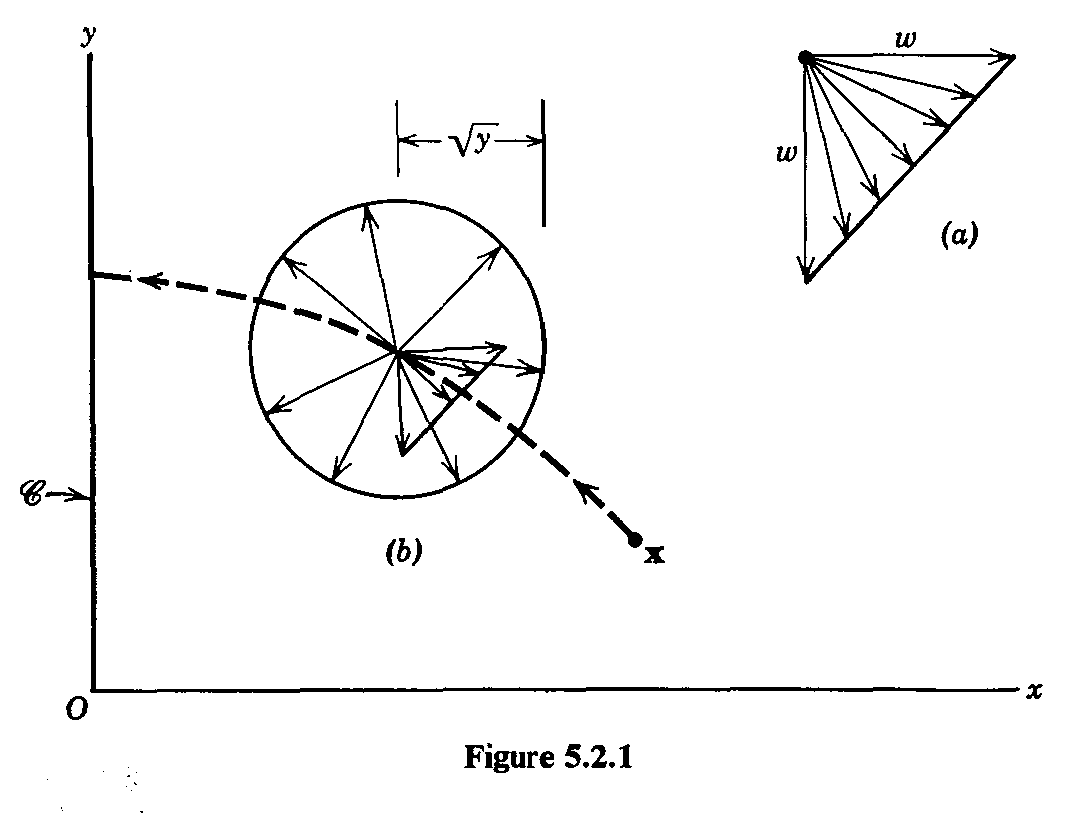
Dolichobrachistochrone



























 with 



% Initial conditions for system

problem.states.x0=[0.5 1.5 -1 -1];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];

4991r-1.1456906e+01 4.60e-01 2.02e+03 -2.5 4.80e-08 10.6 1.00e+00 1.00e+00h 1

4992r-1.1456906e+01 4.60e-01 1.73e+04 -2.5 3.63e-08 12.0 1.00e+00 5.00e-01h 2

4993r-1.1456906e+01 4.60e-01 1.19e+04 -2.5 3.62e-08 11.5 1.00e+00 1.00e+00h 1

4994r-1.1456906e+01 4.60e-01 4.38e+03 -2.5 3.62e-08 11.0 1.00e+00 1.00e+00h 1

4995r-1.1456906e+01 4.60e-01 9.90e+03 -2.5 3.61e-08 11.4 1.00e+00 1.00e+00h 1

4996r-1.1456906e+01 4.60e-01 3.85e+03 -2.5 3.61e-08 10.9 1.00e+00 1.00e+00h 1

4997r-1.1456906e+01 4.60e-01 1.81e+03 -2.5 6.10e-08 10.5 1.00e+00 1.00e+00h 1

4998r-1.1456906e+01 4.60e-01 3.01e+03 -2.5 3.63e-08 10.9 1.00e+00 1.00e+00h 1

4999r-1.1456906e+01 4.60e-01 1.66e+03 -2.5 6.29e-08 10.4 1.00e+00 1.00e+00h 1

iter objective inf\_pr inf\_du lg(mu) ||d|| lg(rg) alpha\_du alpha\_pr ls

5000r-1.1456906e+01 4.60e-01 2.70e+03 -2.5 3.66e-08 10.8 1.00e+00 1.00e+00h 1

Number of Iterations....: 5000

(scaled) (unscaled)

Objective...............: -1.1456905723240167e+01 -1.1456905723240167e+01

Dual infeasibility......: 2.7024780959154923e+03 2.7024780959154923e+03

Constraint violation....: 4.6007206172071197e-01 4.6007206172071197e-01

Complementarity.........: 3.2411140409095403e-03 3.2411140409095403e-03

Overall NLP error.......: 2.2908179444644534e+00 2.7024780959154923e+03

Number of objective function evaluations = 9189

Number of objective gradient evaluations = 165

Number of equality constraint evaluations = 9291

Number of inequality constraint evaluations = 9291

Number of equality constraint Jacobian evaluations = 5107

Number of inequality constraint Jacobian evaluations = 5107

Number of Lagrangian Hessian evaluations = 5000

Total CPU secs in IPOPT (w/o function evaluations) = 113.098

Total CPU secs in NLP function evaluations = 90.175

EXIT: Maximum Number of Iterations Exceeded.

computation time:

203.3048

minimized cost:

-11.4569

Maximum absolute local error:

0.0713 0.0518 0 0.0054

Maximum relative local error:

0.0060 0.0058 0 0.0027

Maximum absolute constraint violation:

0 0 0 0 0 0 0 0 0 0

Number of active constraints:

1

Determine sparsity structure

Formatting matrices for the hermite-simpson approximation

Generate finite-difference pertubation vectors

This is Ipopt version 3.12.9, running with linear solver ma57.

% Initial conditions for system

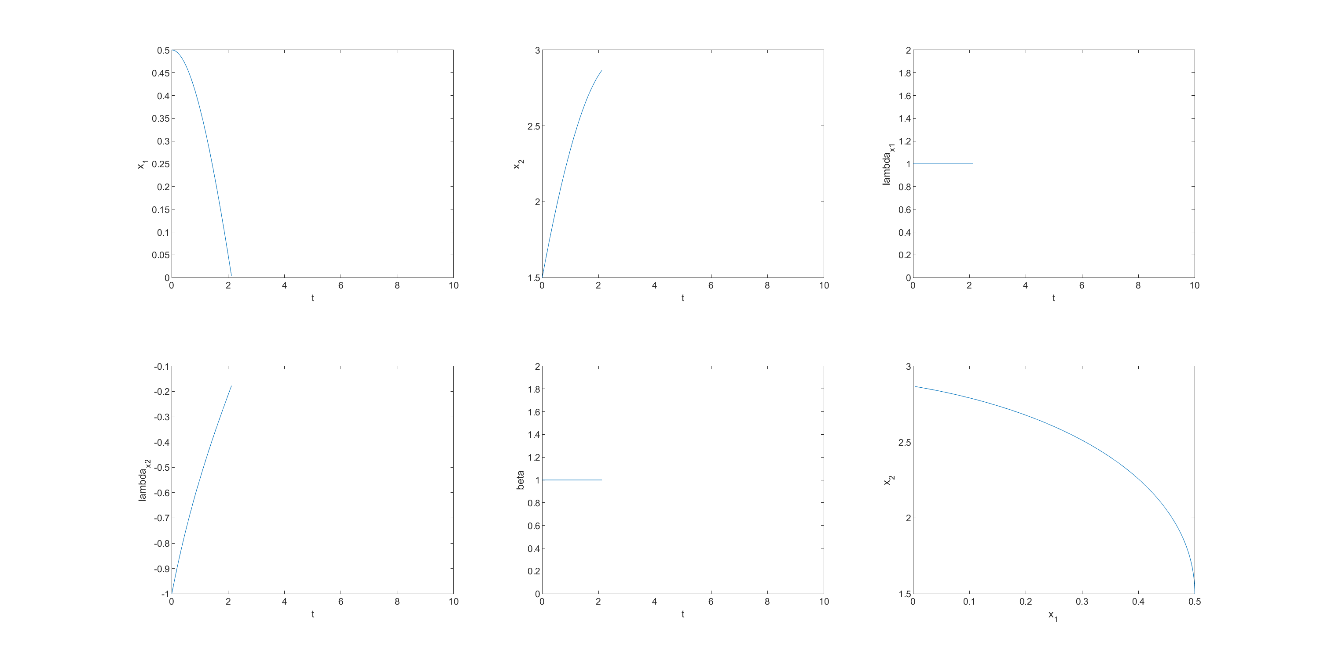
problem.states.x0=[0.5 1.5 1 -1];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];

boundaryCost = -tf;



和解析结果接近

然而，改变初始设定

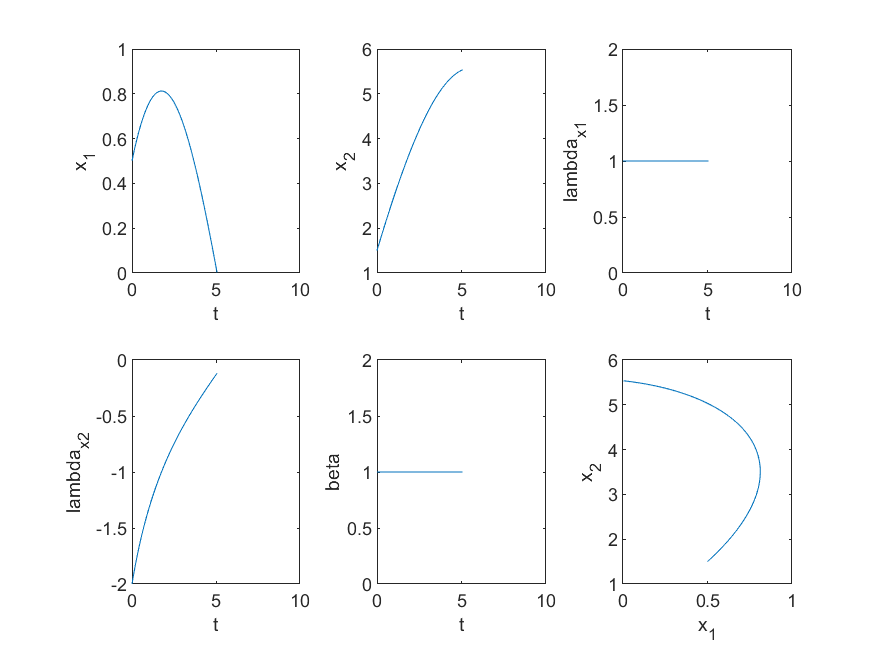
% Initial conditions for system

problem.states.x0=[0.5 1.5 1 -2];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];



