AMPL: Risoluzione di Problemi Nonlineari Parte 2

F. Rinaldi

Dipartimento di Matematica
Università di Padova

Corso di Laurea Matematica

Outline

AMPL: Risoluzione di Problemi Nonlineari Parte 2

Esempi PNL

Esempio 1: Gestione Ottima di un Portafoglio Titoli

Esempio 1

Abbiamo:

- N titoli con rendimento r aleatorio avente valor medio $\bar{r}$ e covarianza *Cov*.
- Budget B=1.

Determinare un portafoglio ottimo (con varianza minima) al variare del rendimento atteso $r_m = 1.01, \ldots, 1.1$ (con passo 0.01).

Mostrare ad ogni iterazione il portafoglio ottenuto (con titolo e quantitá solo se x_i>0.0001) con valor medio e varianza del rendimento.

Ricorda:

$$\bar{r}_j = \frac{1}{T} \sum_{i=1}^{T} x_{ij}$$

$$Cov(j,k) = \frac{1}{T-1} \sum_{i=1}^{T} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$$

Gestione Ottima di un Portafoglio Titoli

- `asset`: Insieme titoli (n elementi);
- `tempi`: Intervallo temporale (m elementi);
- r_j: Rendimento medio titoli $j \in$ `asset` ;
- Cov_{ij}: Covarianza titoli $i, j \in$ `asset` ;
- B, r_m: Budget, Rendimento minimo atteso;
- x_j: Porzione di capitale investita nel titolo $j \in$ `asset`;

Modello PNL

Otteniamo il seguente problema di PL:

$$\begin{aligned} \min \quad & x^T(Cov)x \\ & \sum_{j=1}^{n} r_j x_j \geq r_m \\ & \sum_{j=1}^{n} x_j \leq B \\ & x_j \geq 0 \quad j = 1, \ldots, n \end{aligned} \tag{1}$$

AMPL: Esempio 1

optrisk.mod

```
#INSIEMI
set A;          # asset
param inizio, fine;
set T := {inizio..fine}; # intervallo temporale

#PARAMETRI
param r_min  ;     # rendimento minimo
param R {T,A};            # rendimenti titoli
#media
param mean {j in A} := ( sum{i in T} R[i,j] )/card(T);
#rendimento normalizzato
param Rtilde {i in T, j in A} := R[i,j] - mean[j];
#matrice di covarianza
param Cov {j in A, k in A}:= sum {i in T} (Rtilde[i,j]*Rtilde[i,k]) / (card(T)-1);
#correlazione
param Corr {j in A, k in A} := Cov[j,k]/sqrt(Cov[j,j]*Cov[k,k]);

#VARIABILI
var x{A} >=0;

#FUNZIONE OBIETTIVO
minimize risk:
     sum{i in A, j in A} Cov[i,j]*x[i]*x[j] ;

#VINCOLI
subject to reward_bound: r_min <= sum{j in A} mean[j]*x[j];
subject to budget: sum{j in A} x[j] = B;
```

AMPL: Esempio 1

optrisk.run

```
reset;
options solver cplex;
model optrisk.mod;
data  optrisk.dat;

printf: "Correlazione\n";
display Corr;
printf: "============\n";

for {r in 1.01 .. 1.1 by 0.01}
{ let r_min := r;
  printf: "Rendimento minimo richiesto: %f\n", r_min;
  solve;
  printf: "------------------------------------------------------------\n";
  printf: "                                   Asset       Media    Varianza \n";
  printf {j in A}: "%45s %10.7f %10.7f \n",
             j, mean[j], sum{i in T} Rtilde[i,j]^2 / card(T);
  printf: "\n";
  printf: "Optimal Portfolio:                 Asset       Fraction \n";
  printf {j in A: x[j] > 0.001}: "%45s %10.7f \n", j, x[j];
  printf: "Mean = %10.7f, Variance = %10.5f \n", sum{j in A} mean[j]*x[j], risk;
  printf: "------------------------------------------------------------\n";
  printf: "------------------------------------------------------------\n";

};
```

Esempio 2: Circle Packing

Esempio 2

Dobbiamo posizionare *n* cerchi nel quadrato unitario in modo che:

- i cerchi non si sovrappongano;
- i cerchi siano all'interno del quadrato;
- il raggio, uguale per tutti, sia il piú grande possibile;

Circle Packing

- `centri`: Insieme centri (n centri);
- (x_j, y_j): coordinate centri ($j \in$ centri);
- r: raggio;

Modello PNL

Otteniamo il seguente problema di PNL:

$$\begin{array}{ll} \max & r \\ & (x_i - x_j)^2 + (y_i - y_j)^2 \geq 4r^2 \quad i,j = 1,\ldots,n; i < j \\ & r \leq x_j \leq 1 - r, \quad j = 1,\ldots,n \\ & r \leq y_j \leq 1 - r, \quad j = 1,\ldots,n \end{array} \tag{2}$$

AMPL: Esempio 2

circle_packing.mod

```
#parametri
param N;

#variabili
var x{j in 1..N} := Uniform(0,1),  >= 0, <=1;
var y{j in 1..N} := Uniform(0,1),  >= 0, <=1;
var r:=Uniform(0,1), >=0, <=1;
;
#funzione obiettivo
maximize raggi: r;

#vincoli
s.t. vinc_distanza{i in 1..N, j in 1..N : i<j}:
                              4*r**2<= (x[i]-x[j])**2 + (y[i]-y[j])**2;

s.t. bounds{i in 1..N}: x[i]>=r;
s.t. bounds2{i in 1..N}: y[i]>=r;
s.t. bounds3{i in 1..N}: x[i]<=1-r;
s.t. bounds4{i in 1..N}: y[i]<=1-r;
```

Risultati ottimi

Raggio massimo al variare di n:

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

Circle Packing: Quesiti

- Quale algoritmo usiamo per risolvere il problema?
- Soluzione ottenuta é ottimale?

Esempio 3

Esempio 3

Implementare l'algoritmo multistart per il caso del Circle Packing.

AMPL: Esempio 3

circle_packing.run

```
reset;
option randseed 0;
model circle_packing.mod;
data circle_packing.dat;
option solver minos;

param sol;
let sol:=-10000000;
for {i in 1..100}
{
# resetta parametri generati randomicamente
reset data x;
reset data y;
reset data r;
#risolve modello e verifica miglioramento soluzione
solve;
display x, y, r;
if (r>sol) then
           {printf "Migliorata Soluzione\n";
            let sol:=r;}
           else
           {printf "Non migliorata Soluzione\n";}
}
printf "=====================================\n";
printf "Soluzione finale= %12.6f\n", sol;
printf "=====================================\n";
```

Circle Packing: Quesiti Finali

- Quale criterio d'arresto utilizziamo?