Project 2: Constrained Optimization

Philippe Weingertner
pweinger@stanford.edu

1 Description of algorithms

The methods used, Interior Point Methods with Conjugate Gradient (CG) and BFGS are described in Kochenderfer and Wheeler [1]

The interior point method requires a feasible point from which to start the search: a Conjugate Gradient algorithm minimizing a quadratic penalty function is used.

$$p_{\text{quadratic}}(x) = \sum_i \max(c_i(x), 0)^2$$

To find a first feasible point, I tried other algorithms like BFGS, Momentum and Newton. They all work fine and pretty quickly but nevertheless Conjugate Gradient is the fastest: requiring the lowest number of evaluations, around ten evaluations on average.

1.1 For Simple1, Simple2, Simple3 and Secret 2

For the Interior search and all the problems except for Secret1, I used a BFGS search with a backtracking line search applied on a log barrier objective function:

$$p_{\text{barrier}}(x) = \begin{cases} \infty & \text{if } c_i(x) > 0 \\ -\sum_i \log(-c_i(x)) & \text{if } c_i(x) \geq -1 \text{ for some } i \\ -\sum_i 0 & \text{otherwise} \end{cases}$$

.

It provides better results than the inverse barrier. The hyper-parameters used are:

- ρ scheduling for increasing penalty: I typically start with $\rho = 10$ and $\min_x f(x) + \frac{1}{\rho} p_{\text{log\_barrier}}(x)$ with a BFGS algorithm. Then ρ is multiplied by 10. And so on.
- For the BFGS search: it is stopped when the norm of the gradient is below some threshold 0.01 for simple problems or $\frac{1}{\rho}$ for secret 2.
- For the Backtracking line search: I typically start with a maximum step size $\alpha_{\text{max}} = 1.4$ and check the first Wolfe condition for a maximum of 60 iterations, dividing the the step size by 2, $\frac{\alpha}{2}$, when the Wolfe condition is not met.
- The gradient of $f(x) + \frac{1}{\rho} p_{\text{log\_barrier}}(x)$ is computed via Finite Difference and a forward difference approximation.

For theses 4 different problems, there are very few differences. For Secret2, ρ is multiplied by 2 in between two BFGS searches.

1.2 For Secret1

This problem is different. Using the above method for Secret1, I tried to set the approximation of the Hessian to the Identity matrix and improved the results over BFGS. So I switched to a first order

method with Conjugate Gradient. Also the inverse barrier provides better results on this problem than the log barrier. In addition, the Backtracking Line search starts with a much larger step size with $\alpha_{\max} = 7.4$. So to summarize, to improve results there are 3 main changes:

- Use of Conjugate Gradient method instead of BFGS
- Use of Inverse barrier instead of Log barrier
- Starts Backtracking Line Search with a much larger step size

At some point I thought there could be many local optima and iterating with different starting points could help. But it provided no improvement.

1.3 Discussion

Interior Point method with BFGS is a solid method in general used by numerous standard optimization packages like Ipopt. So I assumed it was a reasonable starting point. The dimensionality of the problem is either $\mathbb{R}^2, \mathbb{R}^3, \mathbb{R}^{10}$or $\mathbb{R}^{60}$ which means approximating gradients via Finite Difference is still feasible. The hyper-parameters used for all the problems except Secret1 are pretty standard. They were tuned one at a time. The main peculiarity is with Secret1: which required using different settings and algorithms (CG instead of BFGS and Inverse barrier instead of Log barrier) and improved by using much larger step sizes at some point. We were told the objective function of Secret1 is highly non-convex: so I looked in 2 directions, many local minima and not using 2nd order approximation.

Secret2 is a Linear Problem with Linear constraints. So a simplex method would probably do much better than the Interior Point method I implemented but we were not given the coefficients of the linear constraints and objectives.

One pro of the methods used is that they are pretty general or generic and supported by state of the art solvers. We also exploit gradient information to guide the search. One con is that I am not using any kind of stochasticity and no population methods: so I may be trapped in local minima and miss the global optimum. Now with a limited number of evaluations I was reluctant to test population methods.