初始值改变，结果也发生很大的改变。实际上初值应该是一个变量，不应该是一个固定的值。

% Initial conditions for system

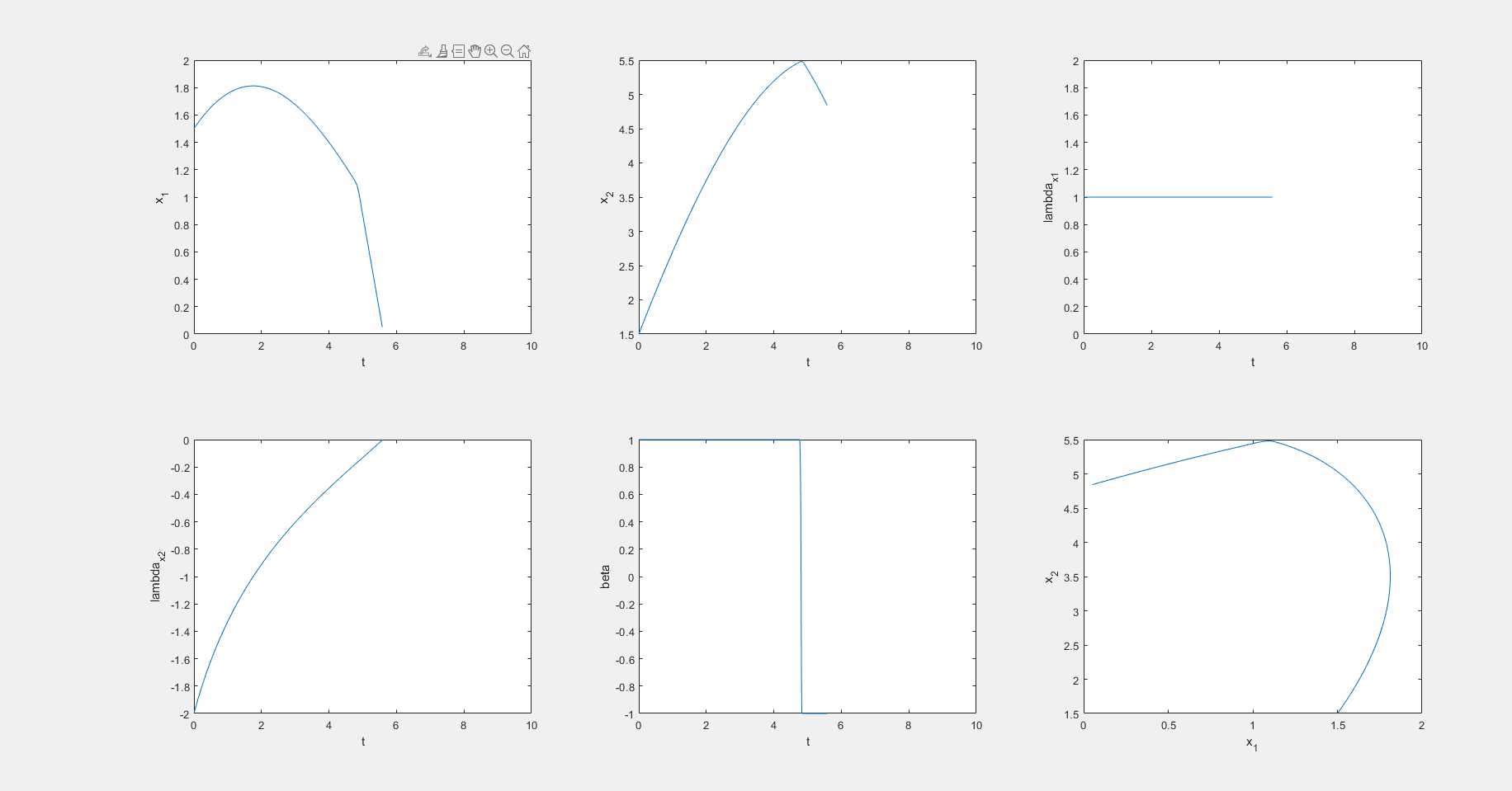
problem.states.x0=[1.5 1.5 1 -2];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];

boundaryCost = -tf;



时间长度上面很是接近，但是控制的切换方向反了。

有什么东西的方向反了。

% Initial conditions for system

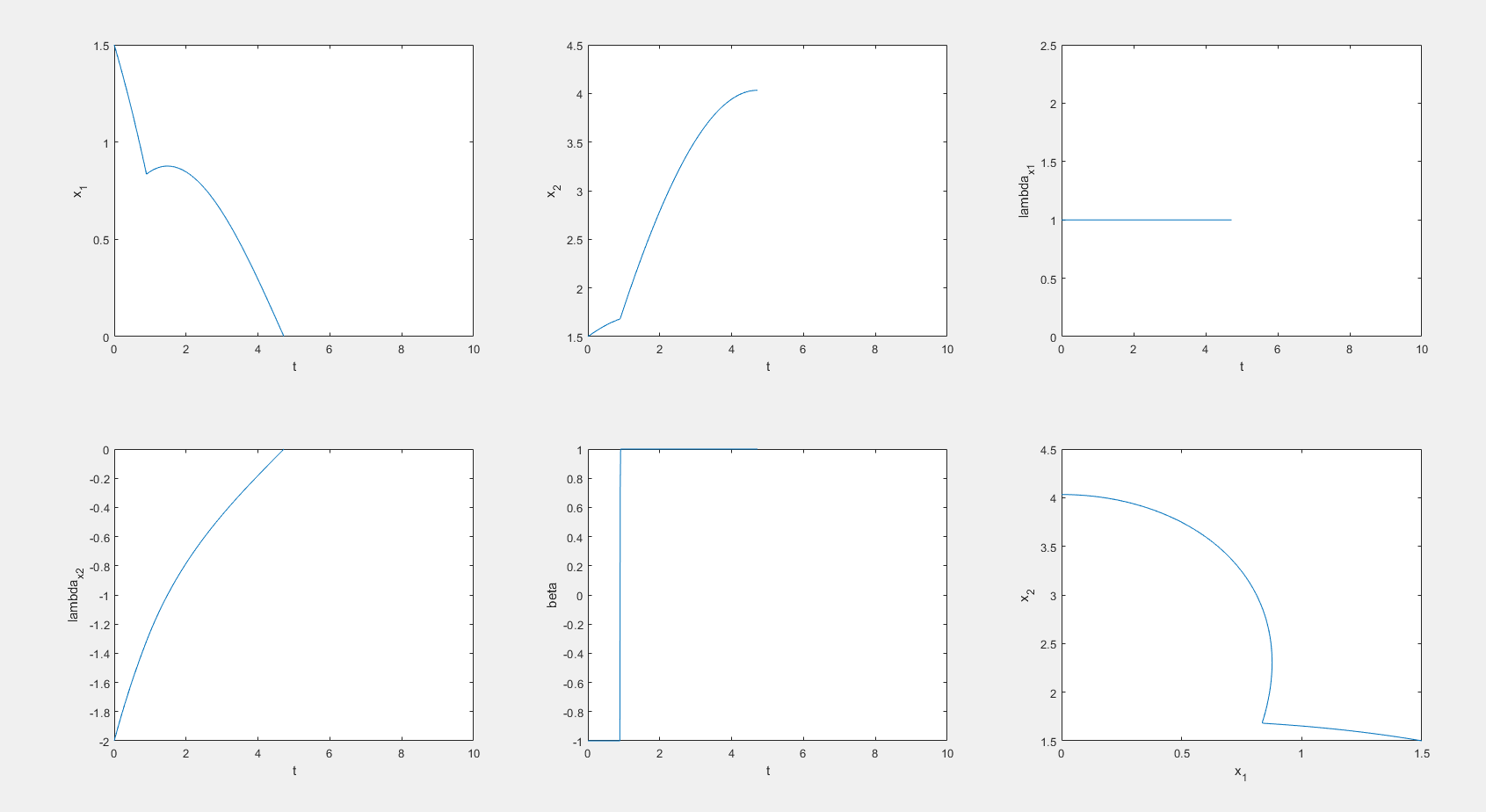
problem.states.x0=[1.5 1.5 1 -2];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];

boundaryCost = tf;



% Initial conditions for system

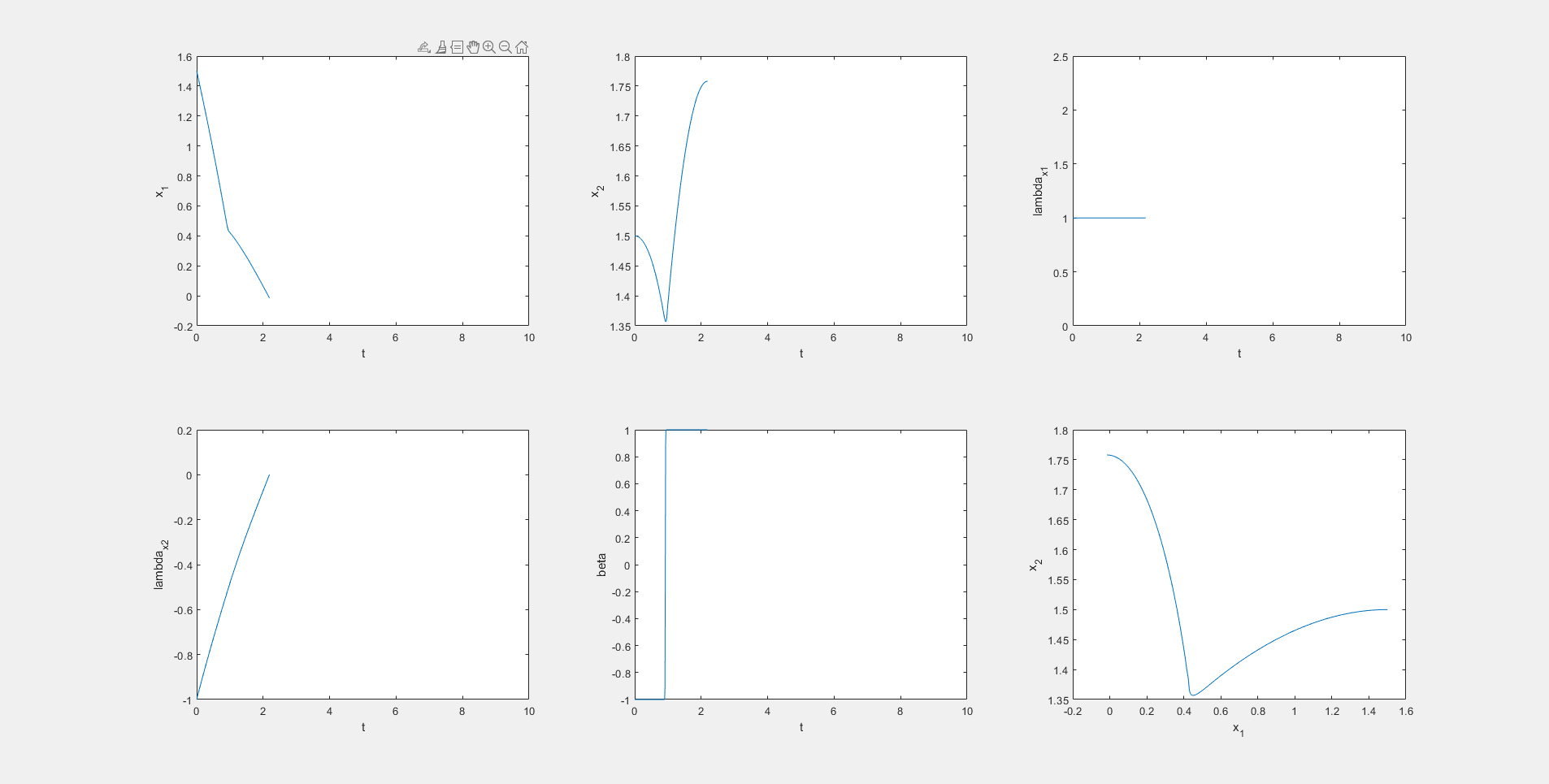
problem.states.x0=[1.5 1.5 1 -1];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];

boundaryCost = tf;



