Chapter 4:

Adjustable Robust Linear Programming

Why we need adjustable robust models?

Why worry about decision sequences?

- Consider a simple inventory management problem :

$$\underset{x,y}{\text{minimize}} \sum_{t=1}^{T} \left(\overbrace{c_t x_t}^{\text{ordering cost}} + \overbrace{h_t (y_{t+1})^+}^{\text{holding cost}} + \overbrace{b_t (-y_{t+1})^+}^{\text{backlog cost}} \right)$$

$$\begin{aligned} \text{s.t.} \quad & y_{t+1} = y_t + x_t - d_t, \ \forall\, t, && \text{(Stock balance)} \\ & 0 \leqslant x_t \leqslant M_t, \ \forall\, t, && \text{(Min/max order size)} \\ & y_1 = a\ , && \text{(Initial stock level)} \end{aligned}$$

x_t : the amount ordered for time t
y_t : the amount in inventory at beginning of t
d_t : the demand at time t
a : the initial inventory

A linear programming reformulation

- This is the linearized version of the inventory model:

$$
\begin{aligned}
\underset{x,y,s^+,s^-}{\text{minimize}} \quad & \sum_{t=1}^{T} \left(c_t x_t + h_t s_t^+ + b_t s_t^-\right) \\
\text{s.t.} \quad & s_t^+ \geqslant 0,\ s_t^- \geqslant 0\,,\ \forall\, t, \\
& s_t^+ \geqslant y_{t+1}\,,\ \forall\, t, \\
& s_t^- \geqslant -y_{t+1}\,,\ \forall\, t, \\
& y_{t+1} = y_t + x_t - d_t\,,\ \forall\, t, \\
& 0 \leqslant x_t \leqslant M_t\,,\ \forall\, t,
\end{aligned}
$$

How can we make this model robust to demand perturbations?

Naïve robustification

- Given that the vector of demand d is assumed to lie in some uncertainty set U, let's consider the robust optimization model:

$$
\begin{aligned}
\underset{x,y,s^+,s^-}{\text{minimize}} \quad & \sum_{t=1}^{T} \left(c_t x_t + h_t s_t^+ + b_t s_t^- \right) \\
\text{s.t.} \quad & s_t^+ \geqslant 0,\ s_t^- \geqslant 0\,,\ \forall\, t \\
& s_t^+ \geqslant y_{t+1}\,,\ \forall\, t \\
& s_t^- \geqslant -y_{t+1}\,,\ \forall\, t \\
& y_{t+1} = y_t + x_t - d_t\,,\ \forall\, d \in \mathcal{U}\,,\ \forall\, t \\
& 0 \leqslant x_t \leqslant M_t\,,\ \forall\, t
\end{aligned}
$$

- Unfortunately, this makes the model infeasible even when |U| = 2:

$$
\left\{ \begin{array}{l} y_{t+1} = y_t + x_t - d_t^{(1)} \\ y_{t+1} = y_t + x_t - d_t^{(2)} \end{array} \right\} \Rightarrow d_t^{(1)} = d_t^{(2)}
$$

A less naïve robustification

- Robustify an alternate linear programming reformulation:

$$
\begin{aligned}
\underset{x,s^+,s^-}{\text{minimize}} \quad & \sum_{t=1}^{T} \left(c_t x_t + h_t s_t^+ + b_t s_t^-\right) \\
\text{s.t.} \quad & s_t^+ \geqslant 0,\ s_t^- \geqslant 0,\ \forall t, \\
& s_t^+ \geqslant y_1 + \sum_{t'=1}^{t} x_{t'} - d_{t'},\ \forall t, \\
& s_t^- \geqslant -y_1 + \sum_{t'=1}^{t} d_{t'} - x_{t'},\ \forall t, \\
& 0 \leqslant x_t \leqslant M_t\ \forall t\ ,
\end{aligned}
$$

where we simply replaced $y_{t+1} := y_1 + \sum_{t'=1}^{t} x_{t'} - d_{t'}$ in order to capture the fact that stock level evolves according to demand.

A less naïve robustification

- Robustify an alternate linear programming reformulation:

$$
\begin{aligned}
\underset{x,s^+,s^-}{\text{minimize}} \quad & \sum_{t=1}^{T} \left(c_t x_t + h_t s_t^+ + b_t s_t^- \right) \\
\text{s.t.} \quad & s_t^+ \geqslant 0,\ s_t^- \geqslant 0,\ \forall t, \\
& s_t^+ \geqslant y_1 + \sum_{t'=1}^{t} x_{t'} - d_{t'},\ \forall\, d \in \mathcal{U},\ \forall\, t, \\
& s_t^- \geqslant -y_1 + \sum_{t'=1}^{t} d_{t'} - x_{t'},\ \forall\, d \in \mathcal{U},\ \forall\, t, \\
& 0 \leqslant x_t \leqslant M_t, \forall\, t\ .
\end{aligned}
$$

Still two issues remain:

1. the orders should be adjustable w.r.t. the observed demand
2. (s_t^+, s_t^-) should be fully adjustable (more subtle)

Why (s+,s-) should be fully adjustable

- Consider the two-stage problem with $d_1 \in [0,\ 2]$:

<u>Deterministic model:</u>

$$\min_{x_1} \quad \overbrace{0.5x_1}^{\text{ordering}} + \overbrace{(x_1 - d_1)^+}^{\text{holding}} + \underbrace{(d_1 - x_1)^+}_{\text{backlog}}$$

$$\text{s.t.} \quad 0 \leqslant x_1 \leqslant 2\ ,$$

<u>Less naïve model:</u>

$$\begin{aligned} \min_{x_1, s_1^+, s_1^-} \quad & 0.5x_1 + s_1^+ + s_1^- \\ \text{s.t.} \quad & s_1^+ \geq 0\ ,\ s_1^- \geq 0 \\ & s_1^+ \geq x_1 - d_1\ ,\ \forall\, d_1 \in [0,\ 2] \\ & s_1^- \geq d_1 - x_1\ ,\ \forall\, d_1 \in [0,\ 2] \\ & 0 \leq x_1 \leq 2 \end{aligned}$$

