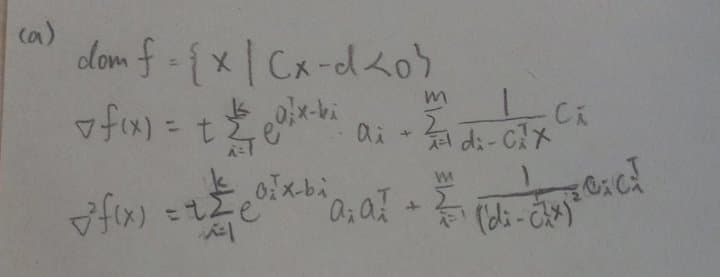
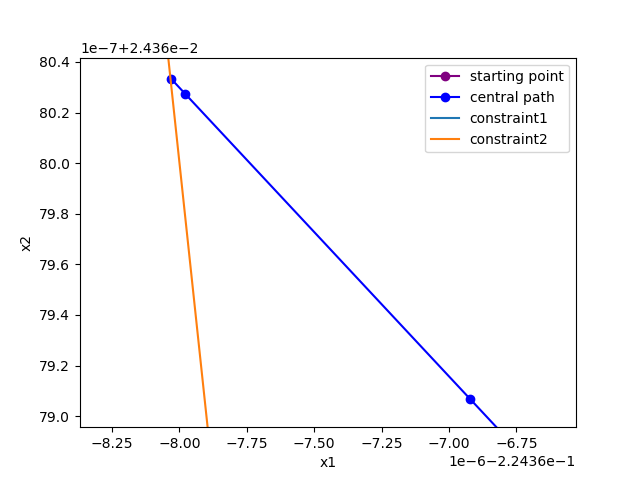
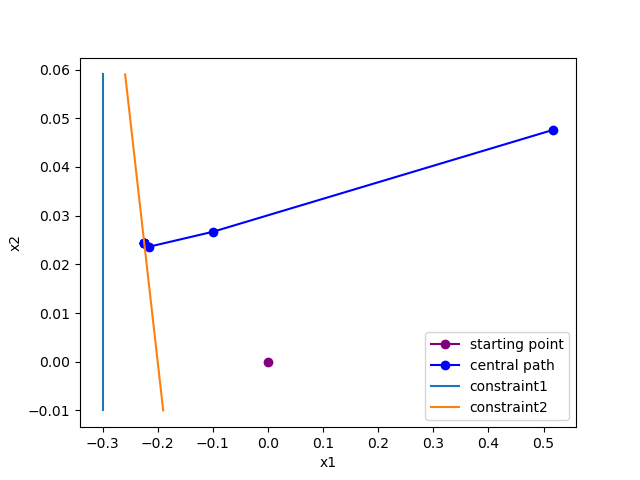
Report

(a)