% Initial conditions for system

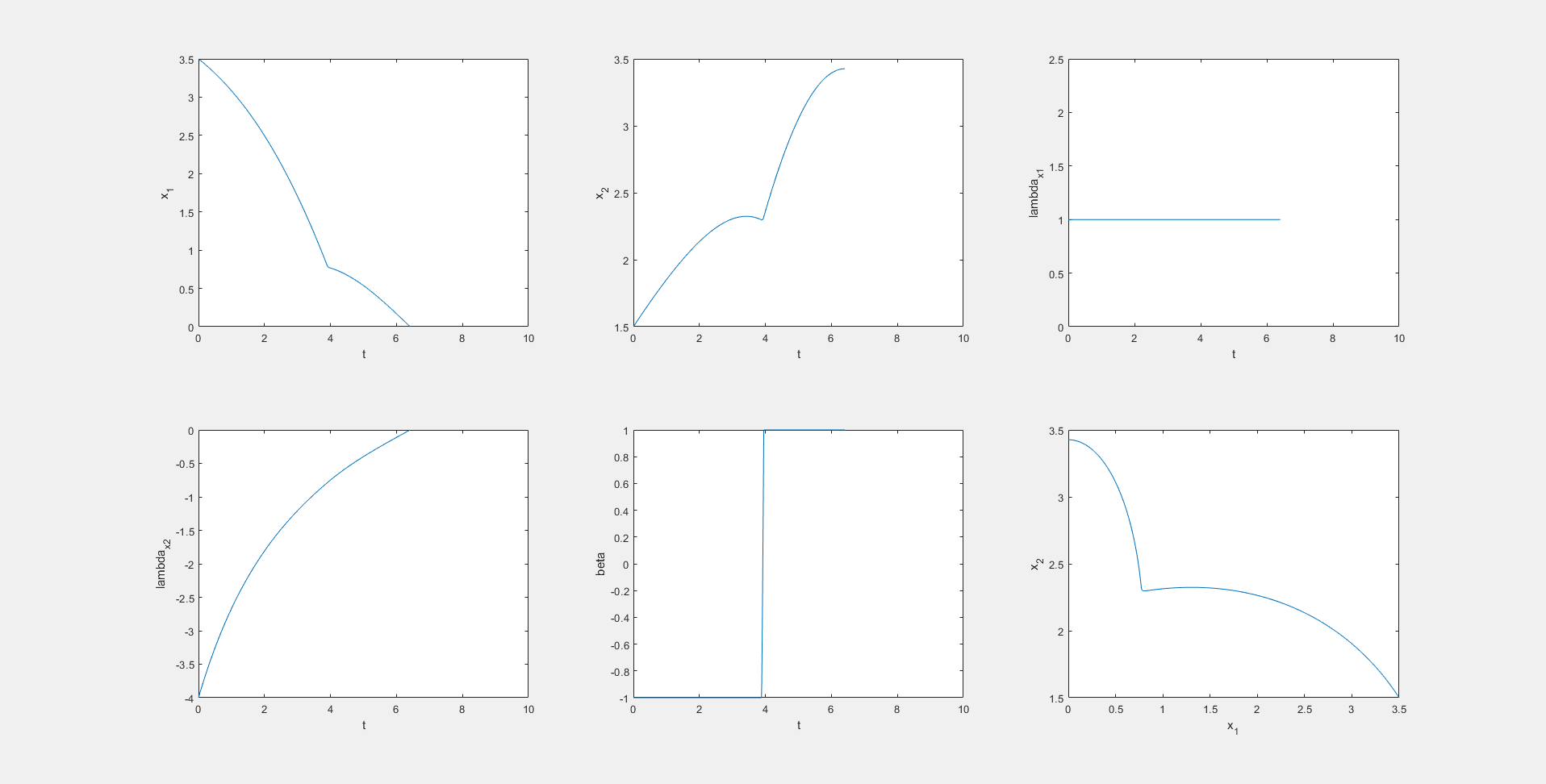
problem.states.x0=[3.5 1.5 1 -4];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[3.5 1.5 -10 -10];

problem.states.x0u=[3.5 1.5 10 10];

boundaryCost = tf;



终于知道为什么

boundaryCost = tf;

才可以得到正确答案了

系统动态写错了，stupid

%------------- BEGIN CODE --------------

x\_1 = x(:,1);x\_2 = x(:,2);lambda\_x1 = x(:,3);lambda\_x2 = x(:,4);

beta = u(:,1);

sinalph = -lambda\_x2./sqrt(lambda\_x1.^2+lambda\_x2.^2);

cosalph = -lambda\_x1./sqrt(lambda\_x1.^2+lambda\_x2.^2);

dx(:,1) = sqrt(x\_2) .\* cosalph + (beta + 1)./2;

dx(:,2) = sqrt(x\_2) .\* sinalph + (beta - 1)./2;

dx(:,3) = 0;

dx(:,4) = -(lambda\_x1.\*cosalph+lambda\_x2.\*sinalph)./(2.\*sqrt(x\_2));

写成

dx(:,1) = sqrt(2) .\* cosalph + (beta + 1)./2;

dx(:,2) = sqrt(2) .\* sinalph + (beta - 1)./2;

结果不一样

% Initial conditions for system

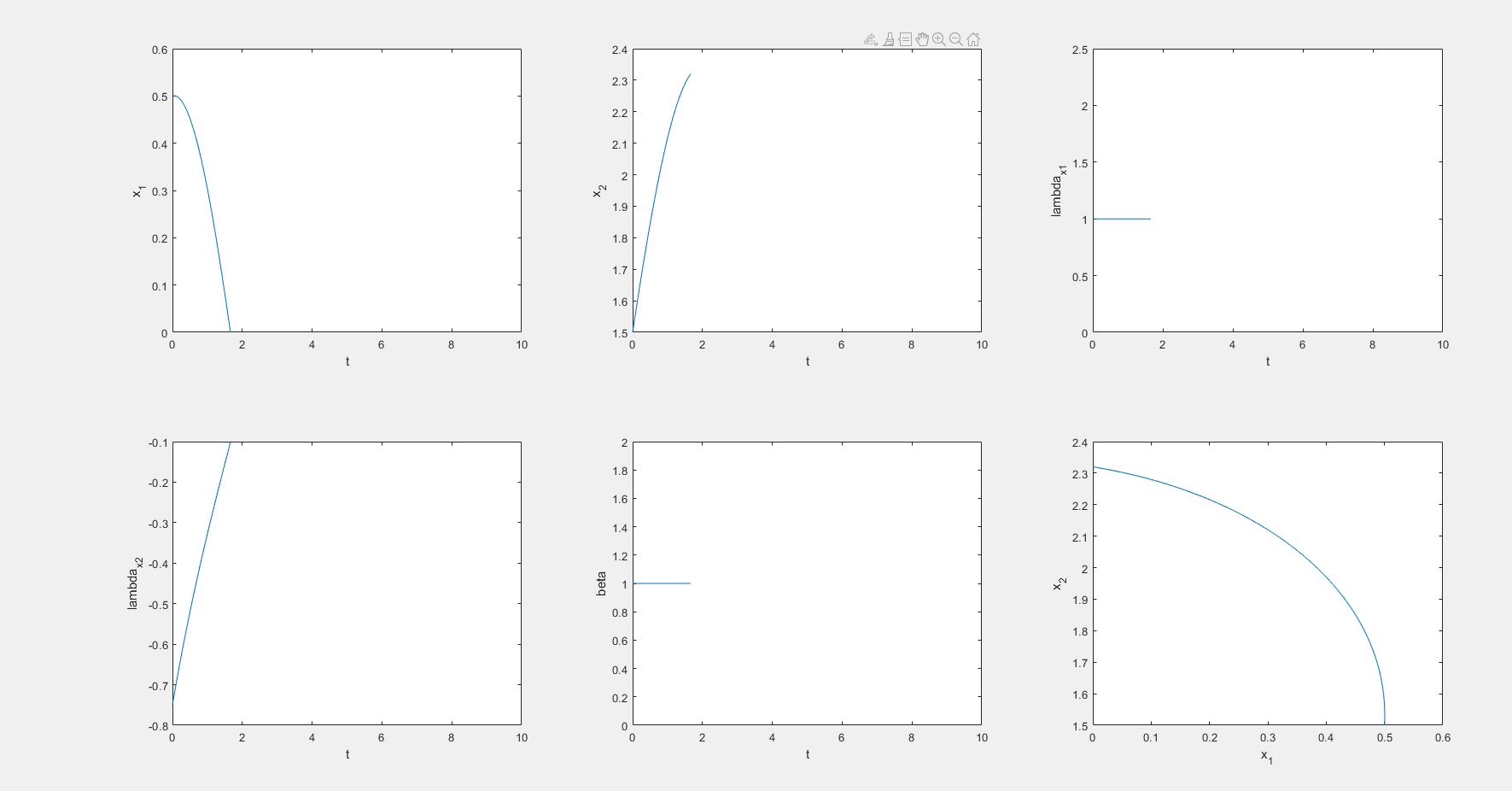
problem.states.x0=[0.5 1.5 1 -0.75];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];

boundaryCost = -tf;



要想获得接近答案的曲线，对lambda\_2要有一个合理的初值猜想，此处-0.75比较合适

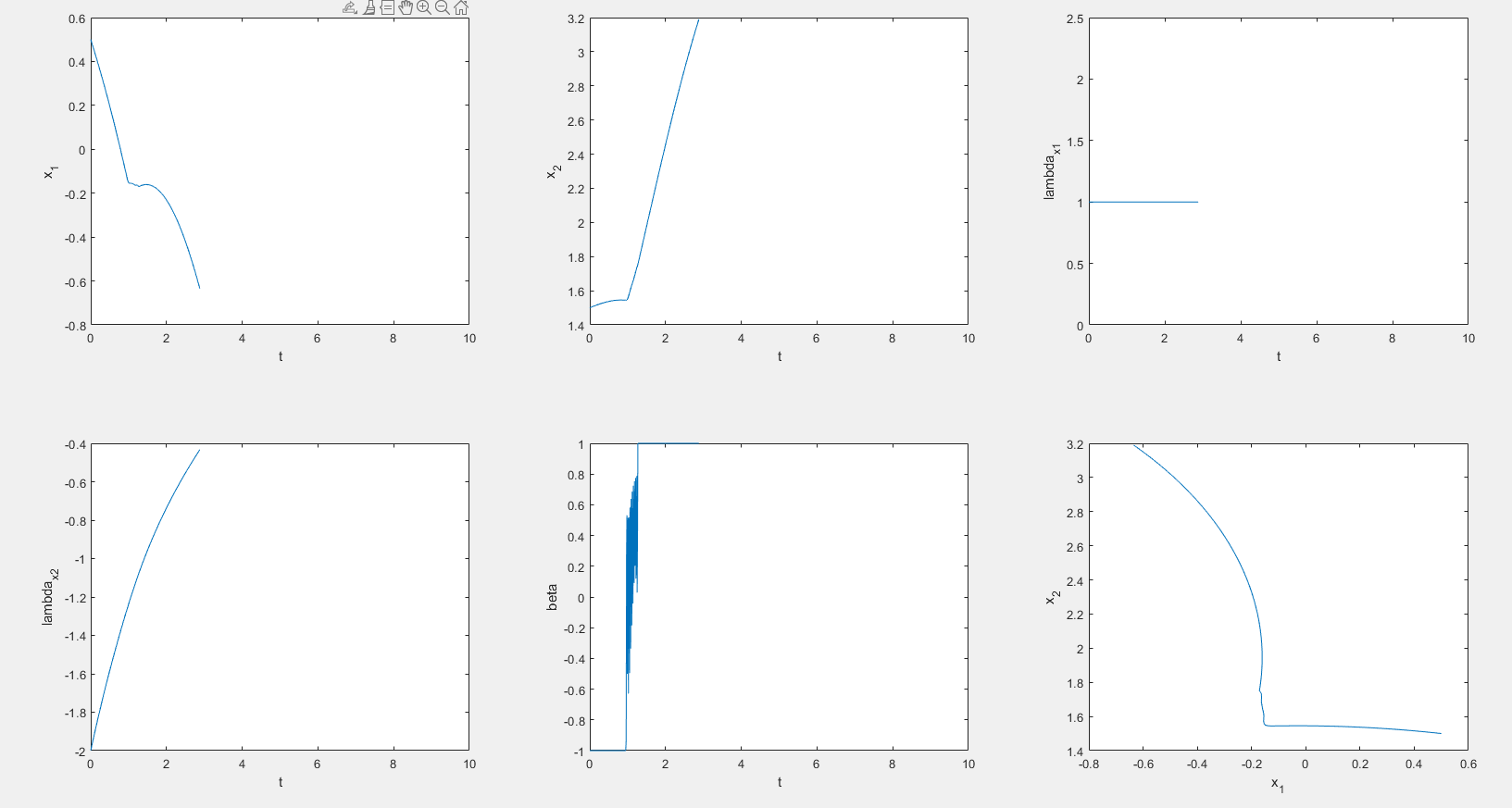
% Initial conditions for system

problem.states.x0=[0.5 1.5 1 -2];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];



