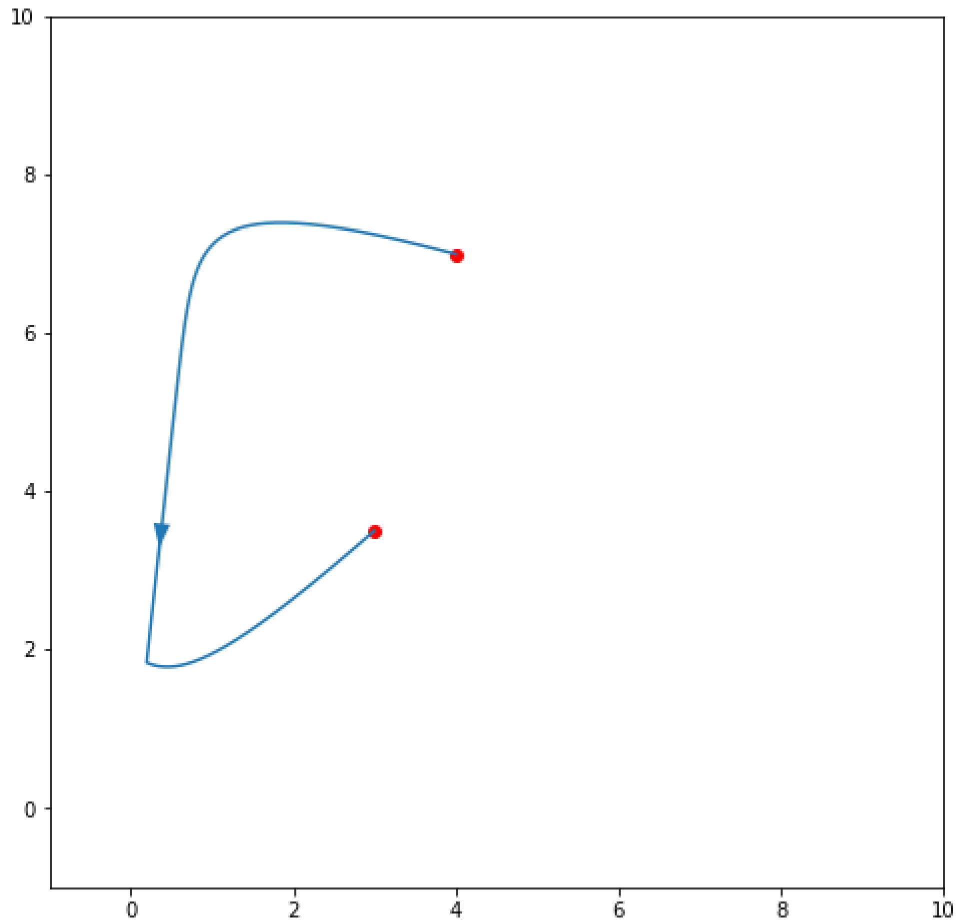
Решение для четвертых граничных условий

In [17]:
```
pipesearch(3)
```






```
Initial p10: -0.4244000000000083

Interrupted! Step less than 1e-18!
Optimized p10: -0.42442890089547874
Данные невязки , невязка, правый конец решения):
Модуль невязки: 0.00046449512377119576
Вектор невязки: [ 0.00034486 -0.00031117]
Правый конец решения: [  2.99965514   3.50031117  11.78392928 -12.58329259]
Plase plot of x1, x2
```



Plot of x1, x2