

Numerical optimization technique for constrained optimization:

Penalty function combined with Newton's method

Wenying Shen 05/10/2021

1. A constrained optimization problem

The general form of a non-linear constrained optimization is given as:

$$\begin{aligned} (NLP) \quad & \text{Min } f(X) \\ & \text{s.t. } g_i(X) \leq 0; \quad (i = 1, 2, \dots, m) \\ & \text{where } f : R^n \rightarrow R, g_i : R^n \rightarrow R; i = 1, 2, \dots, m \text{ are differentiable convex functions.} \end{aligned}$$

A numerical optimization technique or constrained optimization problem aims at converting the NPL to an unconstrained optimization problem, which can be solved using numerical techniques for unconstrained problems. One of the methods to convert the constrained NLP into an unconstrained problem is the introduction of a penalty function.

Consider the following function:

$$p(X) = \begin{cases} 0; & X \in S \\ +\infty; & X \notin S \end{cases}$$

where $S = \{X \in R^n \mid g_i(X) \leq 0 \ (i = 1, 2, \dots, m)\}$ is the feasible set.

The smooth function is defined as:

$$P(X) = \sum_{i=1}^m [\max(g_i(X), 0)]^2$$

Introduction of $P(X)$ in the objective function converts the NLP into equivalent unconstrained

minimization problem as: $(UMP)_\mu \quad \text{Min}_{X \in R^n} f(X) + \mu P(X)$

2. Algorithm for penalty function method

(1) Choose a termination tolerance $\varepsilon = 10^{-5}$.

(2) Choose a suitable penalty function: $P(X)$.

(3) Choose an increasing sequence of positive real numbers which tends to $+\infty$, i.e. a sequence

$\{\mu_k\}_{k=1}^\infty$ such that for each k , $\mu_k > 0, \mu_{k+1} > \mu_k, \{\mu_k\} \rightarrow \infty$. In general, we take

$\mu_1 = 1, \mu_2 = 10, \mu_3 = 100, \mu_4 = 1000$ and so on. (scalar $\Delta = 10$)

(4) Choose an arbitrary point that violate constraints: $X_0, X_0 \in R^n$. Construct the following

unconstrained minimization problem: $(UMP)_{\mu_1} \min_{X \in R^n} f(X) + \mu_1 P(X)$ and solve it using $e.g. \mu_1 = 1$

Newton's method, starting with X_0 . Let X_1 be the optimal solution $(UMP)_{\mu_1}$, set $k = 1$.

(5) Construct $(UMP)_{\mu_{k+1}}$ as $(UMP)_{\mu_{k+1}} : \min z(X, \mu_{k+1}) = f(X) + \mu_k P(X)$ and solve it using Newton's method, starting with X_{k+1} , where X_{k+1} is the optimal solution of $(UMP)_{\mu_{k+1}}$.

(6) Stopping criteria: Continue iterations of step (5) till $\mu_k P(X_k) < \varepsilon$ or $z(X_k, \mu_k) - f(X_k) < \varepsilon$ for some tolerance level $\varepsilon > 0$.¹

3. Compare results between Python and Maple

Example 1: Problem Set 3 Question 5 (using minimize problem instead)

$$\text{Minimize } f = (x_1 - 2)^2 + (x_2 - 2)^2$$

$$s.t. \quad x_1 + 2x_2 \leq 3$$

$$8x_1 + 5x_2 \geq 10$$

$$x_1, x_2 \geq 0$$

$$\Rightarrow z = (x_1 - 2)^2 + (x_2 - 2)^2 + \mu \max\{0, (x_1 + 2x_2 - 3)^2\} + \mu \max\{0, (-8x_1 - 5x_2 + 10)^2\}$$

$$\text{set } X = (x_1, x_2), \text{ tolerance } \varepsilon = 10^{-5}$$

Python result:

Starting point X_0 is (0, 0).

Iteration k	μ_k	X_k	$f(X_k)$	$z(X_k, \mu_k)$	$\mu_k P(X_k)$
1	1	(1.4900, 1.0000)	1.2499	1.5000	0.2500
2	10	(1.4117, 0.8235)	1.7301	1.7647	0.0346
3	100	(1.4012, 0.8023)	1.7928	1.7964	0.0035
4	1000	(1.4001, 0.8000)	1.7999	1.7999	0.00035
5	10000	(1.4000, 0.8000)	1.7999	1.7999	3.5986e-06

Optimal solution is $x_1=1.4, x_2=0.8$; Optimal value of f is 1.7999.