Totally wrong!!!

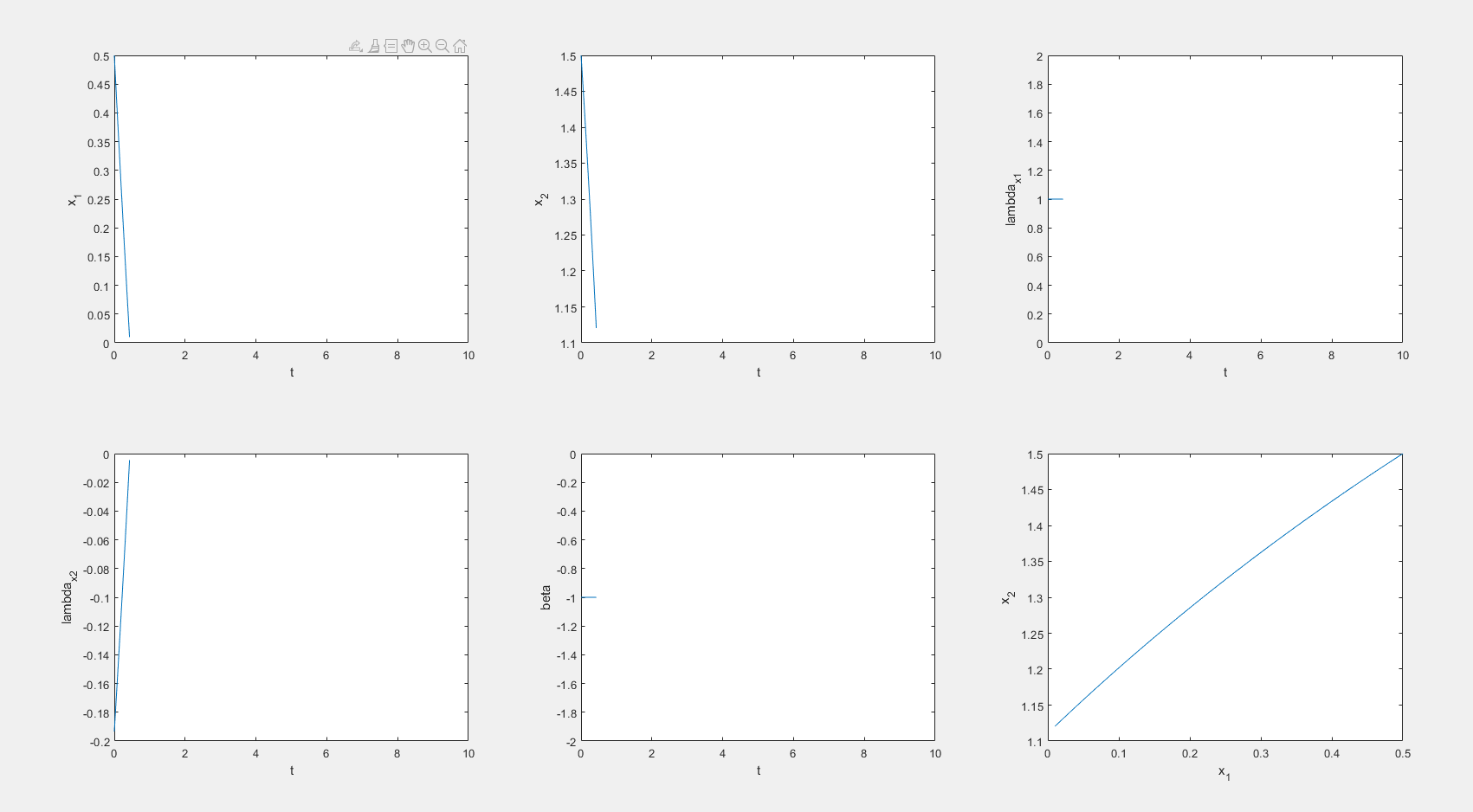
% Initial conditions for system

problem.states.x0=[0.5 1.5 1 1];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[0.5 1.5 -10 -10];

problem.states.x0u=[0.5 1.5 10 10];



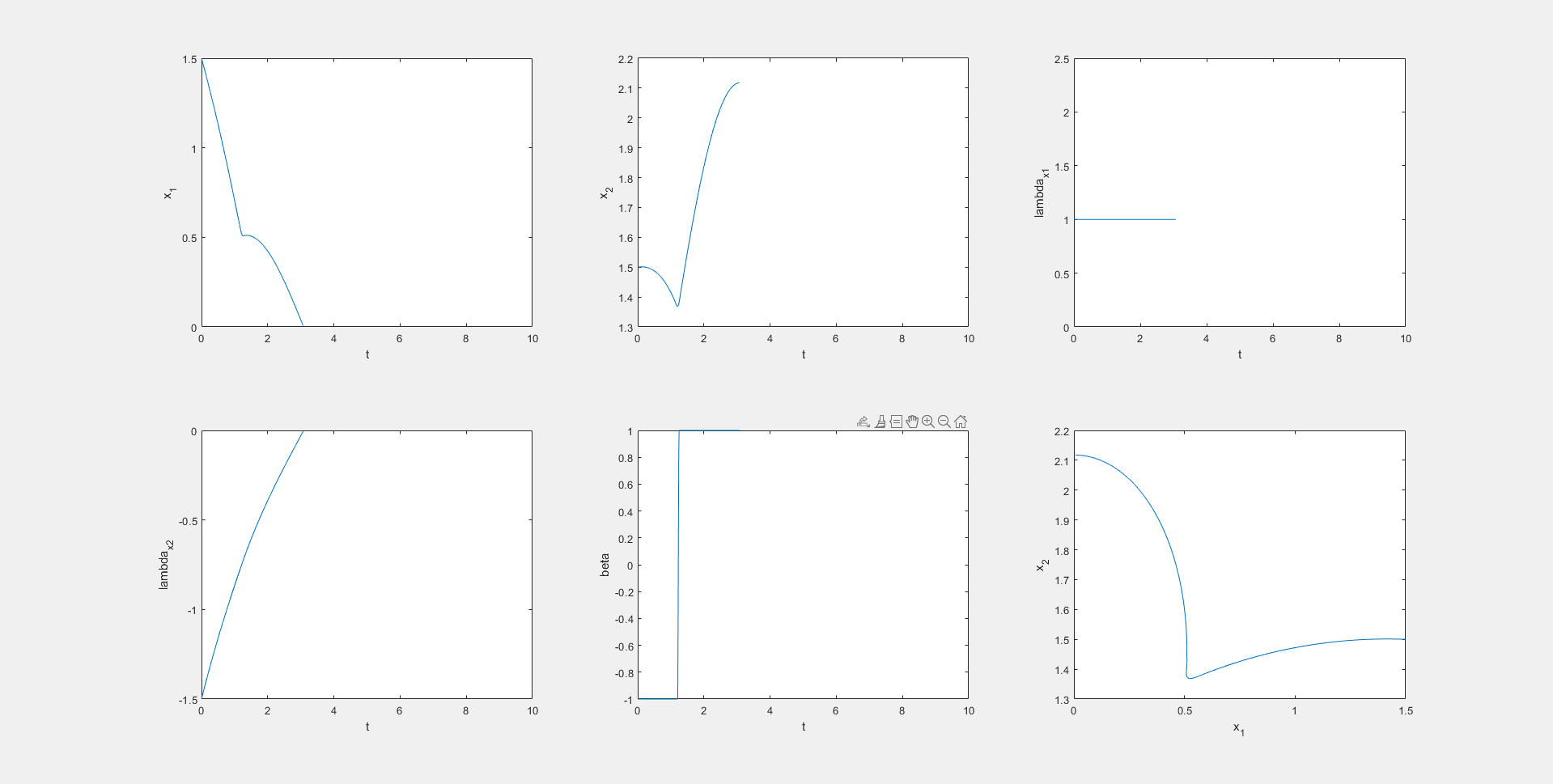
% Initial conditions for system

problem.states.x0=[1.5 1.5 1 -1.5];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];



