The optimization of control parameters K_v and K_p

Based on Computed Torque

By Miao Hong

HITSZ

Computed Torque

We are given a description of the dynamics of a robot manipulator in the form of the equation

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta,\dot{\theta}) = \tau \qquad (1)$$

Consider the following control law:

$$\tau = \mathrm{M}(\theta)(\ddot{\theta}_d - K_v\dot{e} - K_p e) + C(\theta,\dot{\theta})\dot{\theta} + N(\theta,\dot{\theta}) \qquad (2)$$

Where $e = \theta - \theta_d$ and K_v and K_p are constant gain matrices

When substituted into equation (1), the error dynamics can be written as:

$$\mathrm{M}(\theta)(\ddot{e} + K_v\dot{e} + K_p e) = 0$$

Since $\mathrm{M}(\theta)$ is always positive definite, we have

$$\ddot{e} + K_v\dot{e} + K_p e = 0$$

To be continued

Consider the following differential equation group:

$$\ddot{e} + K_v \dot{e} + K_p e = 0$$

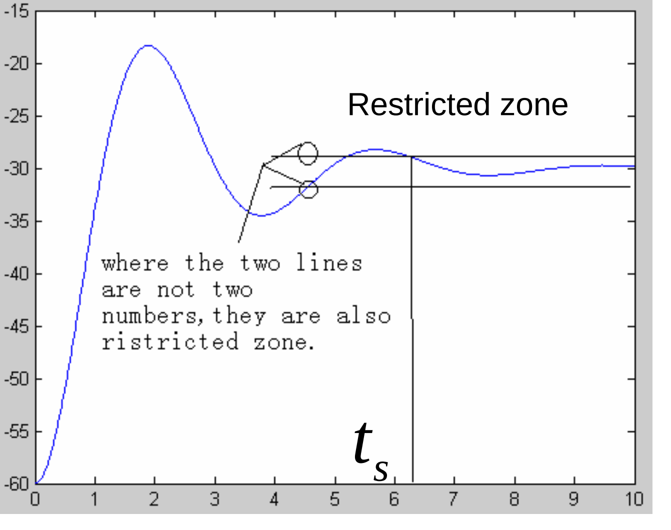
If we can solve the vector e ,then we can get the settle time of system t_s

Analytic Solution and Numerical solution :

1. Analytic Solution: using matlab function “dsolve()”

2. Numerical Solution: Using ODE45 integration method

Given a restricted zone , then search the value like t_s



Analytic Solution based on diag

$$k_p = \omega_n^2$$

$$kv = 2\xi\omega_n$$

$$\Delta = 0.05$$

$$t_s = \frac{3 + \ln\frac{1}{\sqrt{1-\xi^2}}}{\xi\omega_n} = \frac{3 + \ln\frac{1}{\sqrt{1-\frac{kv^2}{4kp}}}}{0.5kv}$$

brought forward, continued

$$\ddot{e} + K_v \dot{e} + K_p e = 0$$

So consider the step response

$$\begin{pmatrix} \ddot{e}_1 \\ \ddot{e}_2 \\ \ddot{e}_3 \end{pmatrix} + \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix} \begin{pmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{pmatrix} + \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = 0$$

That the K_v and K_p are constant gain matrices, both of them are 3-by-3 matrices.

Discussion on K_v and K_p

Since our equation is linear ,it is easy to choose K_v and K_p so that the overall system id stable and $e \to 0$ exponentially as $t \to \infty$ .Moreover ,we can choose K_v and K_p such that we get independent exponentially stable systems(by choosing K_v and K_p diagonal)

At first ,to simplify the optimization procedure, consider K_v and K_p as diagonal matrix:

$$K_v = \begin{pmatrix} k_{v11} & 0 & 0 \\ 0 & k_{v22} & 0 \\ 0 & 0 & k_{v33} \end{pmatrix} \qquad K_p = \begin{pmatrix} k_{p11} & 0 & 0 \\ 0 & k_{p22} & 0 \\ 0 & 0 & k_{p33} \end{pmatrix}$$

For the three arms are coupling, that the general condition is following form:

$$K_v = \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix} \qquad K_p = \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix}$$