Why (s^+,s^-) should be fully adjustable

- Consider the two-stage problem with $d_1 \in [0,\, 2]$:

Deterministic model:

$$\min_{x_1} \quad \overbrace{0.5x_1}^{\text{ordering}} + \overbrace{(x_1 - d_1)^+}^{\text{holding}} + \underbrace{(d_1 - x_1)^+}_{\text{backlog}}$$

$$\text{s.t.} \quad 0 \leqslant x_1 \leqslant 2 \,,$$

Less naïve model:

$$\min_{x_1} \quad 0.5x_1 + \underbrace{x_1}_{s_1^{+*}} + \underbrace{2 - x_1}_{s_1^{-*}}$$

$$\text{s.t.} \quad 0 \leqslant x_1 \leqslant 2 \,.$$

Conclusions:

- Less naïve robust model states $x_1^* := 0,\ s_1^{+*} := 0,$ and $s_1^{-*} := 2$ with worst-case of 2
- Alternatively, x_1=1 achieves a total cost lower than 1.5 for all d_1 in [0,2]

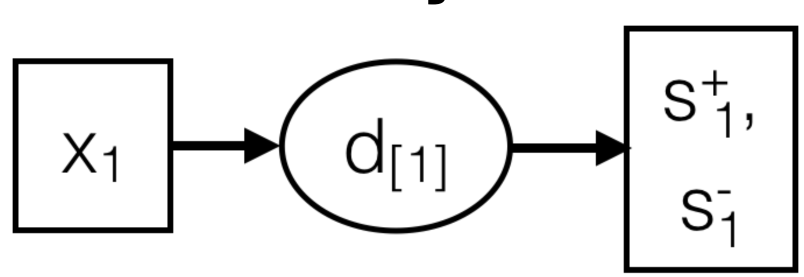
An accurate two-stage robust inventory model

- The robust two-stage problem actually takes the form:

$$\begin{aligned} \underset{x_1}{\text{minimize}} \quad & \sup_{d_1 \in [0,2]} \; 0.5x_1 + h(x_1, d_1) \\ \text{s.t.} \quad & 0 \leqslant x_1 \leqslant 2 \,, \end{aligned}$$

where $$h(x_1, d_1) := \begin{aligned} \min_{s_1^+, s_1^-} \quad & s_1^+ + s_1^- \\ \text{s.t.} \quad & s_1^+ \geqslant 0,\; s_1^- \geqslant 0 \\ & s_1^+ \geqslant x_1 - d_1 \\ & s_1^- \geqslant -x_1 + d_1 \,. \end{aligned}$$

Timing characterized by accurate robust model



Alternate representation of less naïve robust model

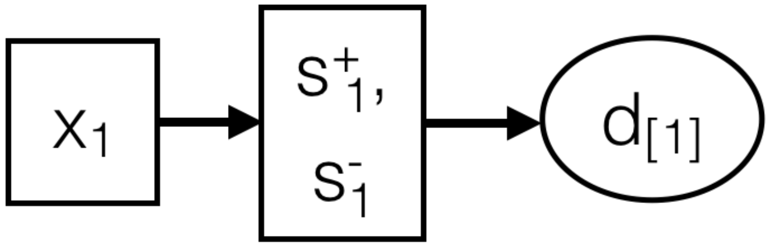
- Comparatively, the less naïve robust model was solving:

$$\begin{array}{lll} \underset{x_1, s_1^+, s_1^-}{\text{minimize}} & \underset{d_1 \in [0,2]}{\sup} & 0.5x_1 + g(x_1, s_1^+, s_1^-, d_1) \\ \text{s.t.} & & s_1^+ \geqslant 0,\ s_1^- \geqslant 0 \\ & & 0 \leqslant x_1 \leqslant 2\ , \end{array}$$

where

$$g(x_1, s_1^+, s_1^-, d_1) := \begin{cases} s_1^+ + s_1^- & \text{if } s_1^+ \geqslant x_1 - d_1 \text{ and } s_1^- \geqslant d_1 - x_1 \\ \infty & \text{otherwise} \end{cases} .$$

Timing characterized by less naïve robust model



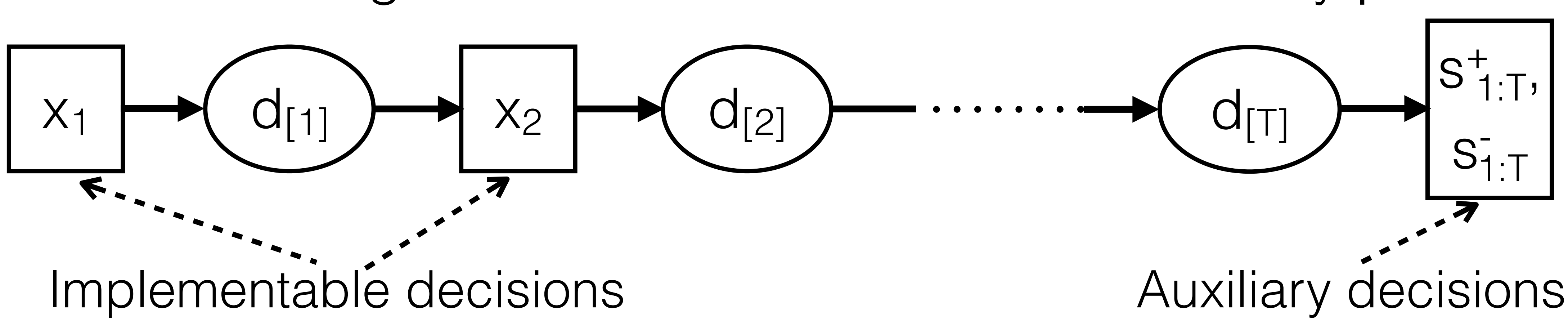
Takeaway message about adjustable decisions

When robustifying decision models that either involve

- "implementable" decisions at different time periods
- "auxiliary" decisions such as s^+, s^- that are used to assess overall performance of implemented decisions

one needs to carefully identify the chronology of decisions and observations and employ the <u>adjustable robust counterpart</u> framework introduced in (Ben-Tal et al., 2004)