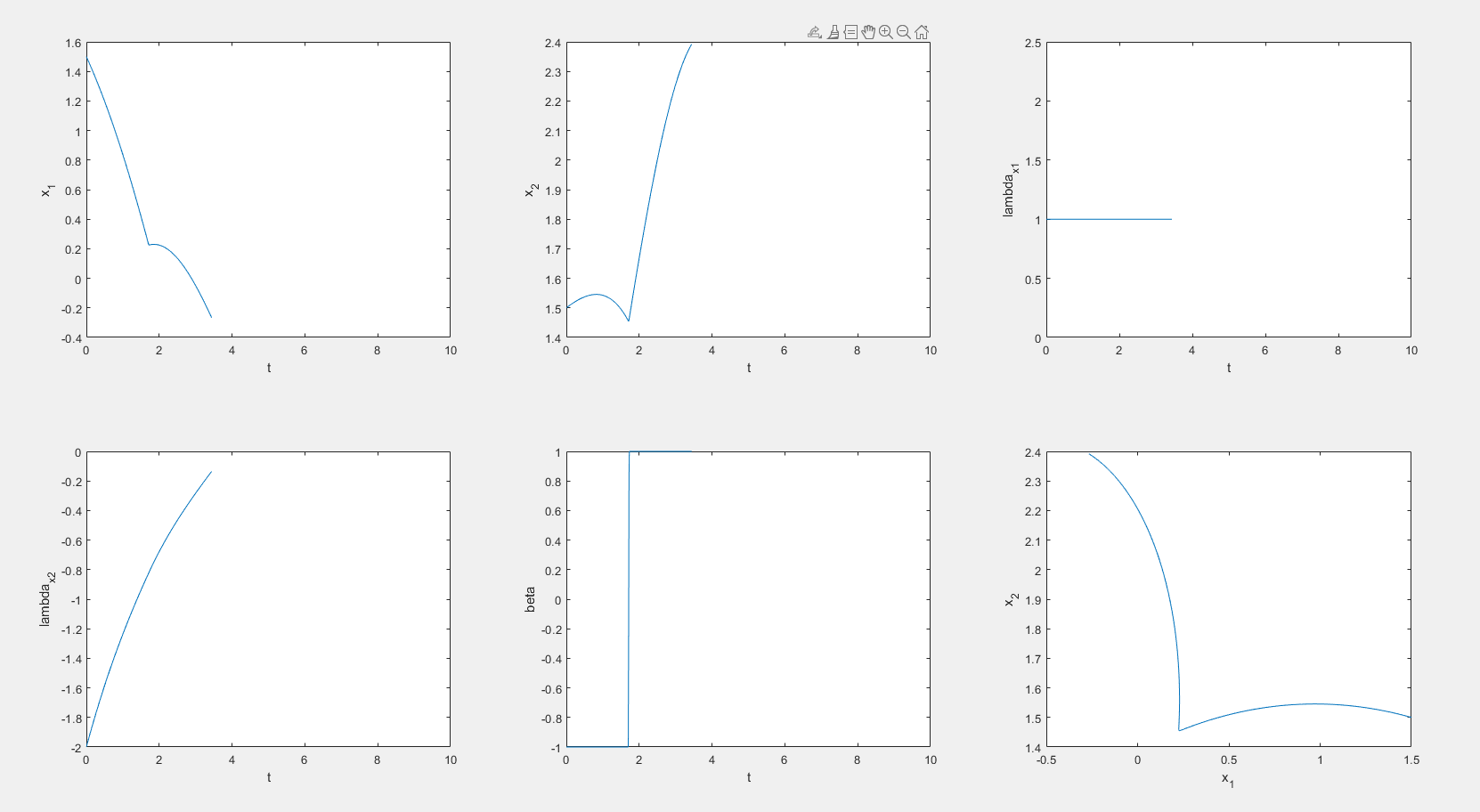
% Initial conditions for system

problem.states.x0=[1.5 1.5 1 -2];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];



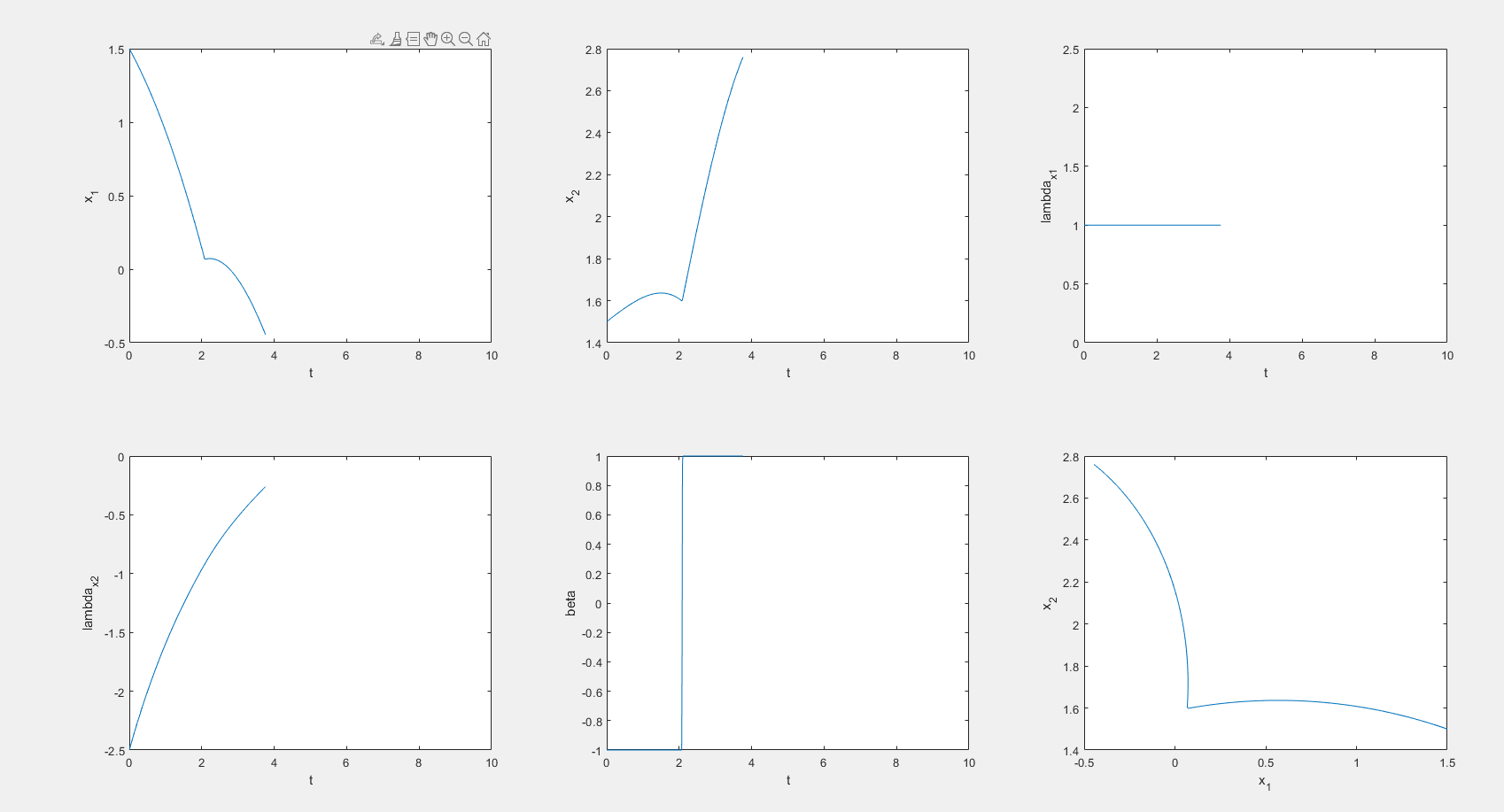
% Initial conditions for system

problem.states.x0=[1.5 1.5 1 -2.5];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];



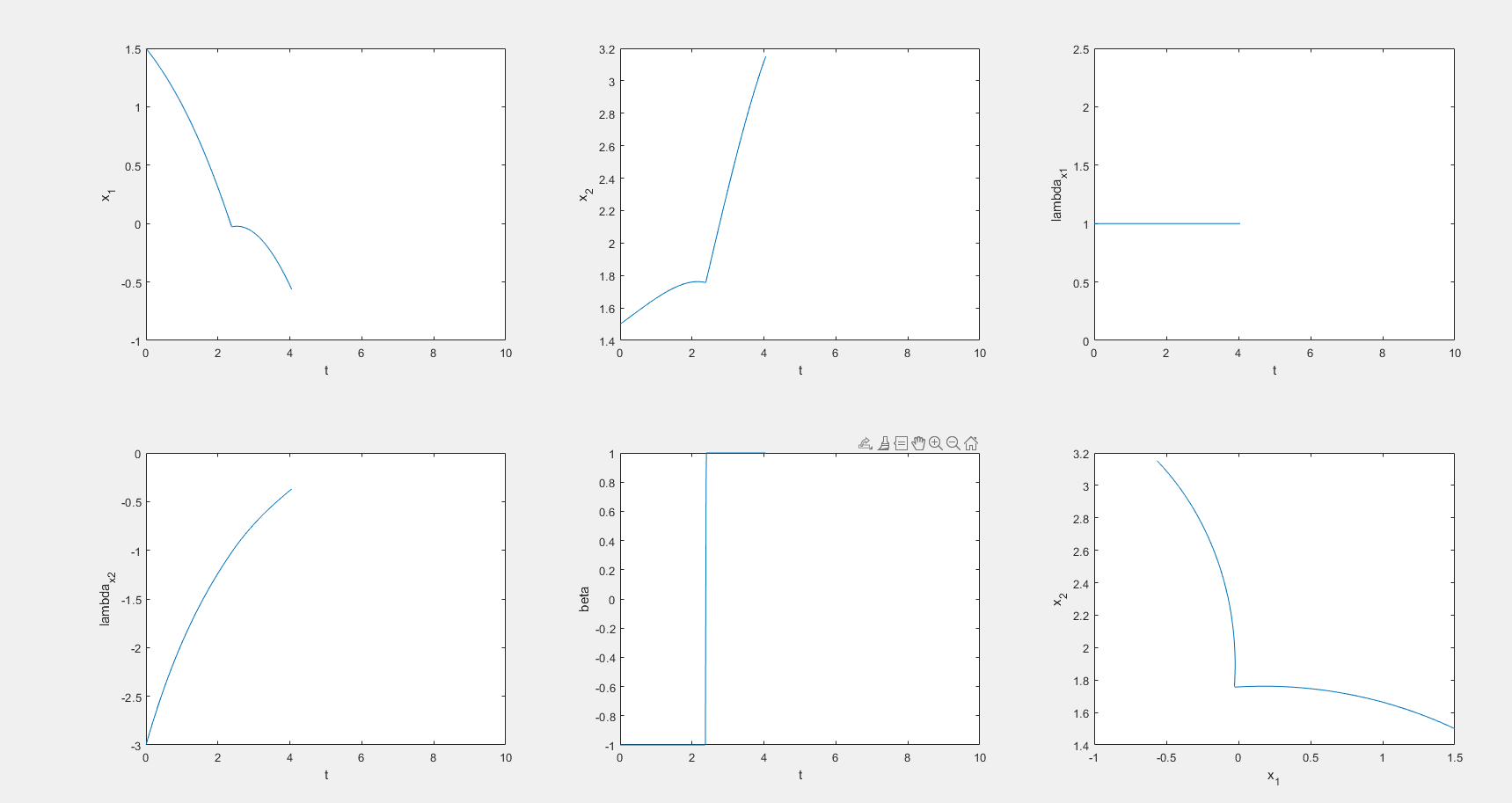
% Initial conditions for system

problem.states.x0=[1.5 1.5 1 -3];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];