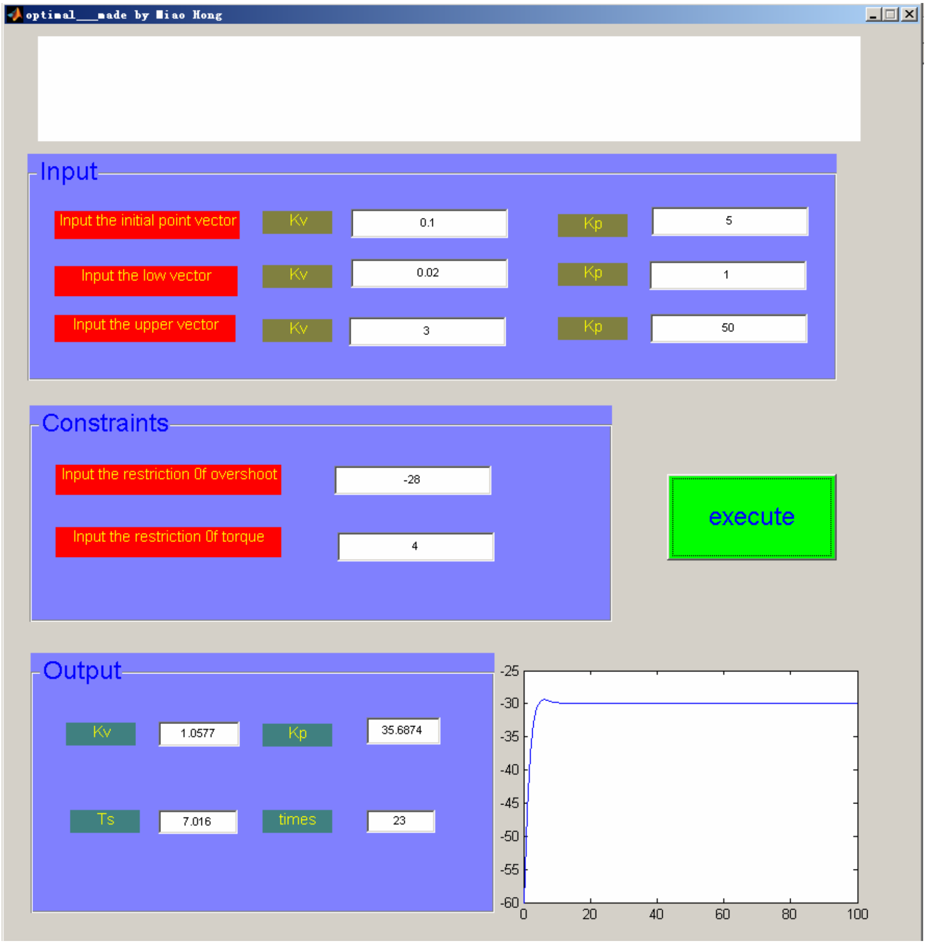
Attempt 1

Using a random algorithm (CRS) and K_v K_p are diagonal matrix

Minimize the settle time t_s

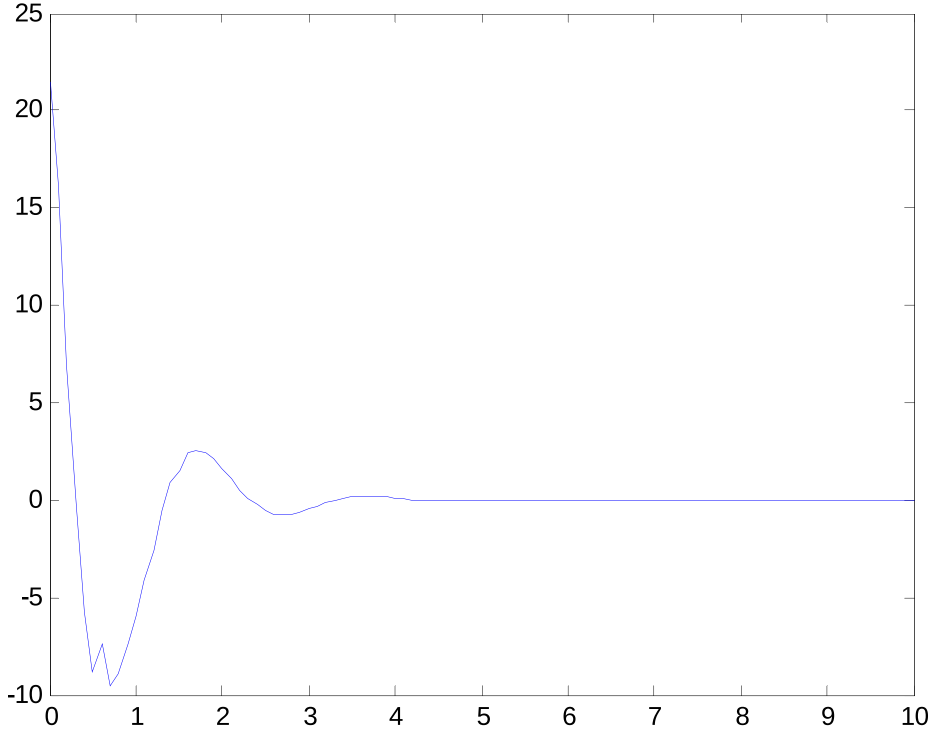
Given the constraint condition: Overshoot and Torque

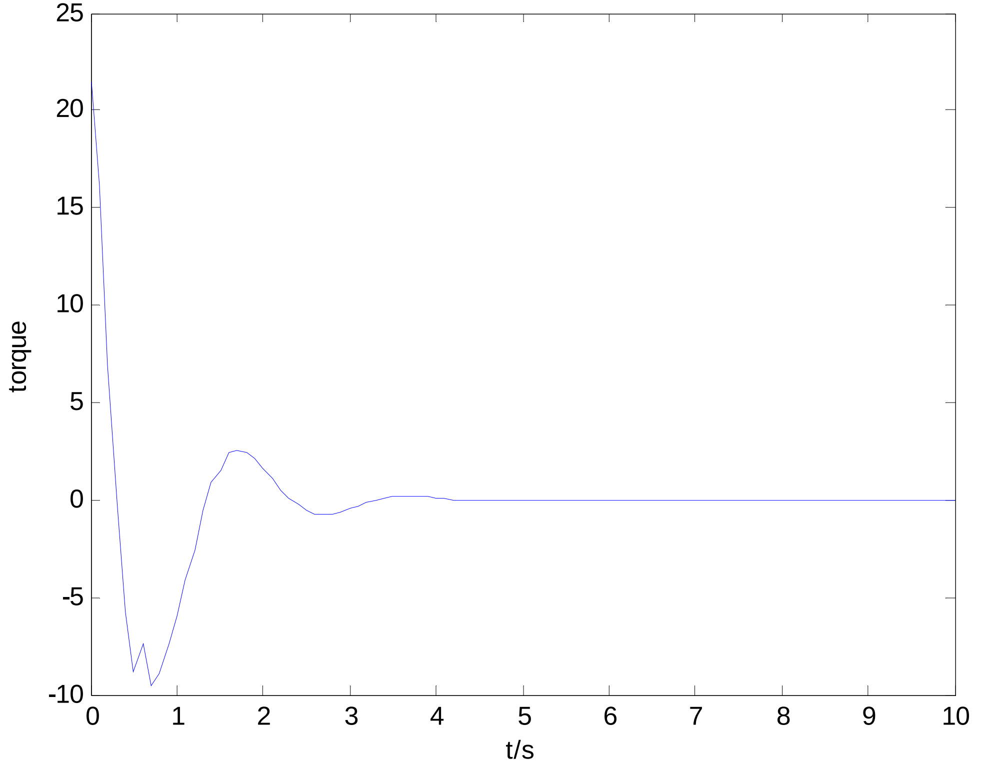
Using CRS algorithm (a random algorithm)



Ts:5%, torque=8

Item / serial number	K_v	K_p	t_s	zone
1	2.5422	27.6986	2.6270	(0.01 10) (1 30)
2	2.7137	27.6232	2.5800	(0.01 20) (10 40)
3	2.8040	40.2054	2.2080	(0.1 30) (10 100)
4	3.1482	49.0453	1.9950	(1 20) (30 100)
5	3.9381	83.5124	1.5360	(1 40) (50 200)


