Ricerca Operativa - Laboratorio

Lezione 8 - Risoluzione di Problemi di Programmazione Non Lineare

Docente: Luigi De Giovanni

Dipartimento di Matematica “Tullio Levi-Civita”
Università degli Studi di Padova

`luigi@math.unipd.it`
`https://www.math.unipd.it/∼luigi/`

Corso di Laurea Magistrale in Matematica
Università degli Studi di Padova
a.a. 2019–2020

Problemi di PNL: modelli e risolutori

Per risolvere un problema di programmazione non lineare in AMPL, si può usare il solutore MINOS (usa metodi basati su gradiente proiettato e funzioni di penalità).

Problemi di PNL: modelli e risolutori

Per risolvere un problema di programmazione non lineare in AMPL, si può usare il solutore MINOS (usa metodi basati su gradiente proiettato e funzioni di penalità).

N.B. Nel caso non lineare, l'ottenimento di una soluzione non è garantito!

Problemi di PNL: modelli e risolutori

Per risolvere un problema di programmazione non lineare in AMPL, si può usare il solutore MINOS (usa metodi basati su gradiente proiettato e funzioni di penalità).

N.B. Nel caso non lineare, l'ottenimento di una soluzione non è garantito!

$$\Downarrow$$

Può essere necessario:

- inizializzare l'algoritmo in maniera opportuna...
  Per inizializzare le variabili si usa:
  - nel file `.dat` o nel file `.run`, il comando `let x := 1`
  - oppure si può scrivere `var x := 1` nel file `.mod`
- riformulare in modo opportuno (e.g. trasformare la f.o.)

Problemi di PNL: un esempio (i)

Esempio 8.1

Dato il parametro $L \geq 0$, determinare i lati a, b e c del triangolo di area massima il cui perimetro sia uguale a L (si consideri il caso $L = 100$).

Problemi di PNL: un esempio (i)

Esempio 8.1

Dato il parametro $L \geq 0$, determinare i lati a, b e c del triangolo di area massima il cui perimetro sia uguale a L (si consideri il caso $L = 100$).

È possibile dimostrare che la soluzione del problema è $a = b = c = L/3$, per ogni valore di $L \geq 0$.

Problemi di PNL: un esempio (i)

Esempio 8.1

Dato il parametro $L \geq 0$, determinare i lati a, b e c del triangolo di area massima il cui perimetro sia uguale a L (si consideri il caso $L = 100$).

È possibile dimostrare che la soluzione del problema è $a = b = c = L/3$, per ogni valore di $L \geq 0$.

Formulare il problema come modello di programmazione matematica e verificare se MINOS riesce a risolverlo a partire da $a = b = c = 1$ e da diverse altre soluzioni iniziali.

Problemi di PNL: un esempio (ii)

triangolo0.mod

```
param L >= 0; # perimetro

var x := 1, >=0;
var y := 1, >=0;
var z := 1, >=0;

# formula di Erone per il calcolo dell'area
maximize f :  sqrt(L/2*(L/2-x)*(L/2-y)*(L/2-z));

s.t. v_perimetro: x + y + z = L;
```

Problemi di PNL: un esempio (iii)

triangolo.mod

```
param L >= 0; # perimetro

var x := 1, >=0;
var y := 1, >=0;
var z := 1, >=0;

# formula di Erone per il calcolo del quadrato dell'area
maximize f :  L/2*(L/2-x)*(L/2-y)*(L/2-z);

s.t. v_perimetro: x + y + z = L;
```

Problemi di PNL: un esempio (iv)

triangolo.dat

```
param L := 100;
```

triangolo.run

```
reset;
#model triangolo0.mod;
model triangolo.mod;
data triangolo.dat;

#let x:=30; let y:=30; let z:=30;

option solver MINOS;
solve;

display sqrt(f); #f
display x,y,z;
```

Problemi di PNL: modelli e solutori bis

Le difficoltà incontrate dai solutori per PNL durante il processo di risoluzione sono più marcate nel caso di:

- non linearità molto marcate,
- funzioni non differenziabili o non continue.

Problemi di PNL: modelli e solutori bis

Le difficoltà incontrate dai solutori per PNL durante il processo di risoluzione sono più marcate nel caso di:

- non linearità molto marcate,
- funzioni non differenziabili o non continue.

