LINMA1702 modèles et méthodes d'optimisation : Logistique de la vaccination en Belgique

Groupe 6 : Théau Lepouttre, Eliott Van Dieren et Nicolas Mil-Homens Cavaco

2020-2021

0.1 Premier jet

0.1.1 Hypothèses

On considère un certaine population répartie en m classes d'âge dans n provinces en Belgique. Parmi les personnes constituant cette population, on distingue 2 parties disjointes :

- les personnes susceptibles.
- les autres, c'est-à-dire les malades, les vaccinés, les guéris et les morts.

On suppose chaque partie de cette population comme constante dans le temps. Toute personne passant dans la catégorie des autres (en tombant malade ou en étant vacciné) sort du système étudié. Notre objectif est de minimiser le nombre de morts dû à l'épidémie.

Notre seul moyen d'action direct est de vacciner les personnes afin de diminuer la taille de la population des susceptibles.

Sous ces hypothèses, On considère que les 2 objectifs suivant sont équivalents :

- minimiser le nombre de mort
- maximiser la vaccination des personnes les plus susceptibles de tomber malade et de mourir.

0.1.2 Question 1.1

Dans ce premier modèle, on considère le cas d'un seule province ($n = 1$). En outre, le stockage des vaccins n'est pas autorisé. On suppose également que la durée de la campagne de vaccination s'étant sur une durée de T jours.

Soit les variables suivantes

- $x^t \equiv$ le nombre de doses qui arrivent au centre de vaccination au temps t ;
- $y_i^t \equiv$ le nombre de doses administrées à la classe d'âge i au temps t ;

contraintes par les paramètres suivants,

- $b_c^t \equiv$ le nombre de vaccins livrés dans l'entrepôt central ;
- $b_l^t \equiv$ le nombre maximal de vaccins livrés au temps t ;
- $b_v^t \equiv$ le nombre maximal de vaccins administrés au temps t ;
- $c_{tr} \equiv$ le coût de transport ;
- $c_v \equiv$ le coût de vaccination ;
- $c_{tot} \equiv$ le budget total disponible ;
- $(n_s)_i^t \equiv$ le nombre de personnes susceptibles parmi les personnes de la classe i au temps t ;
- $(n_{nv})_i^t \equiv$ le nombre de personnes qui ne veulent pas se faire vacciner parmi les suceptibles dans la classe i au temps t.

Soit

- $\lambda_i^t \equiv$ la proportion de malades parmi les personnes de la classe i au temps t.
- $\varepsilon_i^t \equiv$ la proportion de morts parmi les malades de la classe i au temps t.
- $\mu_i \equiv$ la proportion de personnes disposées et autorisées à se faire vacciner.

On identifie les contraintes suivantes :

- Le budget total de la campagne de vaccination est limité.

- Le total de vaccins administrés au temps t ne peut pas dépasser le nombre total de vaccins livrés la veille, et ce sur toute la durée de la campagne.
- Le nombre de vaccins livrés est limité chaque jour par le nombre de vaccins disponibles dans l'entrepôt central.
- Le nombre de vaccins livrés est limité chaque jour par la limite du centre de vaccination.
- Le nombre de vaccins administrés est limité chaque jour.
- Les non-consentants ne peuvent pas être vaccinés.

Finalement, on prend la convention que $x^0 = 0 = x^T$, en d'autre terme aucun vaccin n'a été livré la veille du début de la campagne ou ne doit être livré le dernier jour.