2 Comparison of algorithms

2.1 Feasible region, contour plot and paths taken

Interior Point Method:

- Feasibility Search: with Conjugate Gradient
- Interior Search: with BFGS and either log barrier or inverse barrier

In the plots below, the infeasible region is in red: where I used contour plots with positive values for the constraints to represent these infeasible areas.

2.1.1 Simple1, with log barrier

```
simple1 logbarrier max_count=1953
mean: interior score = 2.4863785232082213e-5, count=982.632
max rho = 1000000
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

We quickly enter the feasible area.

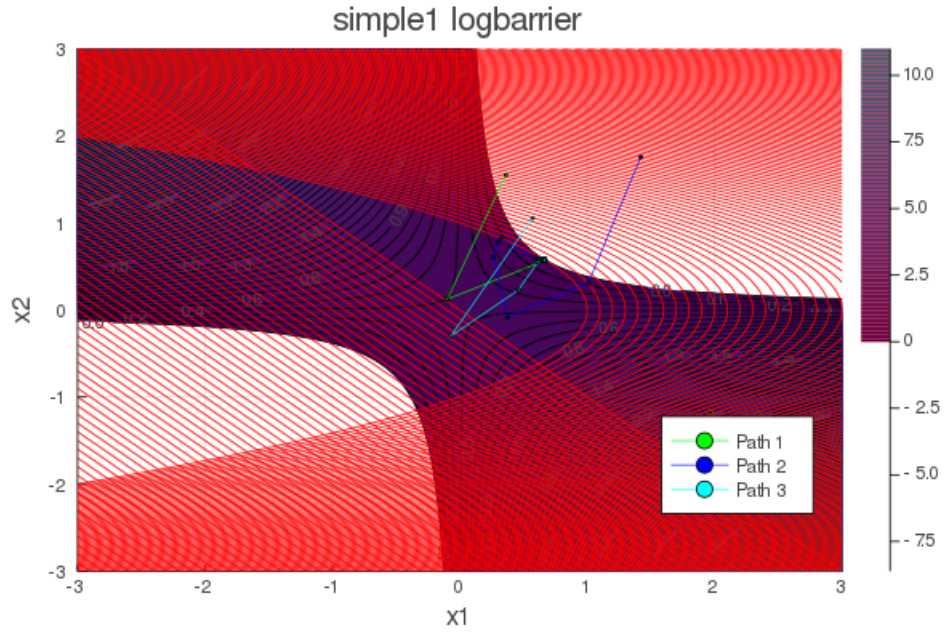

Figure 1: Paths for simple1 with log barrier

2.1.2 Simple1, with inverse barrier

```
simple1 invbarrier max_count=1935
mean: interior score = 0.00026045196654097326, count=1143.504
max rho = 1000000000
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

With the inverse barrier function we see much more zigzags than with the log barrier function, slowing down our progress to a minimum.