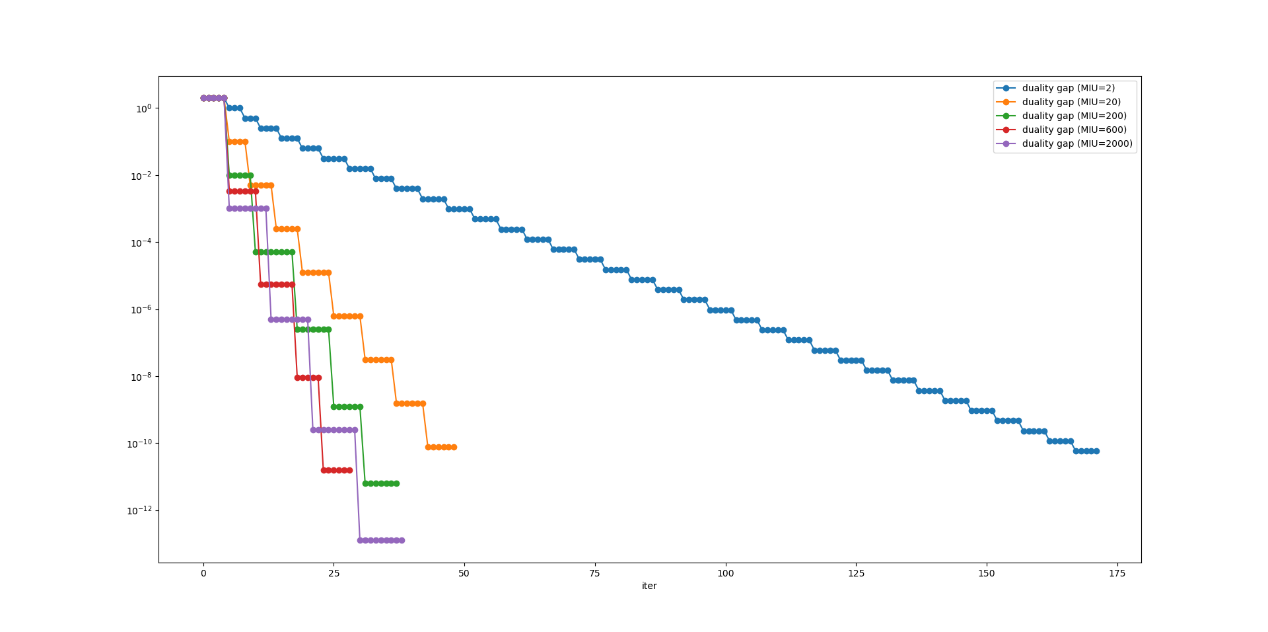
Setting 1:

(l)-(i), (l)-(ii)



(m)  
明顯的central path往boundary走。另外，只有constraint2(-x­1-x2<0.2)被activate，constraint1則inactivate。

(n)



μ越大則inner loop需要越多step，但一次可以使duality gap下降更多讓outerloop可以少一點step，因此中間的trade-off是影響效能的關鍵，在這個數字設定下μ=200~2000表現比小μ好(總step比較少)。

(o)

Mine:

Optimal value = 2.58226629

Optimal x = [-0.22436803, 0.02436803]

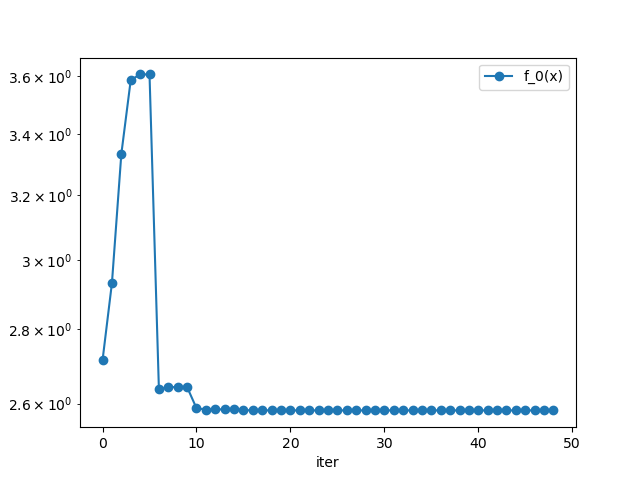
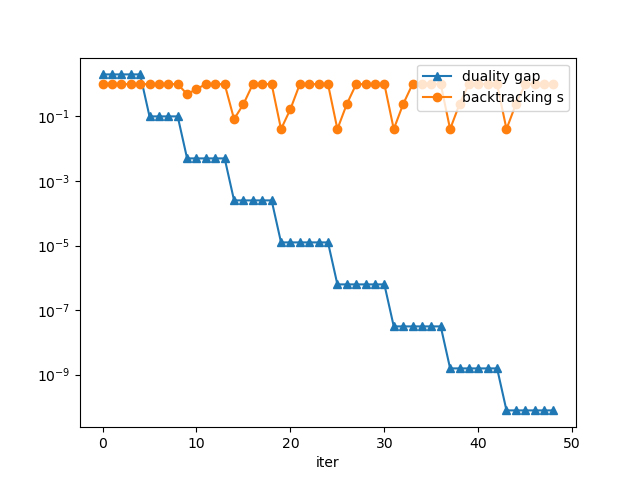
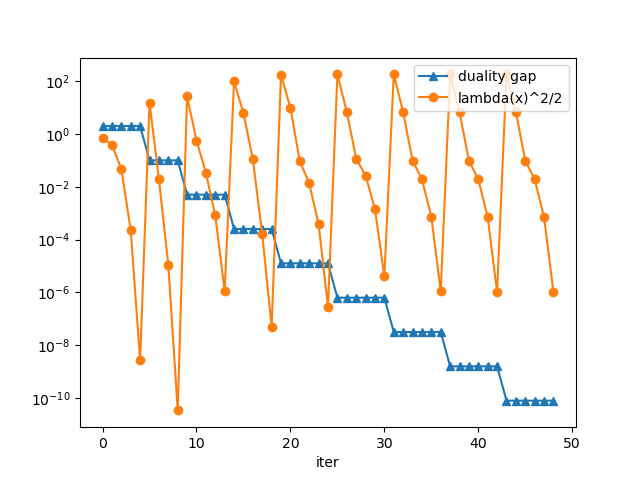
CVX toolbox:

Optimal value = 2.5822662770806897

Optimal x = [-0.22436803, 0.02436804]

我的結果與toolbox算出來的結果非常接近。

(p), (q), (r)



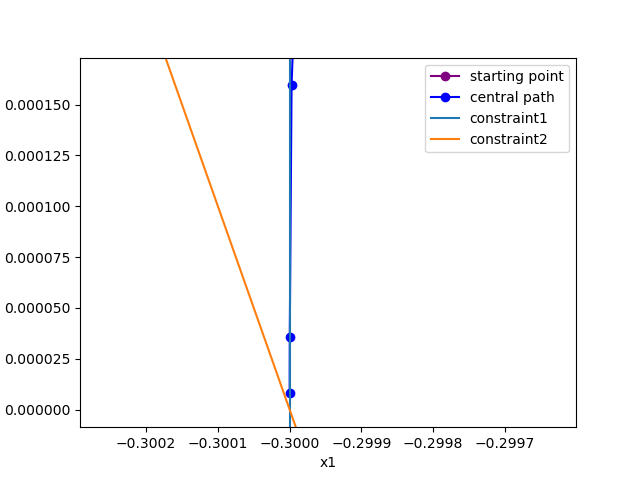
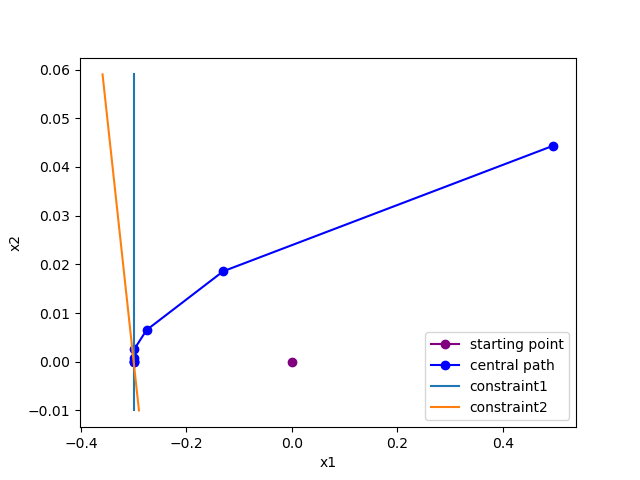
(p) 從λ2/2和duality gap可以看出來，每更新一次t(duality gap出現一個step)，inner loop像是用Newton’s method optimize一個新問題，λ2/2(這個新問題的duality gap)會在幾個step內下降到1e-5以下就跳出來、增大t、重複直到outer loop的duality gap夠小(1e-10)。

(q) 每更新一次t，前幾次的Newton step容易變太大導致走到infeasible的地方，所以s到很小才能走進feasible區域，後面越來越接近(該t的)optimal point後step就可走很準，所以s就不太需要縮小了(s=1就是最好的一步)。

(r) f0並沒有總是下降，但我們的方法仍是descent method，我們是照著f適合的方向走，也就是在f0+log barrier這個function上仍是總是下降的(在同一個t)，雖然一開始f和f0下降的方向不同導致f下降f0上升，但隨著t上升，f和f0越來越近似，所以f下降f0也會跟著下降最後到達optimal value。