Accurate timing of decisions and observations in inventory problem



Two examples in the literature

Operations Research
Vol. 54, No. 1, January–February 2006, pp. 150–168
ISSN 0030-364X | EISSN 1526-5463 | 06 | 5401 | 0150
informs®
DOI 10.1287/opre.1050.0238

A Robust Optimization Approach to Inventory Theory

$$\text{minimize} \quad \sum_{k=0}^{T-1} (cu_k + Kv_k + y_k) \tag{9}$$

subject to

$$y_k \geqslant h\left(x_0 + \sum_{i=0}^{k}(u_i - w_i)\right), \quad k = 0, \ldots, T-1, \tag{10}$$

$$y_k \geqslant -p\left(x_0 + \sum_{i=0}^{k}(u_i - w_i)\right), \quad k = 0, \ldots, T-1, \tag{11}$$

$$0 \leqslant u_k \leqslant Mv_k, \quad v_k \in \{0, 1\}, \quad k = 0, \ldots, T-1, \tag{12}$$

where $w_i = \overline{w}_i + \widehat{w}_i \cdot z_i$ such that $\mathbf{z} \in \mathcal{P} = \{|z_i| \leqslant 1 \ \forall i \geqslant 0, \sum_{i=0}^{k} |z_i| \leqslant \Gamma_k \ \forall k \geqslant 0\}$.

Following the technique developed in §2, the robust approach consists here of maximizing the right-hand side of the constraints over the set of admissible scaled deviations.

PRODUCTION AND OPERATIONS MANAGEMENT POMS
Vol. 20, No. 5, September–October 2011, pp. 772–785
ISSN 1059-1478 | EISSN 1937-5956 | 11 | 2005 | 0772
DOI 10.3401/poms.1080.01194

Facility Location: A Robust Optimization Approach

$$(P') \quad \max_{\mathbf{X},\mathbf{Z},\mathbf{I},\mathbf{Z}_0,\tau} \tau,$$

s.t.

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{t=1}^{T}(\eta - d_{ij})D_{jt}X_{ijt} - \sum_{i=1}^{N}\sum_{t=1}^{T}c_{it}Z_{it} - \sum_{i=1}^{N}(C_{i0}Z_{i0} + K_iI_i) \geq \tau, \tag{7a}$$

$$\sum_{j=1}^{N} D_{jt}X_{ijt} \leq Z_{it} \quad \text{for all } i, t, \tag{7b}$$

$$\sum_{i=1}^{N} X_{ijt} \leq 1 \quad \text{for all } j, t, \tag{7c}$$

$$Z_{i0} \leq MI_i \quad \text{for all } i, \tag{7d}$$

$$Z_{it} \leq Z_{i0} \quad \text{for all } i, t, \tag{7e}$$

$$X_{ijt} \geq 0; I_i \in \{0, 1\} \quad \text{for all } i, j, t. \tag{7f}$$

We now transform problem (P′) into a new problem expressing uncertainty by substituting $\tilde{D}_{jt}$ for D_{jt} in constraints (7a) and (7b) and then augmenting it with the constraint $\tilde{D}_{jt} \in U^B$ for all $j \in N$ and $t = 1, 2, \ldots, T$.

The augmented constraint for (7a) is

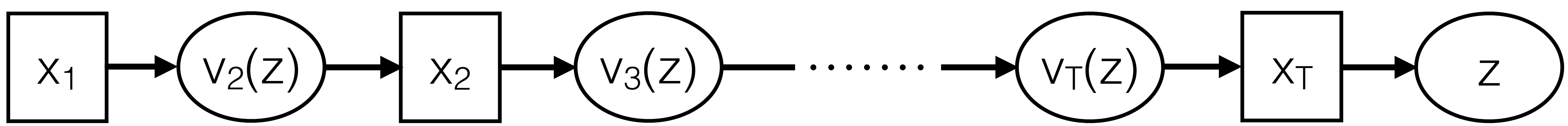
$$\min_{\tilde{D}_{jt} \in U^B} \left\{ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{t=1}^{T} \tilde{D}_{jt}(\eta - d_{ij})X_{ijt} \right\} - \sum_{i=1}^{N}\sum_{t=1}^{T} c_{it}Z_{it} - \sum_{i=1}^{N}(C_{i0}Z_{i0} + K_iI_i) \geq \tau. \tag{8}$$

Next, consider constraint (7b) for a given i and t.

$$\max_{\tilde{D}_{jt} \in U^B} \left\{ \sum_{j=1}^{N} \tilde{D}_{jt} X_{ijt} \right\} \leq Z_{it}.$$

Identifying the chronology of execution and observation

- Here is how we will identify the chronology:



x_t : the decision implemented at time t
z : the underlying uncertainty about the whole future
$v_t(z)$: function that returns what was observed of z at time t (i.e. the « visual evidence » at time t)

Adjustable Robust Linear Programming

- Nominal dynamic problem:

$$\begin{aligned} \underset{\{x_t\}_{t=1}^T}{\text{maximize}} \quad & \sum_{t=1}^{T} c_t^T x_t + d \\ \text{subject to} \quad & \sum_{t=1}^{T} a_{jt}^T x_t \leq b_j \,,\, \forall\, j = 1, \ldots, J, \end{aligned}$$

- Multi-stage Adjustable Robust Counterpart:

$$\underset{x_1, \{x_t(\cdot)\}_{t=2}^T}{\text{maximize}} \quad \inf_{z \in \mathcal{Z}} \; c_1(z)^T x_1 + \sum_{t=2}^{T} c_t(z)^T x_t(v_t(z)) + d(z) \tag{4.8a}$$