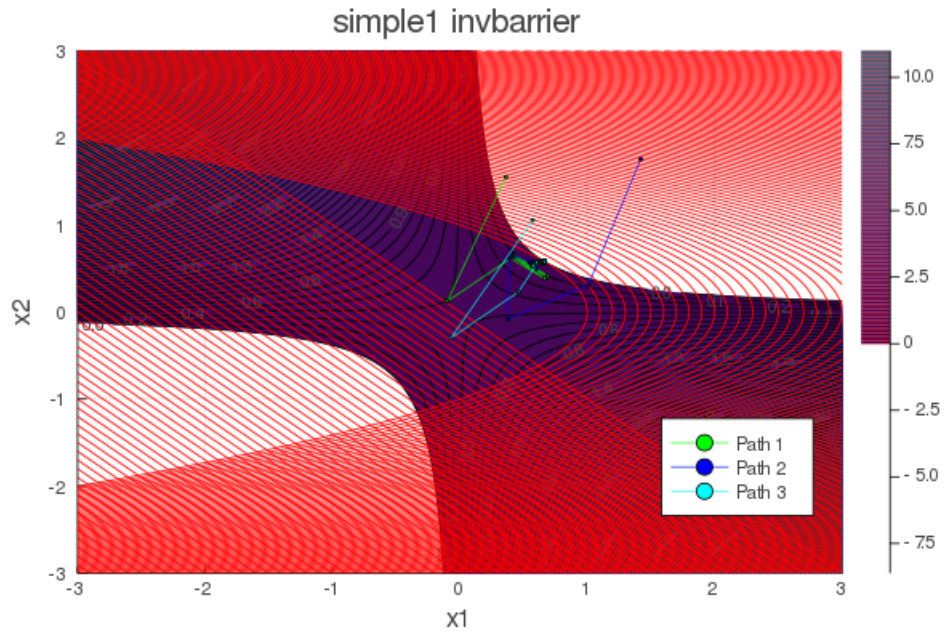

Figure 2: Paths for simple1 with inverse barrier

2.1.3 Simple2, with log barrier

```
simple2 logbarrier max_count=1952
mean: interior score = 0.011675717139669722, count=1929.088
max rho = 31250
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

What is a surprise for me, looking at the plot, is the jump over an infeasible area to progress through the minima. The 3 paths are overlay-ed here: but they all take a similar path and jump.

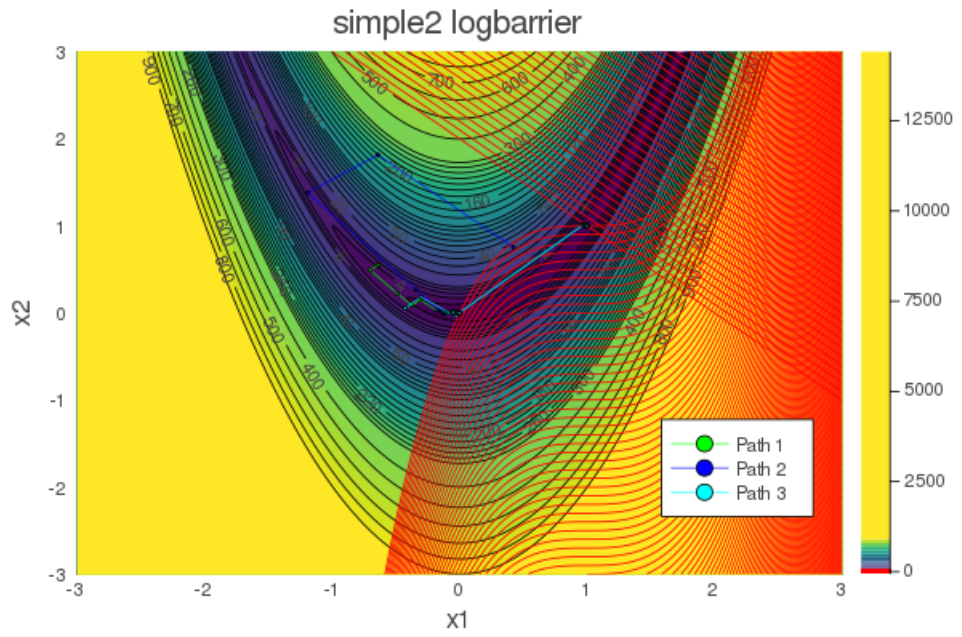

Figure 3: Paths for simple2 with log barrier

2.1.4 Simple2, with inverse barrier

```
simple2 invbarrier max_count=1951
mean: interior score = 0.5247673189083747, count=1925.392
max rho = 781250
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