Setting 2:

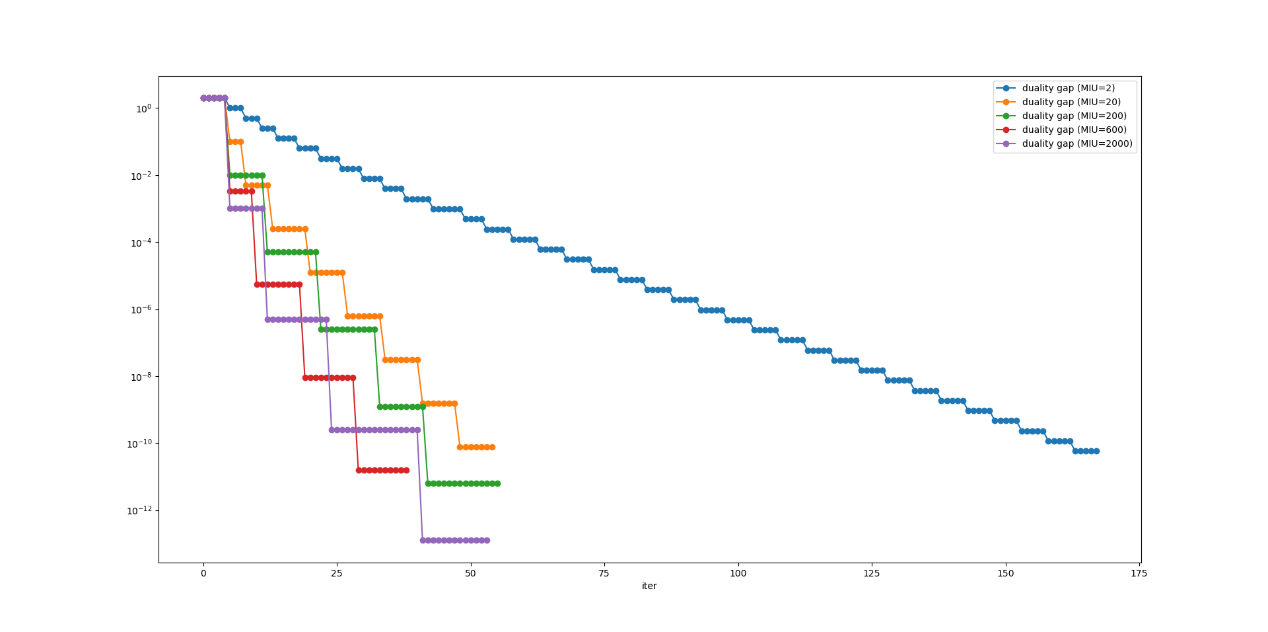
(l)-(i), (l)-(ii)



(m)

只有constraint1(-x­1<0.3)被activate，constraint2則inactivate。

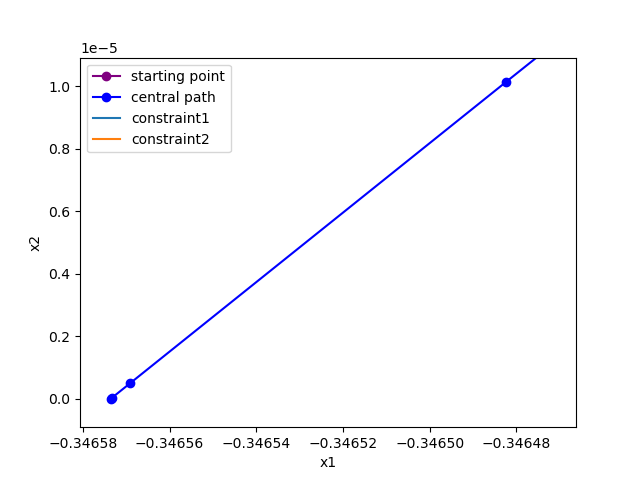
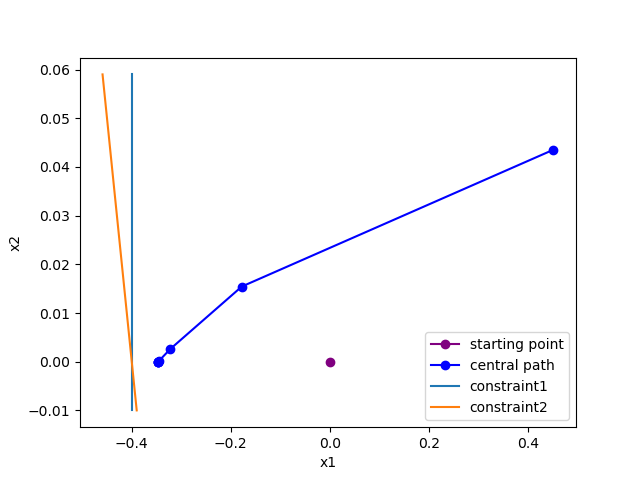
(n)