25
20
15
10
5
0
-5
-10
0
1
2
3
4
5
6
7
8
9
10


25
20
15
10
5
0
-5
-10
torque
0
1
2
3
4
5
6
7
8
9
10
t/s

Attempt 2

K_v K_p are full matrix

Using DE (Differential Evolution) algorithm

Given the constraint condition: Overshoot and Torque

	K_v K_p	t_s
1	[4.26706597 14.92405344 4.05558081 11.07597856 1.37044929 10.98976583 10.81332597 9.77559251 7.14290767 8.34374800 2.16020216 19.02660746 3.76842172 4.84391167 3.75787337 1.82821512 9.89138987 6.65297016]	26.70000000
2	[8.38415344 9.66491588 2.82934740 8.48853407 4.03860291 0.46114840 8.46650245 29.30018339 9.91270916 9.75393670 6.20485862 14.71557669 4.86861018 8.46452498 1.19715496 2.47733172 9.85699738 9.88032570]	1.90000000
3	[1.72673409 1.44809618 1.19204083 1.71647225 1.91543135 1.20000000 1.63146158 1.10856461 1.12551764 1.59470321 1.28321770 1.26734351 2.52512542 1.55333279 1.09185410 1.51563279 1.20000000 1.86734762]	1.40000000
4	[11.06984902 18.72165141 1.62312409 9.31282423 12.64859416 6.83273190 6.71091474 5.91628608 7.81480318 2.74264273 9.97740450 1.92199301 3.13046826 17.45543587 3.20369254 15.61417119 4.85298320 14.62482400]	0.40000000
5	[20.65988912 17.19640978 4.69860201 11.84740403 22.05042745 8.26414227 16.63362331 5.15625052 0.66985597 5.75128478 2.16823081 2.56862454 15.20386660 20.63099331 7.46930910 6.66251992 9.85351572 27.14483642]	0.30000000

Research on the distinction between the full matrix and diagonal matrix

If we choosing K_v and K_p as diagonal, that we can get independent system (three independent systems).

But the three arms are coupling, so the general situation is that K_v and K_p are not diagonal matrix.

To be continued

Consider a damped free oscillation, The dynamics of the system are given by the following equation

$$M\ddot{q} + B\dot{q} + Kq = 0$$

Where M,B, and K are all positive quantities. As a state space equation we rewrite equation as

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{q} \\ -(K/M)q - (B/M)\dot{q} \end{bmatrix}$$

Since this system is a linear system, we can determine stability by examining the poles of the system. The Jacobian matrix for the system is

$$A = \begin{pmatrix} 0 & 1 \\ -K/M & -B/M \end{pmatrix}$$

Which has a characteristic equation

$$\lambda^2 + (B/M)\lambda + (K/M) = 0$$

To be continued

The solutions of the characteristic equation are

$$\lambda = \frac{-B \pm \sqrt{B^2 - 4KM}}{2M}$$

Which always have negative real parts, and hence the system is globally exponentially stable.

From the pp248 (A Mathematical Introduction to Robotic Manipulation[1]),we known the stability is established on diagonal matrix .

So we can see that if we restrict the matrix to diagonal ,we may lose a lot of stability based on full matrix .

As we've seen , the full matrix is more general ,it covered every stability ,so it is more reasonable. Conversely , the diagonal matrix will lose a lot of reasonable results.

So we can predict that the result of full matrix is better than diagonal .