In questi casi bisogna considerare gli accorgimenti già accennati:

- fare attenzione al punto iniziale,
- riformulare per cercare di eliminare (se possibile) anomalie.

Problemi di PNL: un altro esempio (i)

Esempio 8.2

Sia dato l'intero $N > 0$ e la sfera in $\mathbb{R}^3$ di centro l'origine e raggio $R > 0$. Determinare la posizione di N punti sulla sfera, tali che questi risultino a distanza massima tra loro (si consideri il caso numerico $R = 100$ e $N = 2$).

Problemi di PNL: un altro esempio (i)

Esempio 8.2

Sia dato l'intero $N > 0$ e la sfera in $\mathbb{R}^3$ di centro l'origine e raggio $R > 0$. Determinare la posizione di N punti sulla sfera, tali che questi risultino a distanza massima tra loro (si consideri il caso numerico $R = 100$ e $N = 2$).

Osservazione: nel caso $N = 2$, i punti devono trovarsi da parti opposte rispetto ad un qualsiasi asse della sfera.

Problemi di PNL: un altro esempio (i)

Esempio 8.2

Sia dato l'intero $N > 0$ e la sfera in $\mathbb{R}^3$ di centro l'origine e raggio $R > 0$. Determinare la posizione di N punti sulla sfera, tali che questi risultino a distanza massima tra loro (si consideri il caso numerico $R = 100$ e $N = 2$).

Osservazione: nel caso $N = 2$, i punti devono trovarsi da parti opposte rispetto ad un qualsiasi asse della sfera.

Formulare il problema come modello di programmazione matematica e usare l'osservazione precedente per verificare se MINOS riesce a risolverlo.

Problemi di PNL: un altro esempio (ii)

Una prima formulazione del problema:

Problemi di PNL: un altro esempio (ii)

Una prima formulazione del problema:

$$\max \min_{i,j \in \{1..N\}} \|x_i - x_j\|$$
$$\|x_i\| \leq R \quad \forall\, i \in \{1..N\}$$
$$x_i \in \mathbb{R}^3 \quad \forall\, i \in \{1..N\}$$

Problemi di PNL: un altro esempio (ii)

Una prima formulazione del problema:

$$
\begin{aligned}
\max \quad & \min_{i,j \in \{1..N\}} \|x_i - x_j\| \\
& \|x_i\| \leq R \quad \forall\, i \in \{1..N\} \\
& x_i \in \mathbb{R}^3 \quad \forall\, i \in \{1..N\}
\end{aligned}
$$

Accorgimento 1: riformulare la funzione $\min$

Problemi di PNL: un altro esempio (iii)

sfera1.mod

```
param R; # raggio
param N; # numero punti

var x{j in 1..N} := 0;
var y{j in 1..N} := 0;
var z{j in 1..N} := 0;
var t := 0, >=0;

maximize f : t;

s.t. v_dist{i in 1..N, j in 1..N : i<>j} :
  t <= sqrt((x[i]-x[j])**2+(y[i]-y[j])**2+(z[i]-z[j])**2);

s.t. v_sfera{i in 1..N} :
  sqrt(x[i]**2 + y[i]**2 + z[i]**2) <= R;
```

Problemi di PNL: un altro esempio (iv)

Accorgimento 2: evitare ulteriori non-smoothess (`sqrt`)

sfera2.mod

```
param R; # raggio
param N; # numero punti

var x{j in 1..N} := 0;
var y{j in 1..N} := 0;
var z{j in 1..N} := 0;
var t := 0, >=0;

maximize f : t**2;

s.t. v_dist{i in 1..N, j in 1..N : i<>j} :
  t**2 <= (x[i]-x[j])**2+(y[i]-y[j])**2+(z[i]-z[j])**2;

s.t. v_sfera{i in 1..N} :
  x[i]**2 + y[i]**2 + z[i]**2 <= R**2;
```

Problemi di PNL: un altro esempio (v)

Accorgimento 3: cambiare soluzione iniziale

sfera3.mod

```
param R; # raggio
param N; # numero punti

var x{j in 1..N} := 0;
var y{j in 1..N} := 0;
var z{j in 1..N} := 1*(-1)^j;
var t := 1, >=0;

maximize f : t**2;

s.t. v_dist{i in 1..N, j in 1..N : i<>j} :
  t**2 <= (x[i]-x[j])**2+(y[i]-y[j])**2+(z[i]-z[j])**2;

s.t. v_sfera{i in 1..N} :
  x[i]**2 + y[i]**2 + z[i]**2 <= R**2;
```