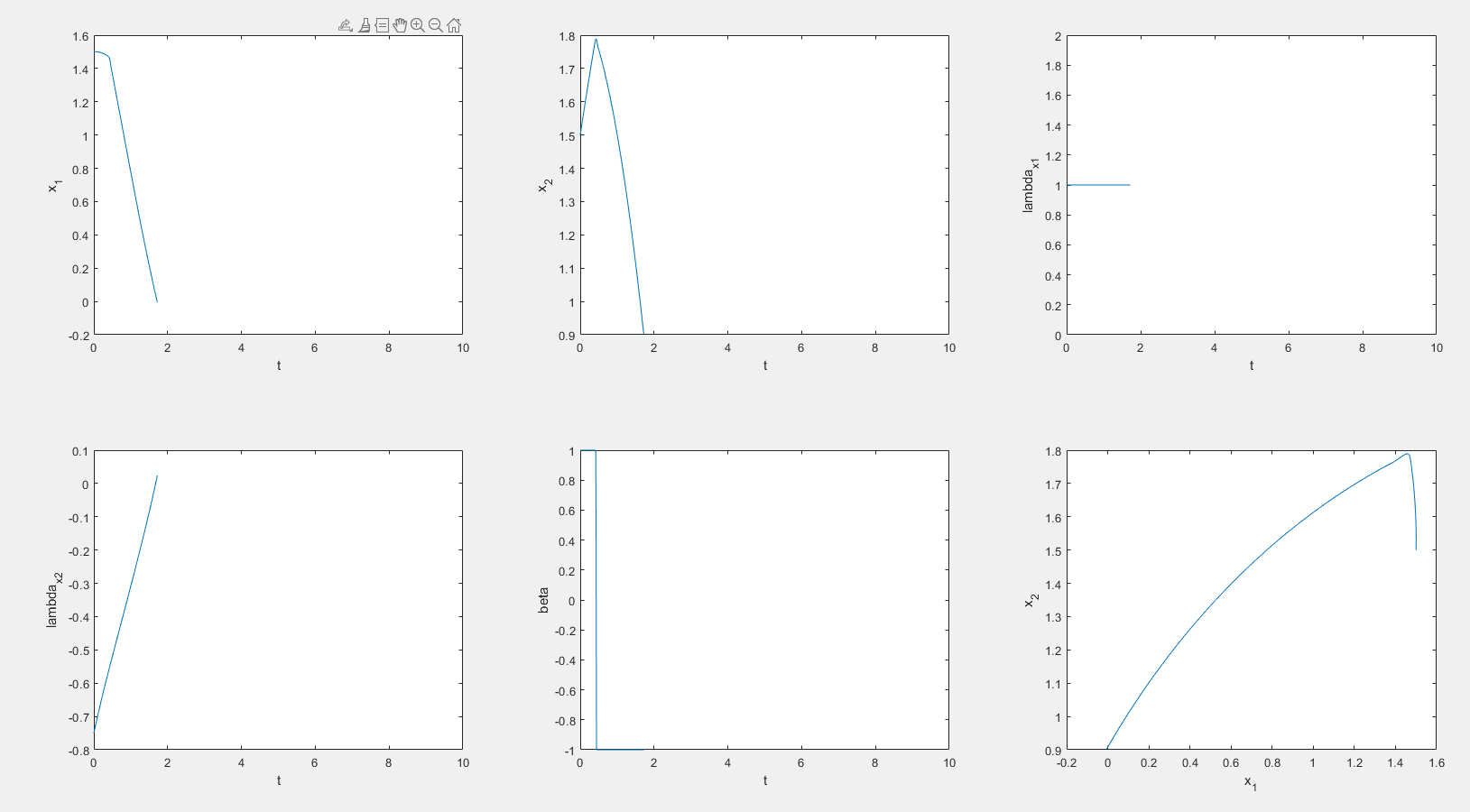
% Initial conditions for system

problem.states.x0=[1.5 1.5 1 -0.75];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[1.5 1.5 -10 -10];

problem.states.x0u=[1.5 1.5 10 10];



Lambda\_x1本身是一个常数，给定初始条件为多少就会是多少，问题中最为重要的是lamda\_x2/lambda\_x1的大小

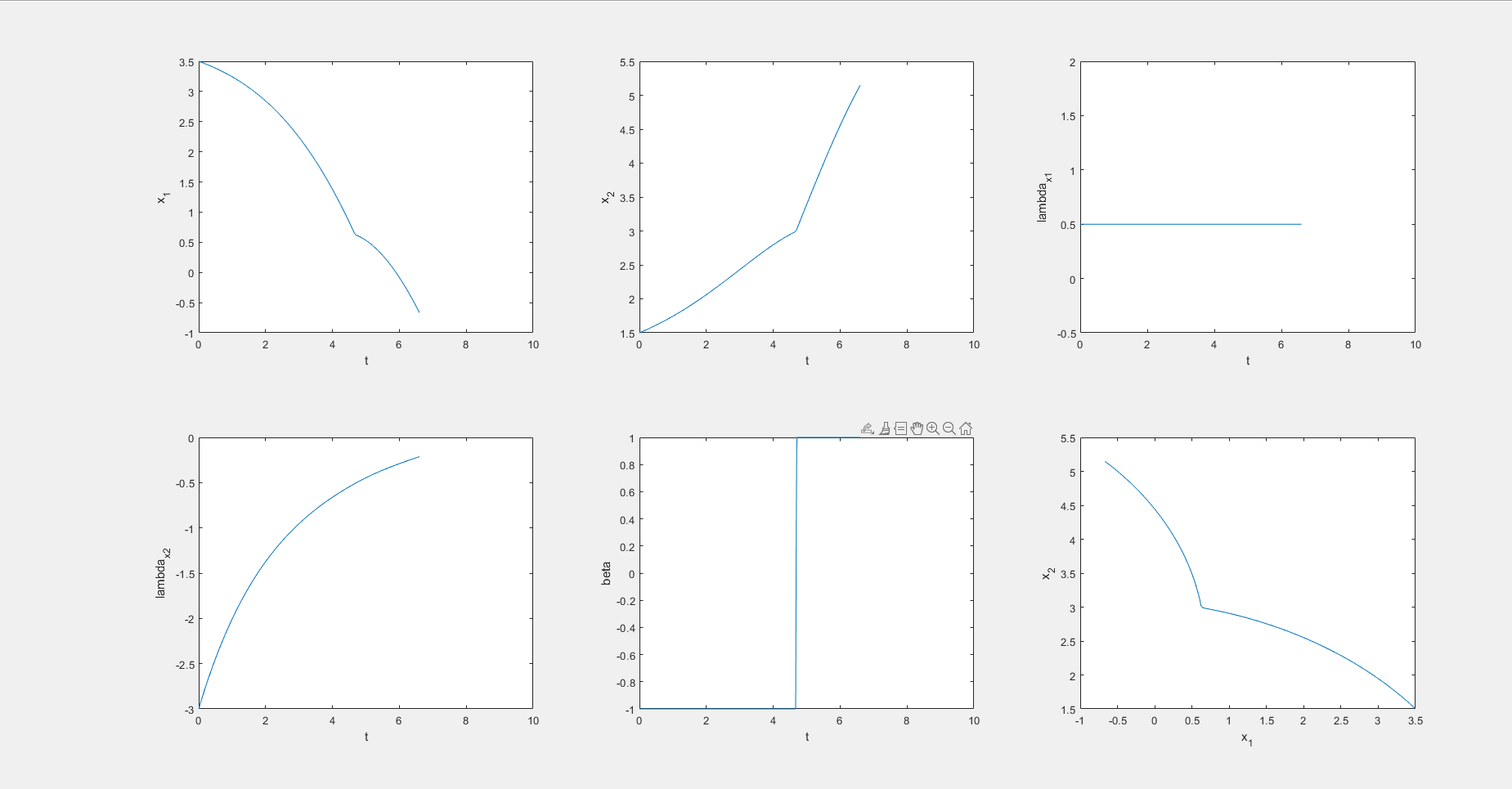
% Initial conditions for system

problem.states.x0=[3.5 1.5 0.5 -3];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[3.5 1.5 -10 -10];

problem.states.x0u=[3.5 1.5 10 10];



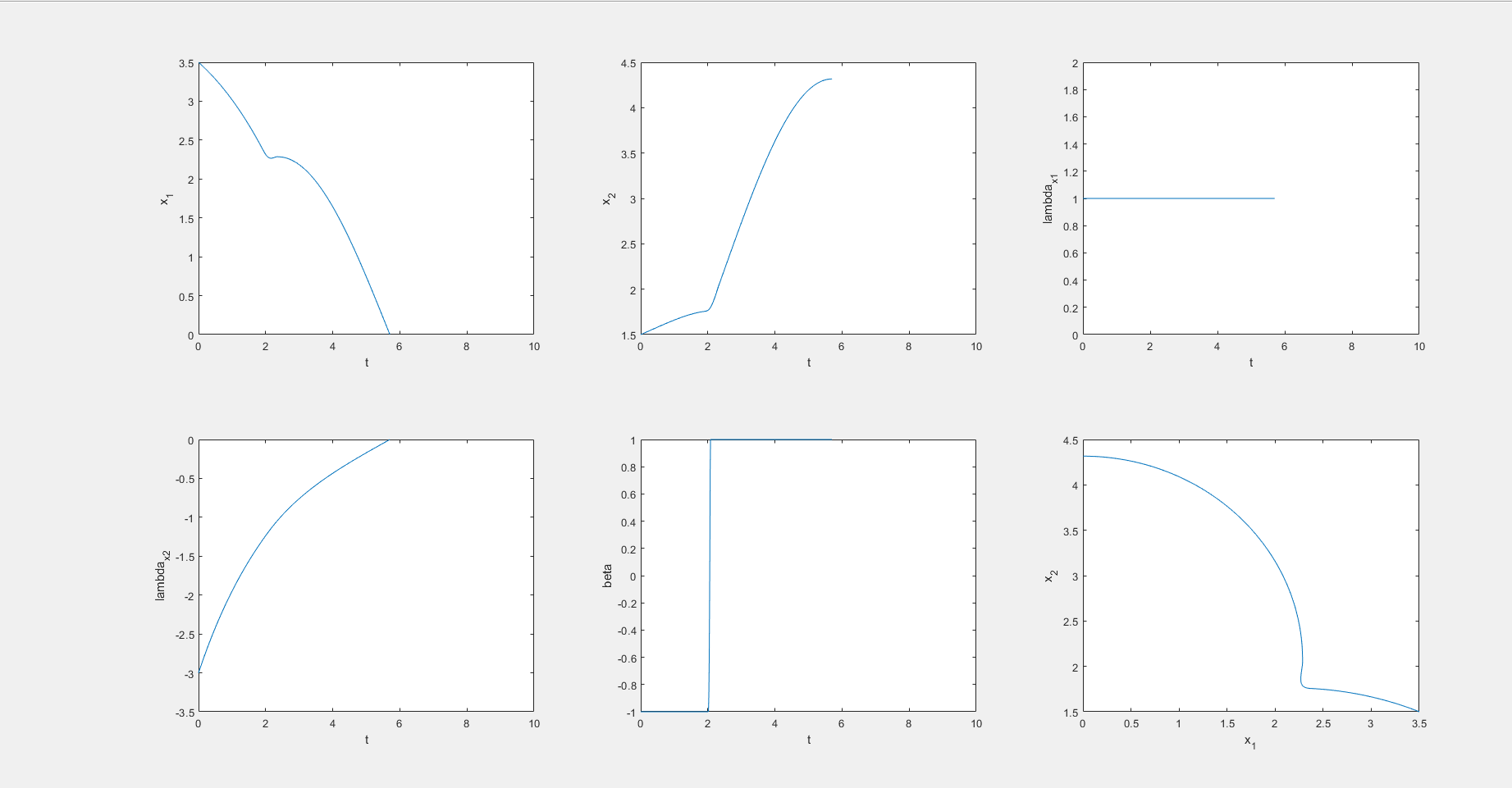
% Initial conditions for system

problem.states.x0=[3.5 1.5 1 -3];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[3.5 1.5 -10 -10];

problem.states.x0u=[3.5 1.5 10 10];