1. Lecture 30: Newton's and Penalty Function Methods https://www.youtube.com/watch?v=z_-iUPg4Iwk

Maple result:

The screenshot shows the Maple software interface. The title bar indicates the file is named 'exemple1.mw'. The menu bar includes 'Start.mw', 'exemple1.mw', and 'C 2D Output'. The toolbar contains various icons for text, math, and editing. The worksheet area displays the following Maple code and results:

```

> f := (x1 - 2)^2 + (x2 - 2)^2;
                                     f := (x1 - 2)^2 + (x2 - 2)^2
                                     (1)

> c1 := x1 + 2*x2 - 3;
                                     c1 := x1 + 2*x2 - 3
                                     (2)

> c2 := 8*x1 + 5*x2 - 10;
                                     c2 := 8*x1 + 5*x2 - 10
                                     (3)

>
with(Optimization);
NLPsolve(f, {c1 ≤ 0, c2 ≥ 0, x1 ≥ 0, x2 ≥ 0});

[1.800000000000000004, [x1 = 1.400000000000000, x2 = 0.800000000000000]]
                                     (4)

```

Example 2: Problem Set 3 Question 3

$$\text{Minimize } f = 4 + 3(1 - x_1)^2 + (1 - x_2)^2$$

$$s.t. \quad 3x_1 + x_2 = 5$$

$$\Rightarrow z = 4 + 3(1 - x_1)^2 + (1 - x_2)^2 + \mu(3x_1 + x_2 - 5)^2$$

set $X = (x_1, x_2)$, tolerance $\varepsilon = 10^{-5}$

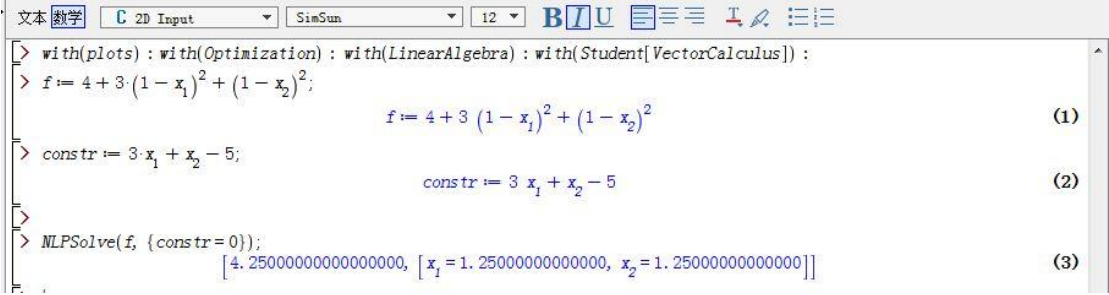
Python result:

Starting point X_0 is $(0, 0)$.

Iteration k	μ_k	X_k	$f(X_k)$	$z(X_k, \mu_k)$	$\mu_k P(X_k)$
1	1	(1.1999,1.2000)	4.1599	4.2000	0.0400
2	10	(1.2439,1.2339)	4.2379	4.2439	00059
3	100	(1.2493,1.2493)	4.2487	4.2493	0.0006
4	1000	(1.2499,1.2499)	4.2498	4.2499	6.2479e-05
5	10000	(1.2499,1.2500)	4.2499	4.2499	6.2598e-06

Optimal solution is $x_1=1.2499$, $x_2=1.2500$; Optimal value of f is 4.2499.

Maple result:



The screenshot shows the Maple software interface with a menu bar at the top containing '文本' (Text), '数学' (Math), 'C 2D Input', 'SimSun', '12', and various icons. The main workspace displays the following commands and results:

```
> with(plots): with(Optimization): with(LinearAlgebra): with(Student[VectorCalculus]):  
> f := 4 + 3*(1 - x1)^2 + (1 - x2)^2;  
f := 4 + 3 (1 - x1)^2 + (1 - x2)^2 (1)  
> constr := 3*x1 + x2 - 5;  
constr := 3 x1 + x2 - 5 (2)  
>  
> NLPsolve(f, {constr=0});  
[4.2500000000000000, [x1 = 1.2500000000000000, x2 = 1.2500000000000000]] (3)
```

4. Appendix: Python code