Extraction d'eau

```
%pip install gurobipy
from gurobipy import Model, GRB, LinExpr, multidict
import random
import matplotlib.pyplot as plt
```

Données du problème

```
random.seed(1)

nplaces = 10                                              # nombre d'emplacements potentiels
cost = [random.randint(5,25) for _ in range(nplaces)]     # cout d'investissement
flow = [random.randint(100,300) for _ in range(nplaces)]  # debit moyen
flowmin = 500
flowmax = 600
nmax = 3
```

Modèle mathématique

$$
\begin{aligned}
& min \sum_j cost_j * x_j \\
& s.t.: Fmin \leq \sum_j flow_j * x_j \leq Fmax \\
& \sum_j x_j \leq Nmax \\
& x_1 + x_2 \leq 1 \\
& x_j \in \{0,1\}, \forall j
\end{aligned}
$$

```
m = Model('extraxtion')
```

Variables de décisions

```
x = m.addVars(nplaces, vtype=GRB.BINARY, name="X")  # emplacement utilisé ?
m.update()
costx = LinExpr(cost, x.values())  # cout d'investissement
flowx = LinExpr(flow, x.values())  # debit total
nx = x.sum()                       # nb pompes
```

Contraintes et objectif

```
ctflowmin = m.addConstr(flowx >= flowmin)
ctflowmax = m.addConstr(flowx <= flowmax)
ctnmax = m.addConstr(flowx <= flowmax)
ctexclusion = m.addConstr(x[1] + x[2] <= 1)

m.setObjective(costx, GRB.MINIMIZE)
```

Optimisation

```
m.optimize()
```

```
m.display()
m.write("myfirstmip.lp")
```

Résultats

```
m.printStats()
```

```
# caractéristiques de la solution
result = {'obj': m.objVal, 'lb': m.objBound, 'time': m.runtime}
print(result)
```

```
# installation optimale
installed = [j for j in range(nplaces) if x[j].x == 1]
print(f"emplacements a installer: {installed}")

costval = costx.getValue()
flowval = flowx.getValue()
print(f"cout = {costval}, debit = {flowval}")
```