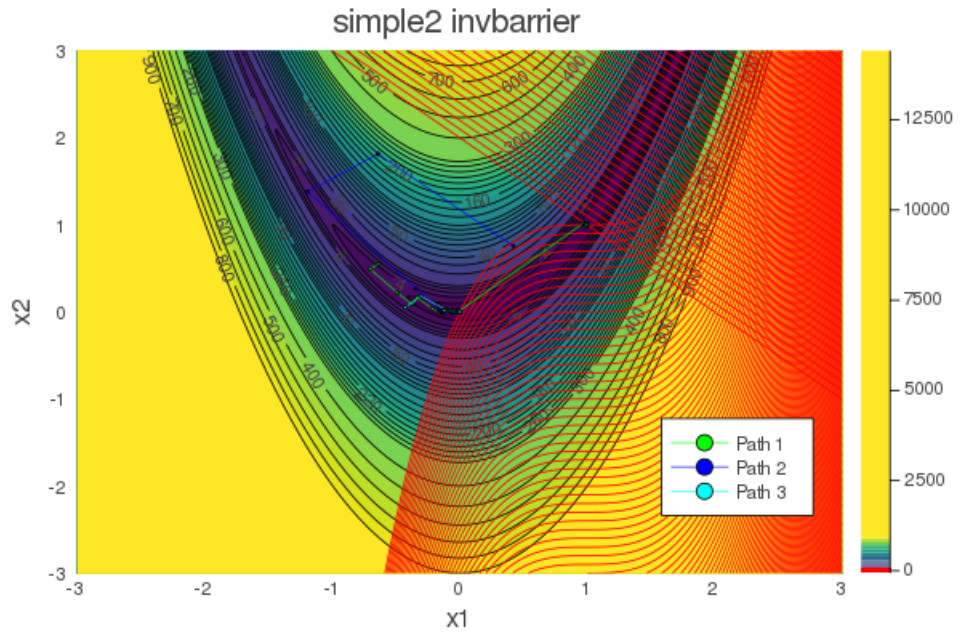

Figure 4: Paths for simple2 with inverse barrier

2.2 Objective function and maximum constraint

2.2.1 Simple2, with log barrier

```
simple2 logbarrier max_count=1952
mean: interior score = 0.011675717139669722, count=1929.088
max rho = 31250
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

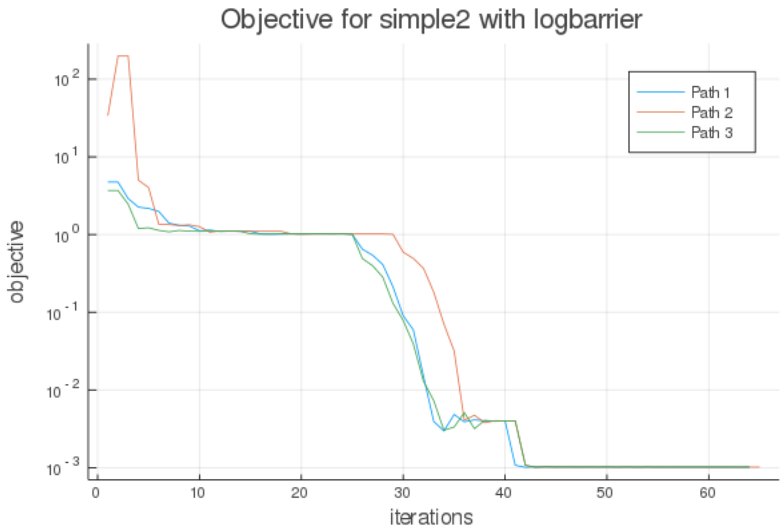

Figure 5: Objective for simple2 with log barrier

The constraints are only violated at the very beginning and then we always remain in the negative value area.

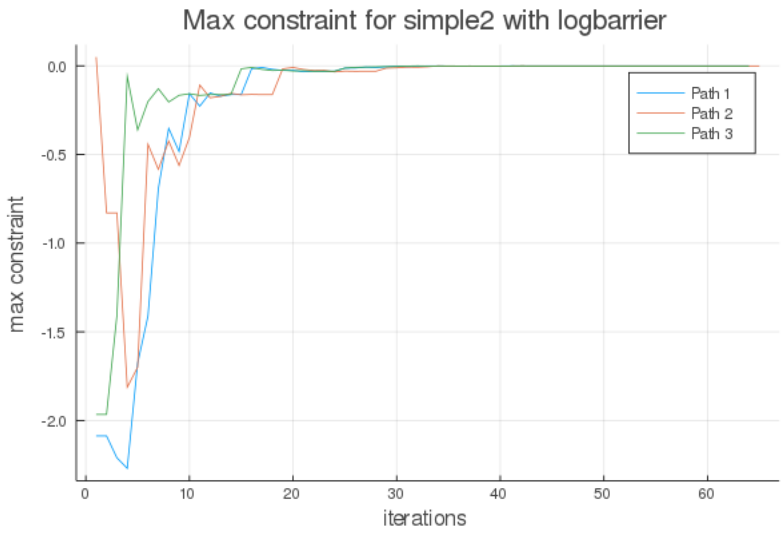

Figure 6: Max constraint for simple2 with log barrier

2.2.2 Simple2, with inverse barrier