Le problème s'écrit donc

$$
\begin{aligned}
\max_{x,y} \quad & \textstyle\sum_{t=1}^{T} \sum_{i=1}^{m} \varepsilon_i^t \lambda_i^t \ (n_s)_i^t \ y_i^t \\
\sum_{t=1}^{T} \left(c_{tr} \ x^t + c_v \sum_{i=1}^{m} y_i^t \right) & \leq & c_{tot} \\
x^{t-1} - \sum_{i=1}^{m} y_i^t & \geq & 0 \\
x^t & \leq & b_c^t \\
x^t & \leq & b_l^t \\
\sum_{i=1}^{m} y_i^t & \leq & b_v^t \\
y_i^t & \leq & \mu_i \ (n_s)_i \\
x, \ y & \geq & 0
\end{aligned}
$$

```python
import numpy as np
import matplotlib.pyplot as plt
from mip import *


# Donnees du probleme

T = 350                 # duree de la campagne
m = 5                   # nombre de classes d'age : [Young, Adult, Senior, Old, Centenarian]
c_tr  = 0               # Prix de livraison d'un vaccin
c_v   = 15              # prix d'administration d'un vaccin
c_tot = 100e6           # budget total autorise
b_l   = float("inf")    # nombre maximal de vaccins livres par jour
b_v   = 14646           # nombre maximal de vaccins administres par jour

# nombre de personnes susceptibles dans chaque classe d'age
n_s   = [3778123, 2846993, 2790883, 1390502, 111533]

lambda_t = lambda t: np.array([0.000298 * (1/5 + np.sin(t/50-1/5)**2),
                               0.000301 * (1/5 + np.sin(t/50)**2),
                               0.000204 * (1/5 + np.sin(t/50-1/5)**2),
                               0.000209 * (1/5 + np.sin(t/50-2/5)**2),
                               0.000329 * (1/5 + np.sin(t/50-2/5)**2)])

epsilon_t = lambda t: np.array([0.000100 * (6/5-t/1000),
                                0.000400 * (6/5-t/1000),
                                0.005790 * (6/5-t/1000),
                                0.027179 * (6/5-t/1000),
                                0.150000 * (6/5-t/1000)])

mu  = np.array([0.3, 0.6, 0.7, 0.9, 0.9])     # proportion de la population disposee a se faire vacciner.

b_c = np.zeros(T);
days = np.arange(1, T+1)
b_c_eff =  [80000, 80000,60000, 60000, 40000, 40000, 40000,
            40000, 60000,60000,60000,60000,80000,80000,80000,
            80000,100000,100000,100000,100000,100000,100000,
            100000,100000,120000,120000,120000,120000,120000,
            120000,120000,120000,120000,120000,120000,120000,
            150000,150000,150000,150000,150000,150000,150000,
            150000,150000,150000,150000,150000,150000,150000]

for t in days-1:
    if(t % 7 == 0):
        b_c[t] = b_c_eff[t//7]

_lambda  = lambda_t(days)
_epsilon = epsilon_t(days)


# Resolution du probleme
model = Model('centre unique', sense=MAXIMIZE, solver_name=CBC)

# Variables
x = np.array( [model.add_var(var_type=INTEGER, lb=0) for t in days-1])
y = np.array([[model.add_var(var_type=INTEGER, lb=0) for t in days-1] for i in range(m)])

# Objectif
model.objective = maximize(xsum(_epsilon[i,t] *_lambda[i,t] * n_s[i] * y[i,t] for i in range(m) for t in days-1))

# Contraintes
model += xsum(c_tr * x) + xsum(c_v * y[i,t] for i in range(m) for t in days-1) <= c_tot
for t in days-1:
    if t > 1:
```

```
        model += x[t-1] - xsum(y[:,t]) >= 0
    model += x[t] <= b_c[t]
    model += xsum(y[:,t]) <= b_v
    for i in range(m):
        model += y[i,t] <= mu[i] * n_s[i]

model.optimize()

# print(f"x = {[x[t].x for t in days-1]}")
# print("\n")
# print(f"y = {[[y[i,t].x for i in range(m)] for t in days-1]}")
print(f"f(x,y) = {model.objective_value}")
```

0.1.3 Question 1.2

On considère à présent le cas de n provinces. Dans le cas de la Belgique, on aura $n = 10$. On autorise également le stockage des vaccins dans les centres.

Par rapport au modèle précédent, on introduit

- $z_j^t \equiv$ le nombre de vaccins stockés au temps t dans le centre de la province j, variable. Ces vaccins ne sont donc pas administrés au jour t.
- $(c_s)_j \equiv$ le coût associé au stockage d'un vaccin dans le centre j, connu.

Outre les contraintes identifiées plus haut, on ajoute le fait que le nombre de vaccins administrés au temps t ne peut déplacer le nombre de vaccins livrés la veille et ceux en stocks la veille également.

Finalement, puisqu'on suppose que chaque province j subit l'épidémie de manière indépendante, les contraintes sont propres à chacune, excepté la contrainte de coût total.

Le problème s'écrit donc

$$
\begin{aligned}
\max_{x,y,z} \quad & \textstyle\sum_{t=1}^{T} \sum_{i=1}^{m} \sum_{j=1}^{n} \varepsilon_{ij}^t \lambda_{ij}^t \ (n_s)_{ij}^t \ y_{ij}^t & & \\
\sum_{t=1}^{T} \sum_{j=1}^{n} \left(c_{tr} \ x_j^t + c_v \sum_{i=1}^{m} y_{ij}^t + c_s z_j^t \right) & & \leq & \quad c_{tot} \\
x_j^{t-1} + z_j^{t-1} - \sum_{i=1}^{m} y_{ij}^t & & \geq & \quad 0 \\
\sum_{j=1}^{n} x_j^t & & \leq & \quad b_c^t \\
x_j^t & & \leq & \quad (b_l)_j^t \\
\sum_{i=1}^{m} y_{ij}^t & & \leq & \quad (b_v)_j^t \\
y_{ij}^t & & \leq & \quad (n_s)_{ij}^t - (n_{nv})_{ij}^t \\
x, \ y, z & & \geq & \quad 0
\end{aligned}
$$