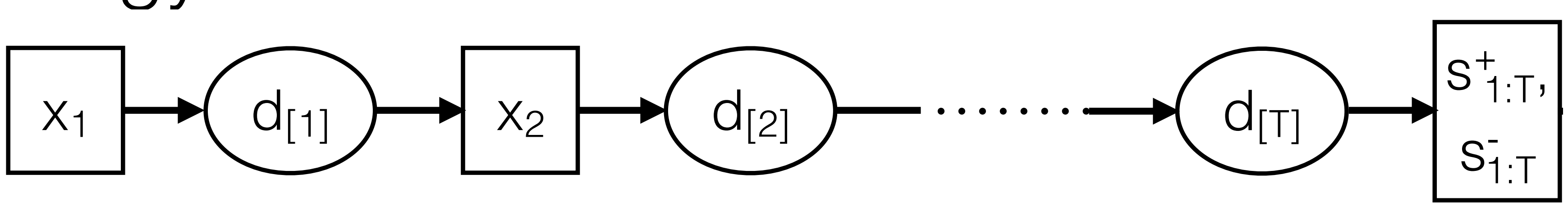
$$\text{subject to} \quad a_{j1}(z)^T x_1 + \sum_{t=2}^{T} a_{jt}(z)^T x_t(v_t(z)) \leq b_j(z) \,,\, \forall\, z \in \mathcal{Z} \,,\, \forall\, j = 1, \ldots, J, \tag{4.8b}$$

Multi-Stage ARC for Inventory Problem

- Nominal problem:

$$\begin{aligned} \underset{x_t, s_t^+, s_t^-}{\text{minimize}} \quad & \sum_t c_t x_t + h_t s_t^+ + b_t s_t^- \\ \text{subject to} \quad & s_t^+ \geq 0,\ s_t^- \geq 0 \\ & s_t^+ \geq y_1 + \sum_{t'=1}^{t} x_{t'} - d_{t'} \\ & s_t^- \geq -y_1 + \sum_{t'=1}^{t} d_{t'} - x_{t'} \\ & 0 \leq x_t \leq M \ , \end{aligned}$$

- Chronology:



- Progressively revealed uncertainty:

$$v_t(d) = d_{[t-1]} := [\ \boldsymbol{I}_{t-1} \quad \mathbf{0}_{t,T-t+1}\ ]d = [\ d_1 \quad d_2 \quad \cdots \quad d_{t-1}\ ]^T$$

Multi-Stage ARC for Inventory Problem

$$
\begin{aligned}
\underset{x_1,\{x_t(\cdot)\}_{t=2}^{T},\{s_t^+(\cdot),s_t^-(\cdot)\}_{t=1}^{T}}{\text{minimize}} \quad & \sup_{d\in\mathcal{U}} \; c_1 x_1 + \sum_t c_t x_t(d_{[t-1]}) + h_t s_t^+(d) + b_t s_t^-(d) \\
\text{subject to} \quad & s_t^+(d) \geq 0,\; s_t^-(d) \geq 0\,,\; \forall\, d \in \mathcal{U},\; \forall\, t \\
& s_t^+(d) \geq y_1 + \sum_{t'=1}^{t} x_{t'}(d_{[t'-1]}) - d_{t'}\,,\; \forall\, d \in \mathcal{U},\; \forall\, t \\
& s_t^-(d) \geq -y_1 + \sum_{t'=1}^{t} d_{t'} - x_{t'}(d_{[t'-1]})\,,\; \forall\, d \in \mathcal{U},\; \forall\, t \\
& 0 \leq x_t(d_{[t'-1]}) \leq M\,,\; \forall\, d \in \mathcal{U},\; \forall\, t\,,
\end{aligned}
$$

Solution methods for two-stage problems

Difficulty of resolution of Multi-stage ARC

- Theorem 4.2: Solving the multi-stage ARC model is NP-hard even for a two-stage problem with polyhedral uncertainty.

- To prove this result, we can show that it can be used to answer the NP-complete 3-SAT problem:

Exact solution methods for Two-stage ARC

- Some exact algorithms have been proposed for the two-stage ARC problem.

- Hypothesis:

  - « fixed » recourse

(TSARC)

$$\begin{array}{lll} \underset{x,y(\cdot)}{\text{maximize}} & \inf_{z\in\mathcal{Z}} c_1(z)^T x_1 + c_2^T y(z) + d(z) \\ \text{subject to} & a_{j1}(z)^T x + a_{j2}^T y(z) \leq b_j(z)\,,\ \forall\, z \in \mathcal{Z}\,,\ \forall\, j = 1,\ldots,J \\ & x \in \mathcal{X}\,, \end{array} \tag{4.12}$$

  - « relatively complete » recourse

$$\mathcal{X} \subseteq \{x \in \mathbb{R}^n \,|\, \forall\, z \in \mathcal{Z}, \exists y \in \mathbb{R}^n, a_{j1}(z)^T x + a_{j2}^T y \leq b_j(z)\,,\ \forall\, j = 1,\ldots,J\}$$

Vertex enumeration

Theorem 4.6. : *Assume that the uncertainty set $\mathcal{Z}$ is given as the convex hull of a finite set:*

$$\mathcal{Z} := ConvexHull(\{\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_K\}) \ .$$

Then, the TSARC presented in problem (4.12) *is equivalent to*

$$\begin{aligned}
&\underset{x,\{y_k\}_{k=1}^K}{\text{maximize}} && \min_k \ c_1(\bar{z}_k)^T x_1 + c_2^T y_k + d(\bar{z}_k) && (4.13a)\\
&\text{subject to} && a_{j1}(\bar{z}_k)^T x_1 + a_{j2}^T y_k \le b_j(\bar{z}_k)\,,\ \forall\, k = 1,\ldots,K,\ \forall\, j = 1,\ldots,J && (4.13b)\\
& && x \in \mathcal{X}\ . && (4.13c)
\end{aligned}$$

Lemma 4.7. : *Assume that the uncertainty set $\mathcal{Z}$ is given as the convex hull of a finite set :*

$$\mathcal{Z} := ConvexHull(\{\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_K\}) \ .$$

Then, given a concave function $h(z)$ over $\mathcal{Z}$, the optimal value of $\min_{z\in\mathcal{Z}} h(z)$ is equal to $\min_{k\in\{1,2,\ldots,K\}} h(\bar{z}_k)$.

Difficulty of vertex enumeration

- A polyhedron described by *m* linear constraints, can have up to $2^{m/2}$ vertices, e.g. the box uncertainty set.