在這個數字設定下仍然是μ=200~2000表現比小μ好(總step比較少)。

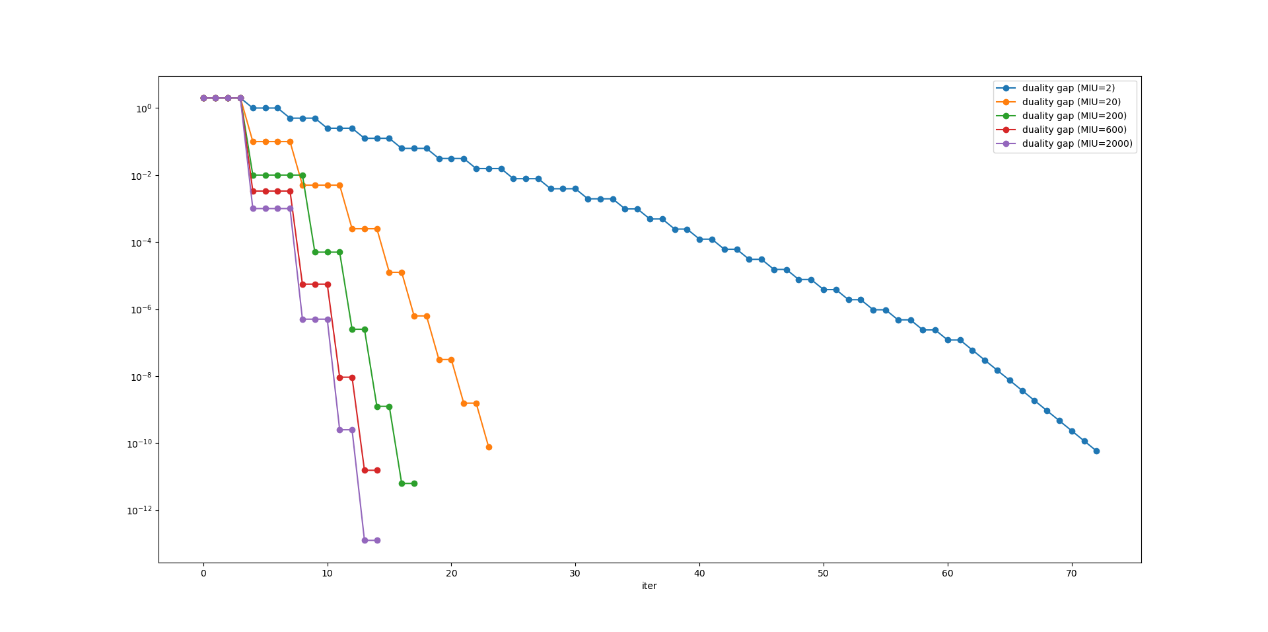
Setting 3:

(l)-(i), (l)-(ii)



(m)  
兩個constraint都沒有被activate。

(n)



這題因為constraint都不會被activate，所以outer loop其實可以直接走大步一點(inner loop都可以很快讓收斂)，所以μ=2000時表現最好。