Note:[1] Richard M.Murray, Zexiang Li and S.Shankar Sastry. A Mathematical Introduction to Robotic Manipulation.2004

$$A = \begin{pmatrix} 0 & M \\ -\mathrm{M}^{-1} * Kp & -\mathrm{M}^{-1} * Kv \end{pmatrix}$$

$$\mathrm{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad Kp = \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix} \qquad Kv = \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -Kp11 & -Kp12 & -Kp13 & -Kv11 & -Kv12 & -Kv13 \\ -Kp21 & -Kp22 & -Kp23 & -Kv21 & -Kv22 & -Kv23 \\ -Kp31 & -Kp32 & -Kp33 & -Kv31 & -Kv32 & -Kv33 \end{pmatrix};$$

Eigenvalues[A]

We get the characteristic equation of A:

```
_Z^6+(kv33+kv22+kv11)*_Z^5+(-kv23*kv32+kp11+kp22-
kv21*kv12+kv11*kv22+kp33+kv11*kv33+kv22*kv33-
kv31*kv13)*_Z^4+(kp11*kv22-kv11*kv23*kv32-kv13*kp31-kv21*kp12-
kv31*kv13*kv22-kv12*kp21+kv21*kv13*kv32+kv11*kp33-
kv31*kp13+kp11*kv33+kv11*kv22*kv33-
kv21*kv12*kv33+kv11*kp22+kp22*kv33+kv31*kv12*kv23+kv22*kp33-kp23*kv32-
kv23*kp32)*_Z^3+(-kp11*kv23*kv32-kv11*kp23*kv32-kv21*kv12*kp33-
kp13*kp31+kv21*kv13*kp32+kv21*kp13*kv32+kp22*kp11+kv23*kv12*kp31+kv3
1*kv12*kp23+kv31*kp12*kv23-kv22*kv13*kp31+kp22*kp33-kv31*kv13*kp22-
kv31*kp13*kv22-kv12*kp21*kv33-kp12*kp21-
kv21*kp12*kv33+kv13*kp21*kv32+kv11*kv22*kp33+kp33*kp11-kp23*kp32-
kv11*kv23*kp32+kv11*kp22*kv33+kp11*kv22*kv33)*_Z^2+(-kv22*kp13*kp31-
kv12*kp21*kp33-kp11*kv23*kp32-
kv21*kp12*kp33+kv13*kp21*kp32+kv23*kp12*kp31+kp23*kv12*kp31+kv21*kp13
*kp32+kp11*kp22*kv33-kp22*kv13*kp31+kp13*kp21*kv32+kv31*kp12*kp23-
kp11*kp23*kv32-kp12*kp21*kv33-kv11*kp23*kp32-
kv31*kp13*kp22+kp11*kv22*kp33+kv11*kp22*kp33)*_Z+kp23*kp12*kp31+kp11*
kp22*kp33-kp22*kp13*kp31+kp13*kp21*kp32-kp11*kp23*kp32-
kp12*kp21*kp33=0
```

So we write this equation as:

$$Z^6 + a_1 Z^5 + a_2 Z^4 + a_3 Z^3 + a_4 Z^2 + a_5 Z + a_6 = 0$$

Where

a_1 = kv33+kv22+kv11

a_2 = -kv23*kv32+kp11+kp22-kv21*kv12+kv11*kv22+kp33+kv11*kv33+kv22*kv33-kv31*kv13

a3=kp11*kv22-kv11*kv23*kv32-kv13*kp31-kv21*kp12-kv31*kv13*kv22-kv12*kp21+kv21*kv13*kv32+kv11*kp33-kv31*kp13+
kp11*kv33+kv11*kv22*kv33-kv21*kv12*kv33+kv11*kp22+kp22*kv33+kv31*kv12*kv23+kv22*kp33-kp23*kv32-kv23*kp32

a4=-kp11*kv23*kv32-kv11*kp23*kv32-kv21*kv12*kp33-kp13*kp31+kv21*kv13*kp32+kv21*kp13*kv32+kp22*kp11+kv23*kv12*kp31
+kv31*kv12*kp23+kv31*kp12*kv23-kv22*kv13*kp31+kp22*kp33-kv31*kv13*kp22-
kv31*kp13*kv22-kv12*kp21*kv33-kp12*kp21-kv21*kp12*kv33+kv13*kp21*kv32+kv11*kv22*kp33+kp33*kp11
-kp23*kp32-kv11*kv23*kp32+kv11*kp22*kv33+kp11*kv22*kv33

a5=-kv22*kp13*kp31-kv12*kp21*kp33-kp11*kv23*kp32-kv21*kp12*kp33+kv13*kp21*kp32+kv23*kp12*kp31
+kp23*kv12*kp31+kv21*kp13*kp32+kp11*kp22*kv33-kp22*kv13*kp31+kp13*kp21*kv32+kv31*kp12*kp23
-kp11*kp23*kv32-kp12*kp21*kv33-kv11*kp23*kp32-kv31*kp13*kp22+kp11*kv22*kp33+kv11*kp22*kp33

a6=kp23*kp12*kp31+kp11*kp22*kp33-kp22*kp13*kp31+kp13*kp21*kp32-kp11*kp23*kp32-kp12*kp21*kp33