Problemi di PNL: un altro esempio (vi)

sfera.dat

```
param R := 100;
param N := 2;
```

sfera.run

```
reset;
#model sfera1.mod
#model sfera2.mod
model sfera3.mod
data sfera.dat;

option solver MINOS;
solve;

display t;
display x,y,z;
```

Esercizio proposto (i)

Esercizio 8.1

Si assuma di avere n titoli con rendimento aleatorio nei prossimi T periodi. Sia $b = 1$ il budget disponibile. Determinare come comporre un portafoglio ottimo (i.e., con varianza minima) al variare del rendimento minimo atteso r_m tra 1.01 e 1.1 (con passo 0.01). Mostrare, ad ogni iterazione, il portafoglio ottenuto (cioè l'elenco dei titoli, includendo solo quelli con quantità da investire > 0.0001), il valor medio e la varianza del rendimento. Si consideri il caso in cui i rendimenti sono distribuiti in modo uniforme tra 0,8 e 1,3.

Esercizio proposto (ii)

Ricordiamo che, se r_{it} è il rendimento del titolo $i = 1, \dots, n$ nell'intervallo temporale $t = 1, \dots, T$, abbiamo:

- rendimento medio: $\bar{r}_i = \dfrac{1}{T} \displaystyle\sum_{t=1}^{T} r_{it}, \quad i = 1, \dots, n,$
- covarianza: $\Sigma_{ij} = \dfrac{1}{T} \displaystyle\sum_{t=1}^{T} (r_{it} - \bar{r}_i)(r_{jt} - \bar{r}_j), \quad i, j = 1, \dots, n.$

Esercizio proposto (ii)

Ricordiamo che, se r_{it} è il rendimento del titolo $i = 1, \ldots, n$ nell'intervallo temporale $t = 1, \ldots, T$, abbiamo:

- rendimento medio: $\bar{r}_i = \dfrac{1}{T} \displaystyle\sum_{t=1}^{T} r_{it}, \quad i = 1, \ldots, n,$
- covarianza: $\Sigma_{ij} = \dfrac{1}{T} \displaystyle\sum_{t=1}^{T} (r_{it} - \bar{r}_i)(r_{jt} - \bar{r}_j), \quad i, j = 1, \ldots, n.$

Formulazione:

- Variabili: x_i è la quantità di denaro investita nel titolo $i = 1, \ldots, n$
- Funzione obiettivo: $x^T \Sigma x$ (varianza del rendimento da minimizzare)
- Vincoli:
  $e^T x = b$ (budget)
  $\bar{r}^T x \geq r_m$ (rendimento minimo atteso)
  $x \geq 0$ (no vendite allo scoperto)

Esercizio proposto (iii)

portfolio.mod

```
param t_begin; param t_end;
param n; # numero titoli
set A := 1..n; # insieme asset
set T := {t_begin..t_end}; # intervallo temporale

param b; # budget
param r_min; # rendimento minimo atteso
param u_min; param u_max; # estremi distr. uniforme
param r{A,T} := Uniform(u_min,u_max); # rendimenti titoli

# calcolo rendimento medio
param r_avg{i in A} := (sum{t in T} r[i,t]) / card(T);

# calcolo matrice covarianza
param cov{i in A, j in A} :=
(sum{t in T}(r[i,t]-r_avg[i])*(r[j,t]-r_avg[j])) /card(T);
```

Esercizio proposto (iv)

```
# modello
var x{A} >=0;

minimize f: sum{i in A, j in A} cov[i,j]*x[i]*x[j] ;

subject to v_budget: sum{i in A} x[i] = b;
subject to v_reward: sum{i in A} r_avg[i]*x[i] >= r_min;
```

portfolio.dat

```
param t_begin := 1; param t_end := 22;
param n := 8;
param b := 1;
param u_min := 0.8; param u_max := 1.3;
```