会不会问题有多个解，在不同的初始条件范围内最优解不一样

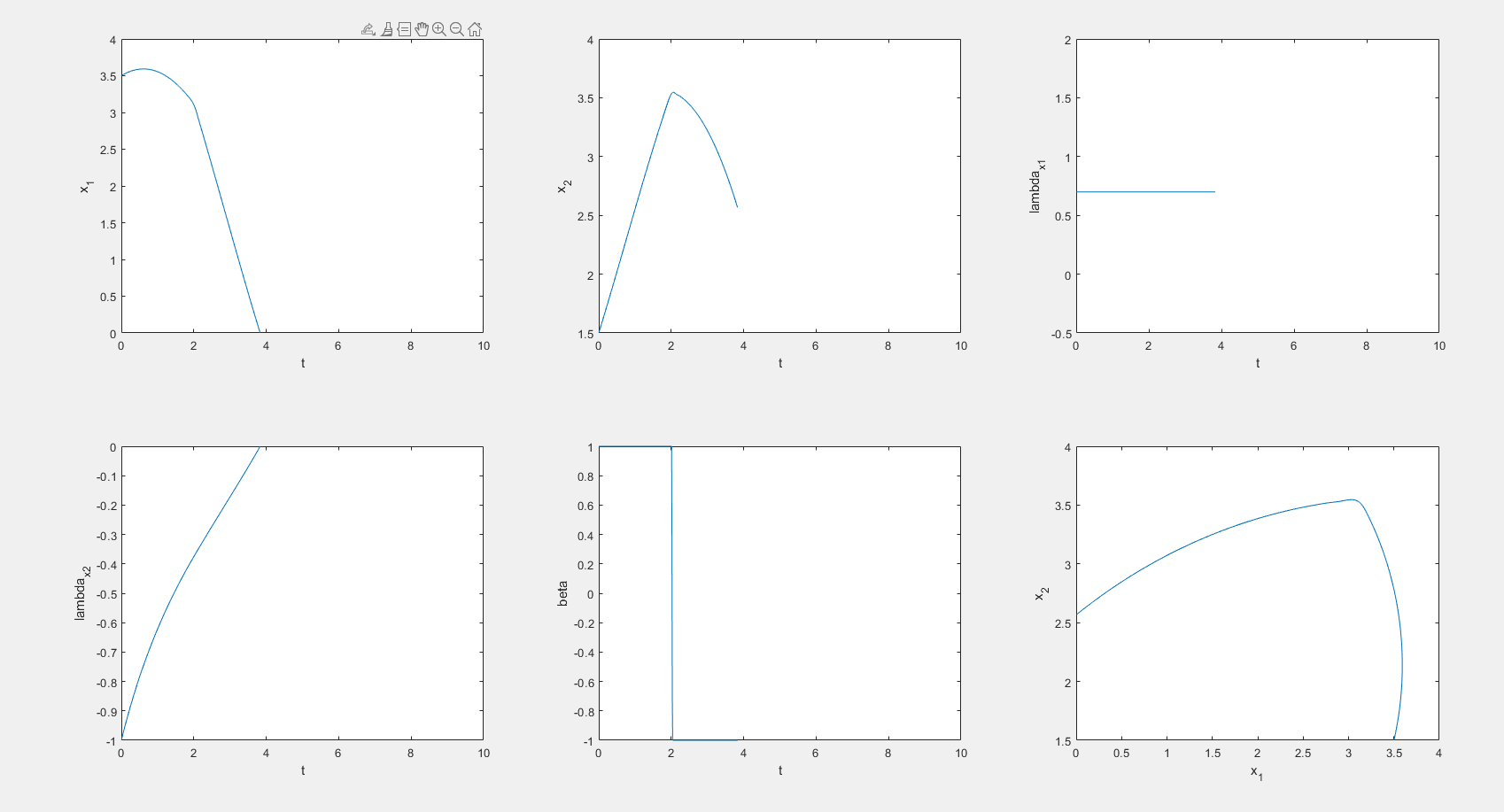
% Initial conditions for system

problem.states.x0=[3.5 1.5 1 -1];

% Initial conditions for system. Bounds if x0 is free s.t. x0l=< x0 <=x0u

problem.states.x0l=[3.5 1.5 -10 -10];

problem.states.x0u=[3.5 1.5 10 10];



只有协态变量选择正确才有正确的结果。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Initial data | Saved files | Meshes |  |
|  | 1.5 1.5 1 -2 | 1.mat | 100 |  |
|  | 1.5 1 1 -2 | 2.mat | 100 |  |
|  | 0.5 1.5 1 -1 | 3.mat | 100 |  |
|  | [0.4 1.5 1 -0.75] | 4.mat | 100 |  |
|  | [1 1.5 1 -1.25] | 5.mat | 100 |  |
|  | x\_0 = 1.25;  y\_0 = 1.5;  lambda\_0 = -1.5; | 6.mat | 100 |  |
|  | x\_0 = 2;  y\_0 = 1.5;  lambda\_0 = -2; | 7.mat | 100 |  |
|  | x\_0 = 2.5;  y\_0 = 1.5;  lambda\_0 = -2.5; | 8.mat | 100 |  |
|  | x\_0 = 3;  y\_0 = 1.5;  lambda\_0 = -3; | 9.mat | 100 |  |
|  | x\_0 = 4;  y\_0 = 1.5;  lambda\_0 = -4; | 10.mat | 100 |  |

tv 时间点

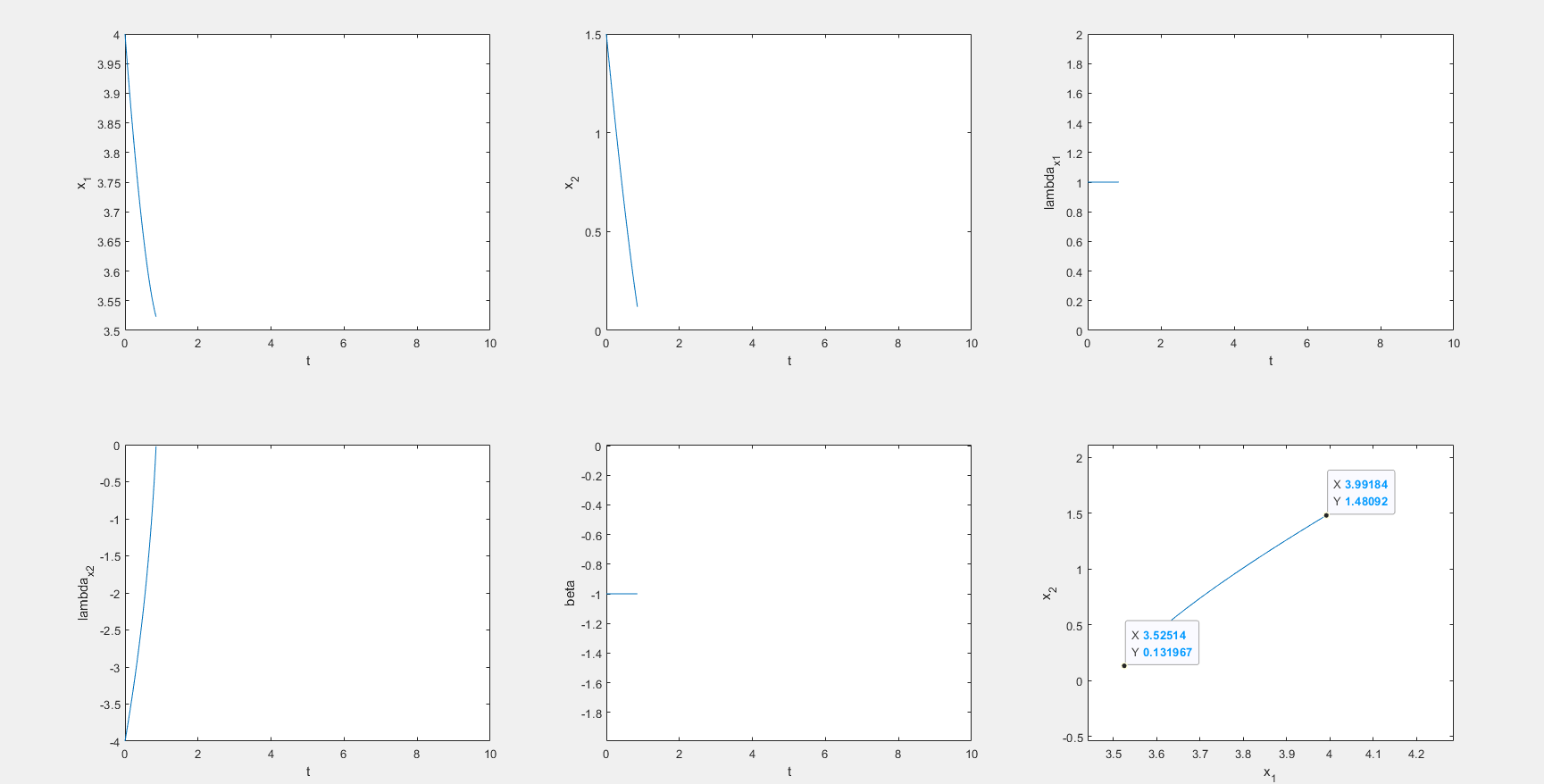
uv beta control variable of one player who want to maximize the completion time

xv(1:2,:) state variables

xv(3:4，：) costate variables

alpha = atan(-lambda\_x2./-lambda\_x1);

使用神经网络做出来的结果不正确。



for lambda\_x1 = -5:0.1:5

for lambda\_x2 = -5:0.1:5

alpha = atan(lambda\_x2./lambda\_x1)+pi;

row = [lambda\_x1 lambda\_x2 alpha];

data = [data;row];

end

end

当步长为0.1时，生成数据训练神经网络，测试结果为

>> load('NN\_data\_generator')

>> net([1;1])

ans =

3.7986

>> atan(1)+pi

ans =

3.9270

产生了比较大的误差。

for lambda\_x1 = -5:0.05:5

for lambda\_x2 = -5:0.05:5

alpha = atan(lambda\_x2./lambda\_x1)+pi;

row = [lambda\_x1 lambda\_x2 alpha];

data = [data;row];

end

end

> net([1;1])

ans =

3.8592

>> pi + atan(1)

ans =

3.9270

相对而言结果的误差变小了些许。

>> net([1;1])

ans =

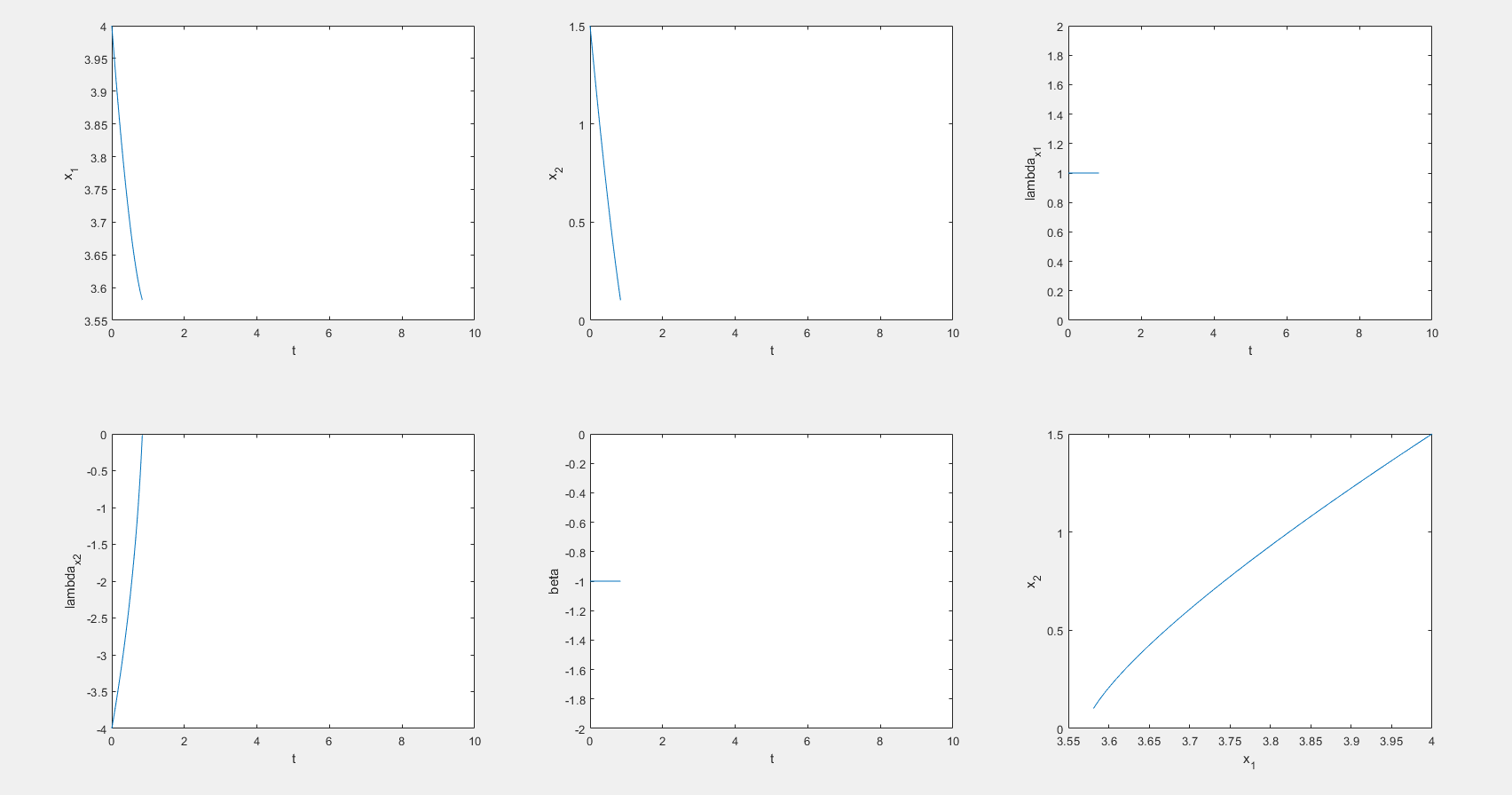
3.8294

>> pi + atan(1)