- The next algorithm will use a column & constraint generation scheme to try to find a small subset of vertices needed to identify the robust first-stage decision.

- The hope is that for an *n*-dimensional x_1, we can work with *n* vertices

Performance of decomposition schemes

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

Solving two-stage robust optimization problems using a column-and-constraint generation method

Bo Zeng*, Long Zhao

Table 3
Performance of Benders-dual and C&CG algorithms on 70 × 70 instances.

Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Avg.
BD (CPU sec.)	776.42	1580.71	1367.34	1300.44	1002.96	935.42	672.68	735.81	619.7	466.68	945.82
C&CG (CPU sec.)	26.16	21.27	72.3	65.22	37.88	54.62	16.72	17.64	9.66	1.55	32.3
Ratio	**29.68**	**74.32**	**18.91**	**19.94**	**26.48**	**17.13**	**40.23**	**41.71**	**64.15**	**301.08**	**63.36**
BD (# iter.)	203.9	152.1	117.5	127.1	137.4	143.6	126.3	134.2	136.6	132.4	141.11
C&CG (# iter.)	6.8	5	4.9	5	5.2	5.9	4.5	5.1	4.9	2	4.93
Ratio	**29.99**	**30.42**	**23.98**	**25.42**	**26.42**	**24.34**	**28.07**	**26.31**	**27.88**	**66.20**	**30.90**
BD Master (sec./iter.)	1.13	0.79	0.57	0.56	0.46	0.41	0.34	0.35	0.33	0.3	0.52
C&CG Master (sec./iter.)	1.45	0.58	0.57	0.58	0.55	0.72	0.47	0.5	0.51	0.12	0.61
Ratio	**0.78**	**1.36**	**1.00**	**0.97**	**0.84**	**0.57**	**0.72**	**0.70**	**0.65**	**2.50**	**1.01**

Column & constraint generation algorithm

- Let $\mathcal{Z}'_v := \{\bar{z}'_1, \bar{z}'_2, \ldots, \bar{z}'_{K'}\}$ be a subset of vertices of $\mathcal{Z}$, and solve:

$$\underset{x,s,\{y_k\}_{k=1}^{K'}}{\text{maximize}} \quad s \tag{4.14a}$$

Optimal solution $= (\hat{s}, \hat{x}, \hat{y}_k)$

$$\text{subject to} \quad s \leq c_1(\bar{z}'_k)^T x + c_2^T y_k + d(\bar{z}'_k)\,,\ \forall\, k = 1, \ldots, K' \tag{4.14b}$$

$$a_{j1}(\bar{z}'_k)^T x_1 + a_{j2}^T y_k \leq b_j(\bar{z}'_k)\,,\ \forall\, k = 1, \ldots, K', \forall\, j = 1, \ldots, J \tag{4.14c}$$

$$x \in \mathcal{X}\ , \tag{4.14d}$$

1. The approximate value ŝ always provides an upper bound to true optimal worst-case value

2. If ŝ is exactly equal to « $\min_z$ h($\hat{x}$,z) », then $\hat{x}$ is exactly optimal, where

$$h(x,z) := \max_{y} \quad c_1(z)^T x + c_2^T y + d(z)$$

$$\text{subject to} \quad a_{j1}(z)^T x + a_{j2}^T y \leq b_j(z)\,,\ \forall\, j = 1, \ldots, J.$$

3. If ŝ is strictly greater than « $\min_z$ h($\hat{x}$,z) », then there exists a new vertex ẑ of $\mathcal{Z}$ for which h($\hat{x}$, ẑ) < ŝ

Column & constraint generation algorithm

- Based on this idea, on can design the following procedure:

1. Take any $\hat{x} \in \mathcal{X}$
2. Identify $\hat{z} := \text{argmin}_{z \in \mathcal{Z}} h(\hat{x}, z)$ and construct $\mathcal{Z}'_v := \{\hat{z}\}$
3. Iterate until algorithm converged:
   (a) Solve problem (4.14) to obtain $\hat{x}$ and $\hat{s}$
   (b) Identify $\hat{z} := \text{argmin}_{z \in \mathcal{Z}} h(\hat{x}, z)$, if $h(\hat{x}, \hat{z}) = \hat{s}$ then the algorithm converged, otherwise add $\hat{z}$ to $\mathcal{Z}'_v$ and iterate

- The procedure will converge in finite amount of time since all polyhedra have a finite number of vertices

Identifying worst-case z vertex for $\hat{x}$

- The difficulty now relies upon solving the following NP-hard problem:

$$\min_{z \in \mathcal{Z}} h(x, z)$$

$$h(x, z) := \max_{y} \quad c_1(z)^T x + c_2^T y + d(z)$$

$$\text{subject to} \quad a_{j1}(z)^T x + a_{j2}^T y \leq b_j(z)\,, \forall\, j = 1, \ldots, J.$$

- One way of doing so is by solving the following MILP:

$$\min_{z \in \mathcal{Z}} h(x, z) := \min_{z \in \mathcal{Z}, y, \lambda, u} c_1(z)^T x + c_2^T y + d(z)$$

$$a_{j1}(z)^T x + a_{j2}^T y \leq b_j(z)\,, \forall\, j = 1, \ldots, J$$

$$\lambda \geq 0$$

$$\lambda_j \leq M u_j\,, \forall\, j = 1, \ldots, J$$

$$b_j(z) - a_{j1}(z)^T x - a_{j2}^T y \leq M(1 - u_j)\,, \forall\, j = 1, \ldots, J$$

$$c_2 = \sum_j a_{j2} \lambda_j \quad u \in \{0, 1\}^J\,,$$

KKT optimality conditions are behind the MILP reformulation

Corollary 4.10. : *Given a linear programming problem*

$$\begin{aligned} \underset{y}{\text{maximize}} \quad & c^T y \\ \text{subject to} \quad & Ay \le b\,, \end{aligned}$$

where $y \in \mathbb{R}^n$. If this optimization problem satisfies strong duality, then any primal dual optimal solution pair must satisfy the following conditions

Primal feasibility → $A\tilde{y} \le b$

$\tilde{\lambda} \ge 0$

Complementary slackness → $\tilde{\lambda}_j (a_j^T \tilde{y} - b_j) = 0\,,\ \forall\, j = 1, 2, \ldots, J$