```
simple2 invbarrier max_count=1951
mean: interior score = 0.5247673189083747, count=1925.392
max rho = 781250
Pass: optimize returns a feasible solution on 500/500 random seeds.
```

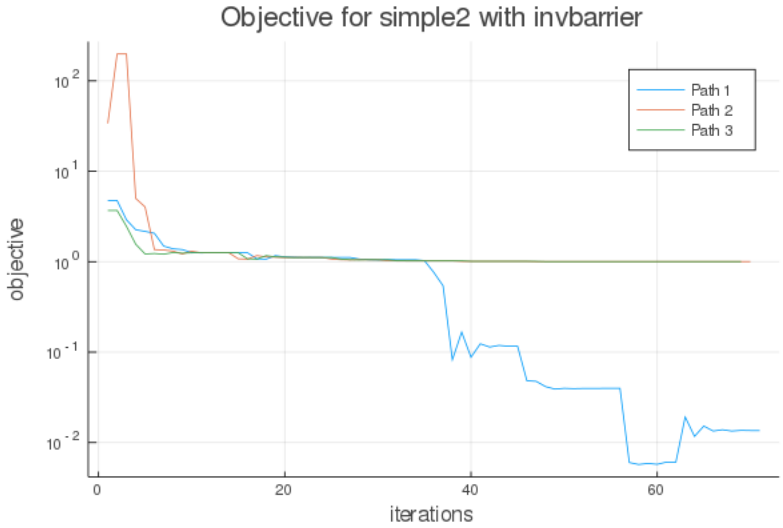

Figure 7: Objective for simple2 with inverse barrier

The constraints are only violated at the very beginning and then we always remain in the negative value area.

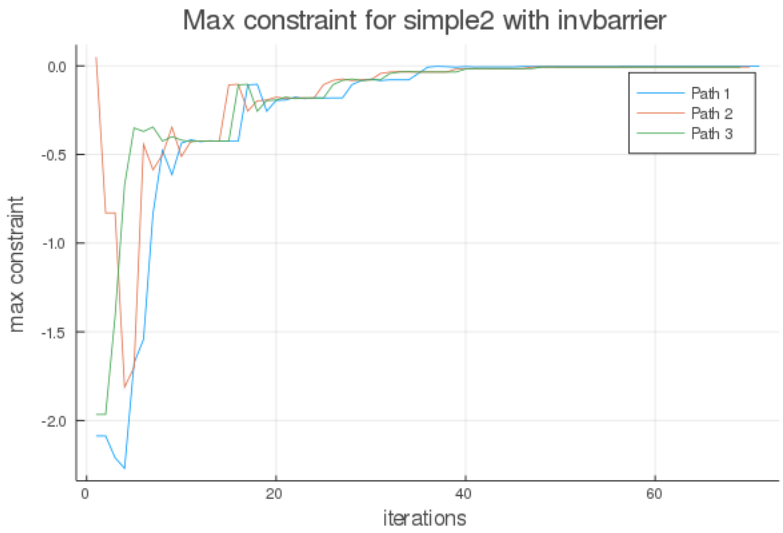

Figure 8: Max constraint for simple2 with inverse barrier

References

[1] Mykel J. Kochenderfer and Tim A. Wheeler. *Algorithms for Optimization*. The MIT Press, 2019.