ans =

3.9270

第二次训练结果竟然不一样了。神经网络的梯度下降训练方法，本身就具有随机性，不一定就是每次都一样。



>> load('NN\_data\_generator')

>> net([100;1])

ans =

1.6387

>> atan(1/100)+pi

ans =

3.1516

这两个就是错误的。

1/100 = 0.05/5 训练数据就在其中才对。

>> net([5;0.05])

ans =

3.0917

从公式来看，数据相等，但是从神经网络角度来说，数据不相等。因为其映射的规则不是确定的公式。

for lambda\_x1 = -5:0.02:5

for lambda\_x2 = -5:0.02:5

alpha = atan(lambda\_x2./lambda\_x1)+pi;

row = [lambda\_x1 lambda\_x2 alpha];

data = [data;row];

end

end

生成这个数据花费的时间很长，5分钟左右。

>> net([1;1])

ans =

3.8163

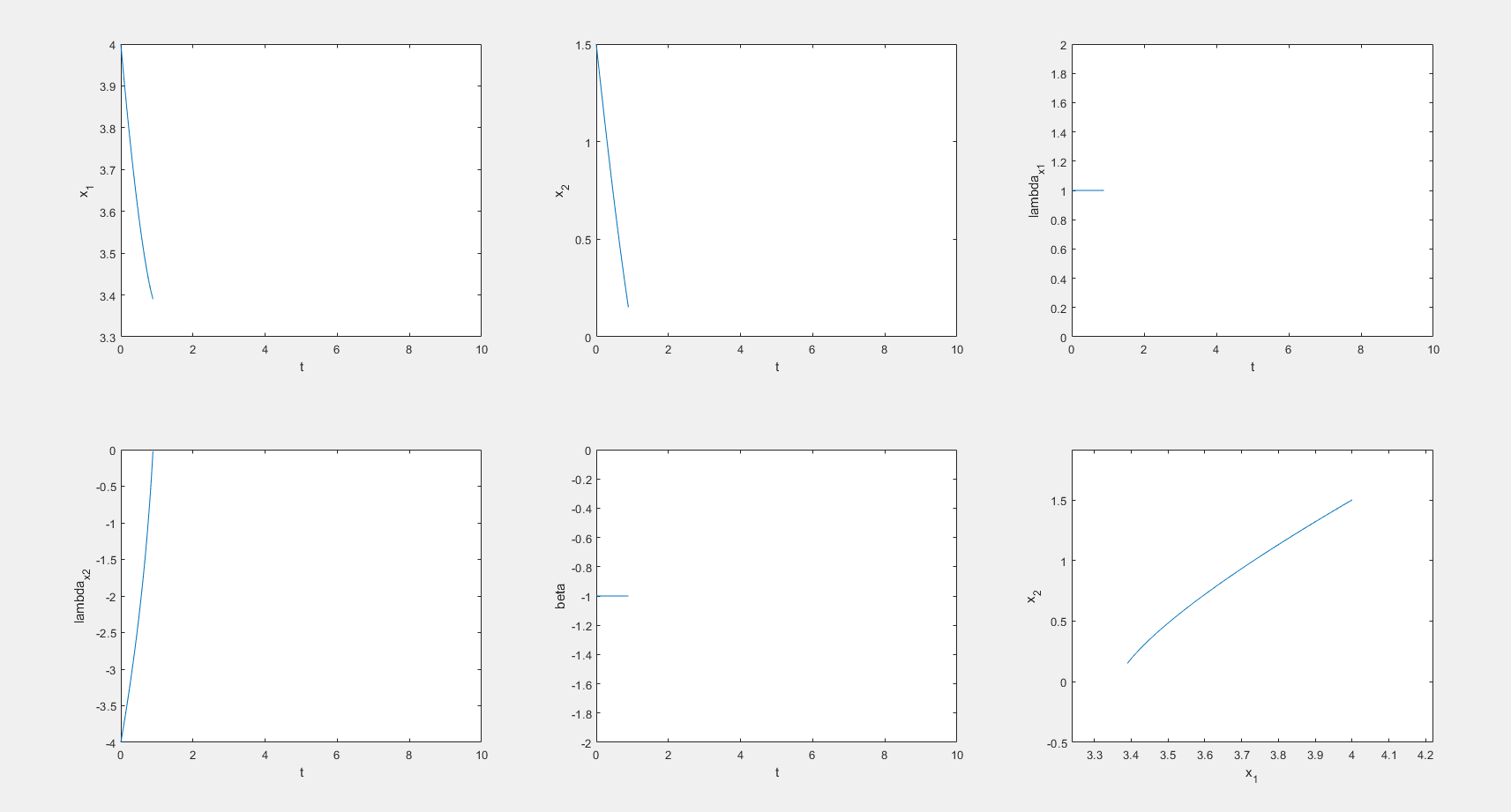
>> pi + atan(1)

ans =

3.9270

相较于上一个步长，误差反而变大了。

计算快慢路径问题也很慢。7分钟左右



看最终结果，好像没有太大区别。

无论甚至怎样的步长，使用神经网络的结果都差不多，神经网络的超参数怎么设置呢？

几个结果都差不多，是哪里出了错误

检查出来一个错误，在

load('NN\_data\_generator');

后，NN\_data\_generator 中包含变量lambda\_x1 lambda\_x1 alpha

使得原本在Dynamics\_Internal中的向量变成了标量。

load('NN\_data\_generator');

alpha = zeros(length(lambda\_x1),1);

for i = 1:1:length(lambda\_x1)

alpha(i) = net([lambda\_x1(i);lambda\_x2(i)]);

end

sinalph = sin(alpha);

cosalph = cos(alpha);

使用for循环程序会很慢，最优使用向量表达式的形式。

load('NN\_data\_generator');

alpha = net([lambda\_x1';lambda\_x2'])';

sinalph = sin(alpha);

cosalph = cos(alpha);

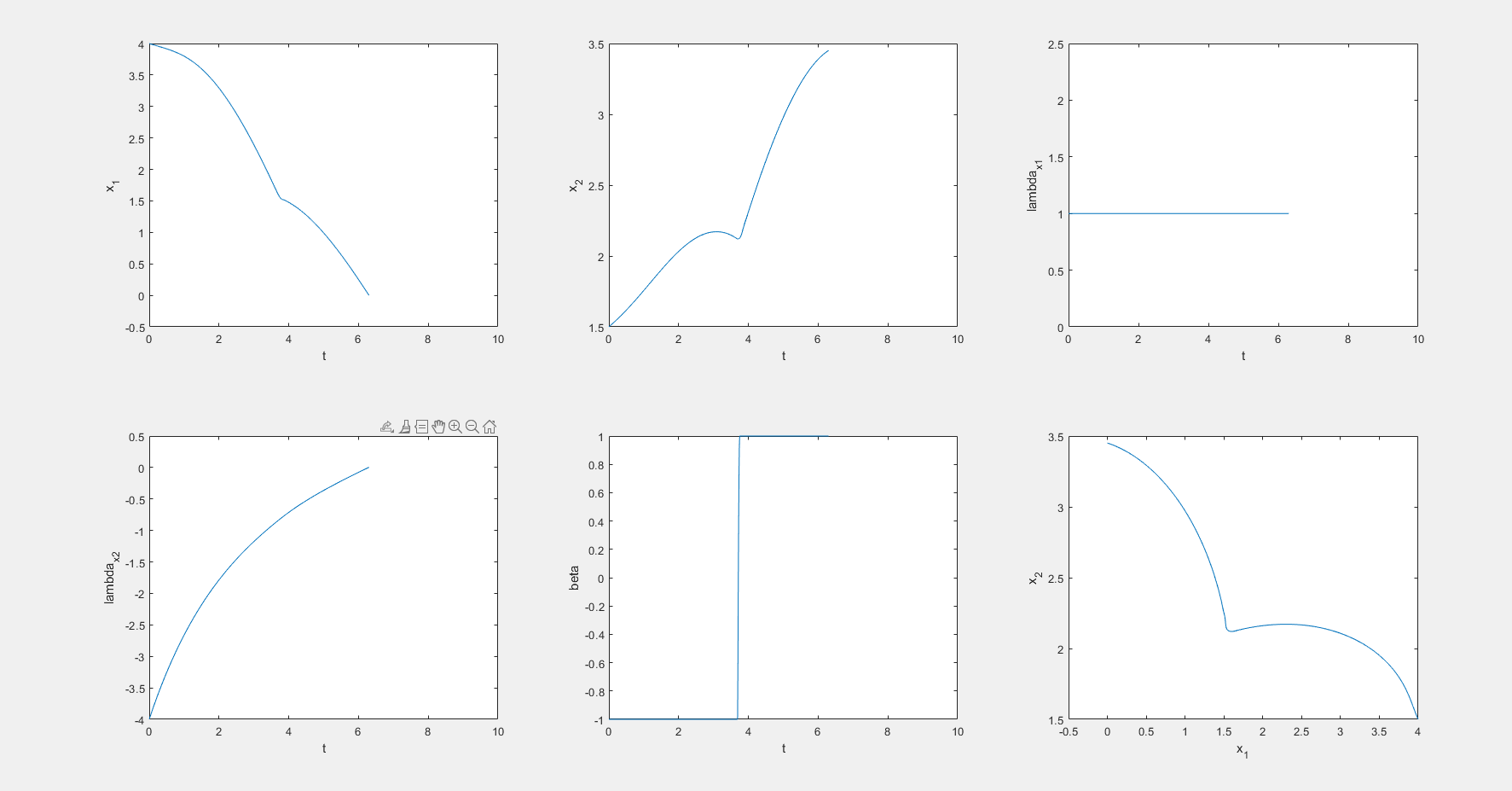
成了。

神经网络数据生成范围

lambda\_x1 = [-5,5] lambda\_x2 = [-5,5]

step = 0.05

delete all other variable except the net object.



可能其他步长以及网络规模也会成功。