Esercizio proposto (v)

portfolio.run

```
reset;
option randseed 1;
model portfolio.mod;
data portfolio.dat;

for {q in 1.01..1.1 by 0.01} {
  let r_min := q;
  printf "\nRendimento minimo atteso = %f\n", r_min > "log.txt";
  solve;
  printf "Valor medio rendimento = %f\n" ,
  sum{i in A} r_avg[i]*x[i] > "log.txt" ;
  printf "Varianza rendimento = %f\n", f > "log.txt";
  printf "Composizione portafoglio:\n" > "log.txt";
  for {i in A} {
    if (x[i] > 0.0001) then {
      printf "x[%d] = %f\n", i, x[i]  > "log.txt";
    }
  }
}
```

Esempi di algoritmi di ottimizzazione globale (i)

Esempio 8.3

Si richiede di posizionare n cerchi all'interno del quadrato con vertici $\{0, 1\}^2$ in modo che i cerchi non si sovrappongano e il raggio, uguale per tutti, sia il più grande possibile.

Risolvere il problema con un algoritmo multi-start.

Esempi di algoritmi di ottimizzazione globale (ii)

Variabili:

- x_i è la coordinata x del centro dell'i-esimo cerchio,
- y_i è la coordinata y del centro dell'i-esimo cerchio,
- r è il raggio dei cerchi.

$$
\begin{aligned}
& \max r \\
& \qquad x_i - r \geq 0, \quad i = 1, \ldots, n \\
& \qquad x_i + r \leq 1, \quad i = 1, \ldots, n \\
& \qquad y_i - r \geq 0, \quad i = 1, \ldots, n \\
& \qquad y_i + r \leq 1, \quad i = 1, \ldots, n \\
& \qquad (x_i - x_j)^2 + (y_i - y_j)^2 \geq (2r)^2, \quad i, j = 1, \ldots, n, \ i \neq j \\
& \qquad r \geq 0
\end{aligned}
$$

Esempi di algoritmi di ottimizzazione globale (iii)

Raggio massimo al variare di *n* (per confrontare la soluzione ottenuta):

1	0.500000000000
2	0.292893218813
3	0.254333095030
4	0.250000000000
5	0.207106781187
6	0.187680601147
7	0.174457630187
8	0.170540688701
9	0.166666666667
10	0.148204322565
11	0.142399237696
12	0.139958844038
13	0.133993513499
14	0.129331793710
15	0.127166547515
16	0.125000000000
17	0.117196742783
18	0.115521432464
19	0.112265437571
20	0.111382347512

Esempi di algoritmi di ottimizzazione globale (iv)

cerchi.mod

```
param N;
set Cerchi := 1..N;

var x{Cerchi} := Uniform(0,1);
var y{Cerchi} := Uniform(0,1);
var r := Uniform(0,1), >= 0;

maximize f : r;

s.t. v_quad_x_l{i in Cerchi}: x[i] - r >= 0;
s.t. v_quad_x_u{i in Cerchi}: x[i] + r <= 1;
s.t. v_quad_y_l{i in Cerchi}: y[i] - r >= 0;
s.t. v_quad_y_u{i in Cerchi}: y[i] + r <= 1;
s.t. v_dist{i in Cerchi, j in Cerchi : i<j} :
  (x[i]-x[j])^2 + (y[i]-y[j])^2 >= (2*r)^2;
```

Esempi di algoritmi di ottimizzazione globale (v)

`cerchi.dat`

```
param N := 3;
```

`cerchi.run`

```
reset;
option randseed 1;
model cerchi.mod;
data cerchi.dat;

option solver minos;
solve;

display x,y,r;
```

Esempi di algoritmi di ottimizzazione globale (vii)

cerchi_multistart.run

```
reset;
option randseed 1;
model cerchi.mod;
data cerchi.dat;
option solver minos;

param r_best;
param x_best{Cerchi};
param y_best{Cerchi};

let r_best := -10000000;
```

Esempi di algoritmi di ottimizzazione globale (vi)

```
for {i in 1..100} {
  reset data x, y, r;    # re-init random
  solve;    # ricerca locale
  display x,y,r;
  if (r > r_best) then {
    printf "Soluzione migliorata\n";
    let r_best := r;
    for {j in Cerchi} {
      let x_best[j] := x[j];
      let y_best[j] := y[j];
    }
  } else {
    printf "Soluzione non migliorata\n";
  }
}
printf "\n================================\n";
printf "Soluzione finale = %12.6f\n\n", r_best;
display x_best, y_best;
```