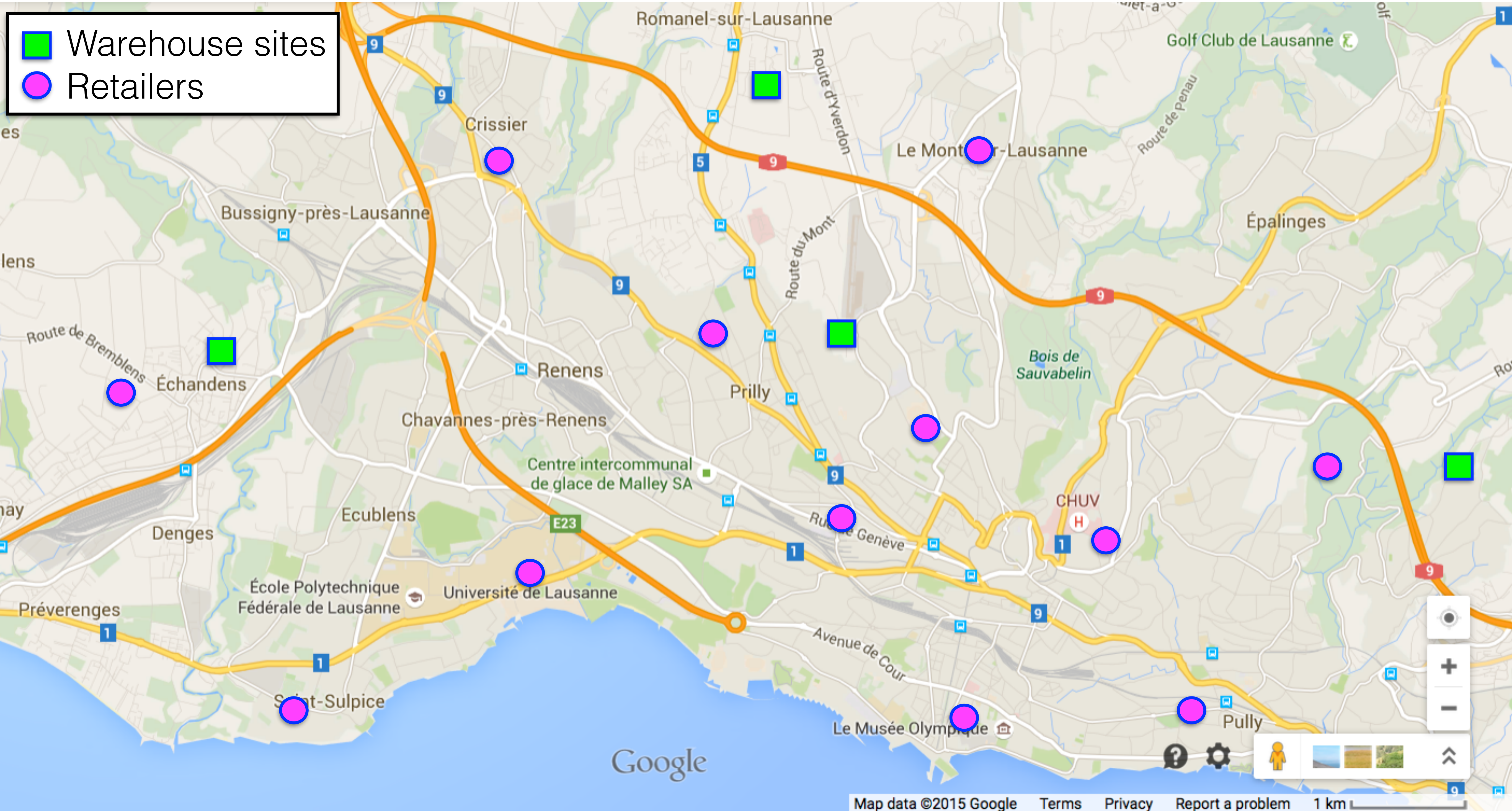
$c = \sum_j \tilde{\lambda}_j a_j$

Dual feasibility (→ $\tilde{\lambda} \ge 0$ and $c = \sum_j \tilde{\lambda}_j a_j$)