To determine the stability of a polynomial , one can simply compute the roots of the polynomial.

So using Routh Criterion, then $a_i > 0, i=1,...,n$

.

We can get the requirement of stability.

$$A = \begin{pmatrix} 0 & M \\ -\mathrm{M}^{-1} * Kp & -\mathrm{M}^{-1} * Kv \end{pmatrix}$$

$$\mathrm{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad Kp = \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix} \qquad Kv = \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix}$$

```
Mx=[1 0 0;0 1 0;0 0 1];
Kx=[k11 k12 k13;k21 k22 k23;k31 k32 k33];
Cx=[c11 c12 c13;c21 c22 c23;c31 c32 c33];
part1=[0 0 0;0 0 0;0 0 0];
part2=[1 0 0;0 1 0;0 0 1];
part3=-inv(Mx)*Kx;
part4=-inv(Mx)*Cx;
whole=[part1 part2;part3 part4];
```

```
evalue=eig(whole);
 evaluex=real(evalue);
 evaluex
```

```
evaluex =  -16.5378    5.8292    5.8292   -2.9461   -0.3716   -0.7789
evaluex =  -28.3576   -5.9326   -5.9326    2.8237   -0.7346    0.9206
evaluex =  -28.2471   -2.9038   -2.9038   -0.3336   -0.3336    1.0525
evaluex =  -34.5607   17.1064   -8.1646    0.7414   -0.1984   -1.3434
evaluex =  -24.8227   -9.7191   -0.9422   -0.9422   -0.2106   -0.2106
evaluex =  -28.7259    6.0426   -7.2991    0.4536   -1.3840   -0.4590
evaluex =  -28.1506    1.1907    1.1907   -0.0246   -1.7298   -1.7298
evaluex =  -41.2884  -16.0734   -5.7253   -0.2879   -0.2879    0.1309
evaluex =  -31.8587  -14.4796    4.6232    1.7373    0.0134   -0.4491
evaluex =  -34.6612    9.6515  -11.0541    0.1651   -0.5390   -1.3423
evaluex =  -35.3750  -13.7301    3.8697   -1.9434    0.1045   -0.6924
```

	Best solution(Diag)	Best solution(Full)	t_s (diag)	t_s (full)
1	2.3833 3.3421	[8.38415344 9.66491588 2.82934740 8.48853407 4.03860291 0.46114840 8.46650245 29.30018339 9.91270916 9.75393670 6.20485862 14.71557669 4.86861018 8.46452498 1.19715496 2.47733172 9.85699738 9.88032570]	3.7500	1.90000000
2	1.5217 1.9587	[1.72673409 1.44809618 1.19204083 1.71647225 1.91543135 1.20000000 1.63146158 1.10856461 1.12551764 1.59470321 1.28321770 1.26734351 2.52512542 1.55333279 1.09185410 1.51563279 1.20000000 1.86734762]	4.4310	1.40000000
3	1.3129 1.0761	[20.65988912 17.19640978 4.69860201 11.84740403 22.05042745 8.26414227 16.63362331 5.15625052 0.66985597 5.75128478 2.16823081 2.56862454 15.20386660 20.63099331 7.46930910 6.66251992 9.85351572 27.14483642]	6.4900	0.30000000
4	1.1431 1.0448	[11.06984902 18.72165141 1.62312409 9.31282423 12.64859416 6.83273190 6.71091474 5.91628608 7.81480318 2.74264273 9.97740450 1.92199301 3.13046826 17.45543587 3.20369254 15.61417119 4.85298320 14.62482400]	6.1510	0.40000000

Thank you!!