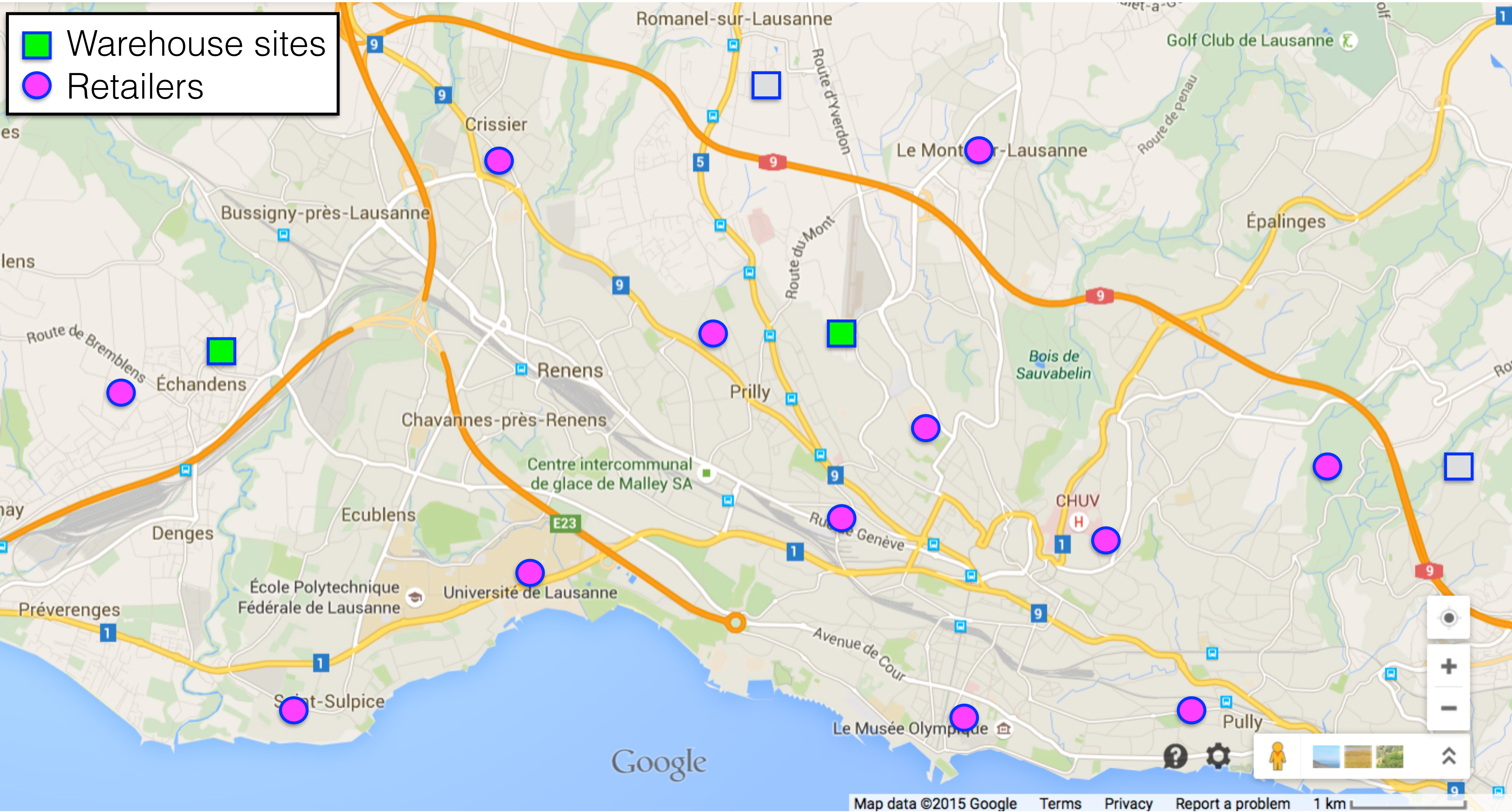
Conversely, let $\tilde{y} \in \mathbb{R}^n$ and $\tilde{\lambda} \in \mathbb{R}^J$ be any points that satisfy the above conditions, then $\tilde{y}$ and $\tilde{\lambda}$ are primal and dual optimal, with zero duality gap.

Examples

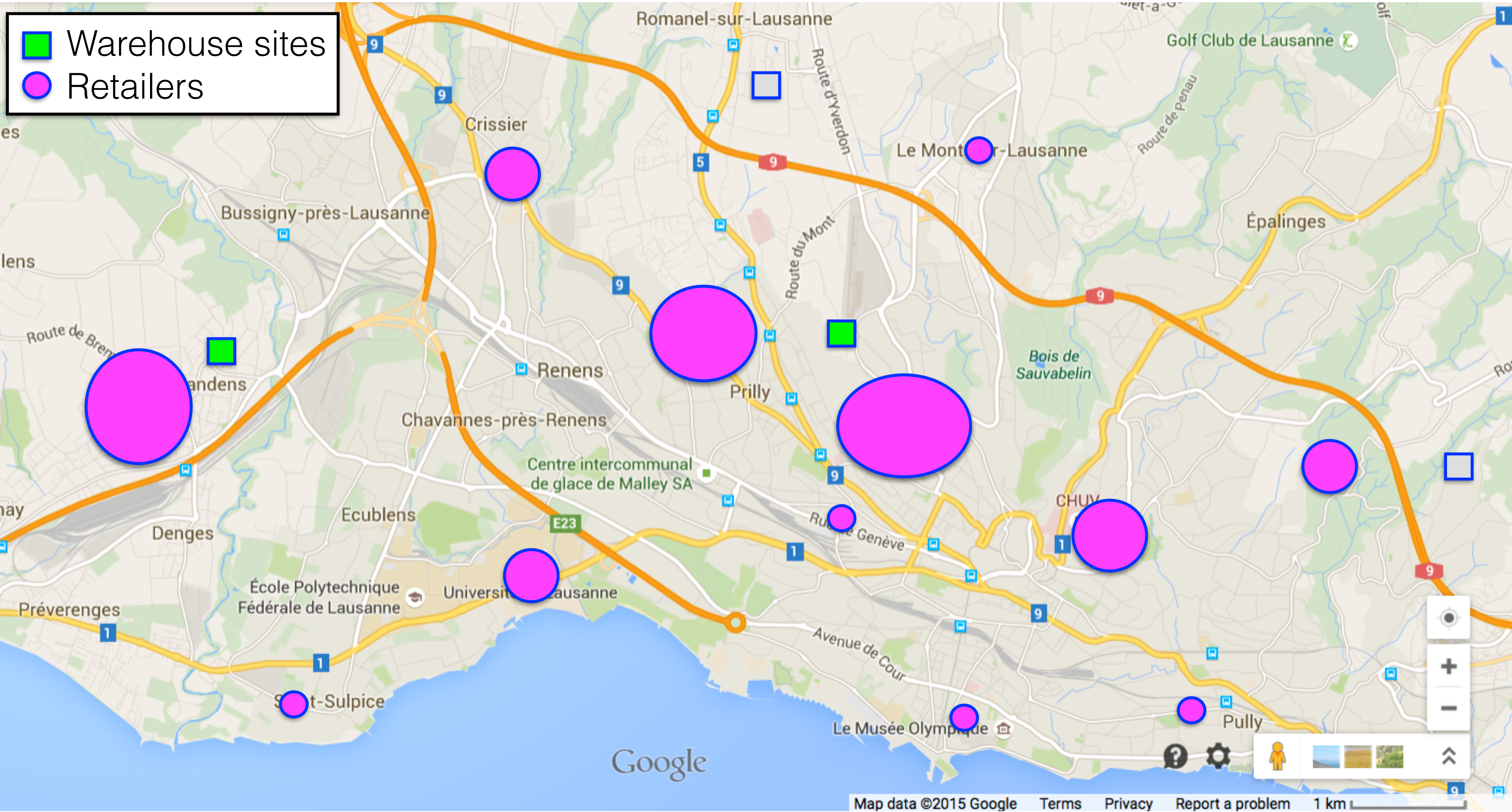
Exercise: Facility location problem



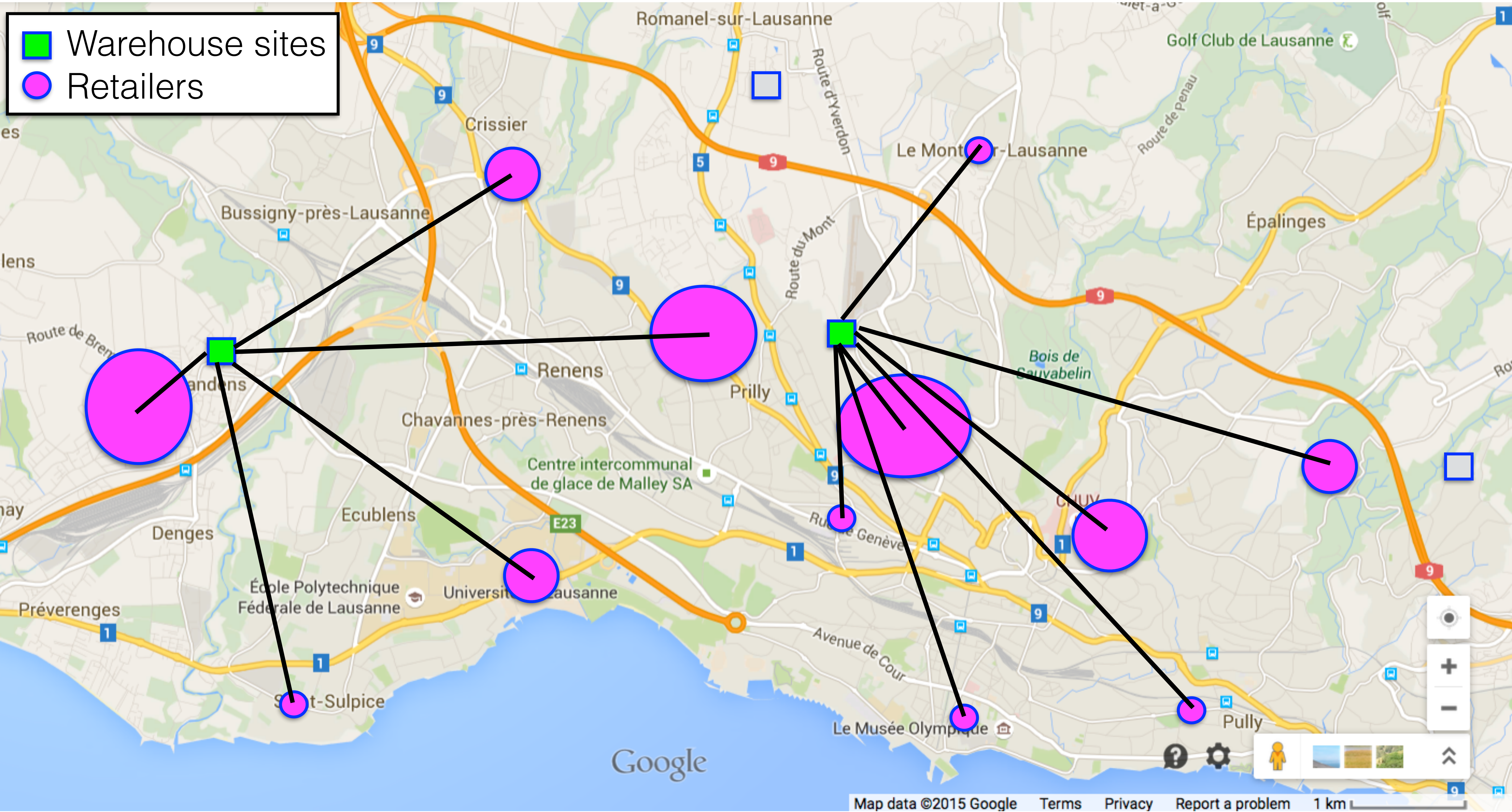
Exercise: Facility location problem



Exercise: Facility location problem



Exercise: Facility location problem



Exercise: Facility location problem

- Nominal decision problem:

$$\underset{x,y}{\text{maximize}} \quad -\sum_{i=1}^{n} c_i x_i + \sum_{j=1}^{m} (r_{ij} - d_{ij}) y_{ij} \tag{4.20a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} y_{ij} \leq D_j \,, \forall\, j \tag{4.20b}$$

$$\sum_{j=1}^{m} y_{ij} \leq P_i x_i \,, \forall\, i \tag{4.20c}$$

$$y_{ij} \geq 0 \;\forall\, i, j \tag{4.20d}$$

$$x \in \{0, 1\}^n \,, \tag{4.20e}$$

Exercise: Facility location problem

- Robust decision problem:

$$\underset{x,y(\cdot)}{\text{maximize}} \quad \inf_{z\in\mathcal{Z}(\Gamma)} \; -\sum_{i=1}^{n} c_i x_i + \sum_{j=1}^{m} (r_{ij} - d_{ij}) y_{ij}(z) \tag{4.21a}$$

$$\text{subject to} \quad \sum_{i} y_{ij}(z) \leq \bar{D}_j + \hat{D}_j z_j \,,\, \forall\, z \in \mathcal{Z}(\Gamma) \,,\, \forall\, j \tag{4.21b}$$

$$\sum_{j} y_{ij}(z) \leq P_i x_i \,,\, \forall\, z \in \mathcal{Z}(\Gamma) \,,\, \forall\, i \tag{4.21c}$$

$$y_{ij}(z) \geq 0 \,,\, \forall\, z \in \mathcal{Z}(\Gamma) \,,\, \forall\, i, j \tag{4.21d}$$

$$x \in \{0,1\}^n \,, \tag{4.21e}$$

$$\mathcal{Z}(\Gamma) := \{z \in \mathbb{R}^m \mid -1 \leq z \leq 1, \sum_{j=1}^{m} |z_j| \leq \Gamma\}$$