STRONG DUALITY REFORMULATION FOR BILEVEL OPTIMIZATION OVER NONLINEAR FLOW NETWORKS

Sophie Demassey (Mines Paris – PSL)

GDR ROAD - GrT PMNL - 23/06/2023

SINGLE-LEVEL REFORMULATIONS OF BILEVEL PROGRAMS

bilevel program

$$\min_{x,y} c(x,y)$$

$$s.t.\ y \in Y$$

$$x \in \arg\min_{z \mid g(z,y) \leq 0} f(z,y)$$

exact reformulation if:

- N&S global optimality conditions
- closed dual form

ex: KKT if convex + CQ; strong duality if linear, quadratic convex, SDP+feas, or monotropic

KKT (or FJ) optimality conditions:

$$g(x,y) \leq 0, u \geq 0, u^\top g(x,y) = 0$$
$$\nabla_x f(x,y) + u^\top \nabla_x g(x,y) = 0$$

value function:

$$g(x,y) \leq 0$$
$$f(x,y) \leq v(y) := \min_{z \mid g(z,y) \leq 0} f(z,y)$$

strong (lagrangian) duality:

$$g(x,y) \leq 0, u \geq 0,$$
$$f(x,y) \leq d(u,y) := \min_z f(z,y) + u^\top g(z,y)$$

1. *monotropic programs* (Rockafellar, 1988): convex programs with practical duals
2. special case: *nonlinear flow networks*
3. bilevel optimization in water distribution networks
4. applications of strong duality reformulation: convex reformulation, cut generation, splitting/alternating heuristic

MONOTROPIC PROGRAMMING (ROCKAFELLAR, 1988)

additive convex objective over linear constraints

$$P : \min_{x \in \mathbb{R}^J} \sum_{j \in J} f_j(x_j)$$

$$s.t. \sum_{j \in J} E_{ij} x_j = d_i \qquad \forall\, i \in I$$

f_j closed proper convex on $\mathbb{R}$ = lower semi-continuous (poss. nonsmooth)

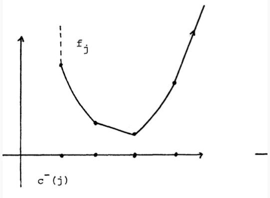


Figure 8.24

- monotropic aka "one-dimension convexity"*
- a class of convex programs behaving like linear programs:
  - combinatorial properties: finite set of descent directions (*elementary vectors*)
  - duality properties: strong duality, explicit symmetric dual

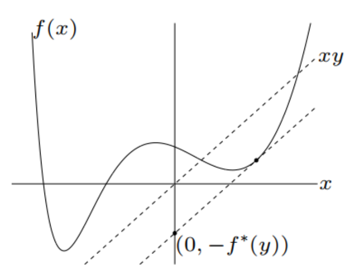
*extended to finite-dimension in [Bertsekas 2008]

MONOTROPIC PROGRAMMING: (FENCHEL) DUALITY

Let $f_j^* : v_j \in \mathbb{R} \mapsto \sup_{x_j}(x_j v_j - f_j(x_j))$ the *convex conjugate function* of f_j $\forall j \in J$

$$(P) : \min_{x \in \mathbb{R}^J} \sum_{j \in J} f_j(x_j)$$
$$s.t. \sum_{j \in J} E_{ij} x_j = d_i \qquad \forall i \in I$$

$$(D) : \min_{u \in \mathbb{R}^I} \sum_{i \in I} d_i u_i + \sum_{j \in J} f_j^*(v_j)$$
$$s.t.\ v_j := \sum_{i \in I} -E_{ij} u_i \qquad \forall j \in J$$



MONOTROPIC PROGRAMMING: (FENCHEL) DUALITY

Let $f_j^* : v_j \in \mathbb{R} \mapsto \sup_{x_j}(x_j v_j - f_j(x_j))$ the *convex conjugate function* of f_j $\forall j \in J$

$$(P) : \min_{x \in \mathbb{R}^J} \sum_{j \in J} f_j(x_j) \qquad \text{s.t.} \sum_{j \in J} E_{ij} x_j = d_i \quad \forall i \in I$$

$$(D) : \min_{u \in \mathbb{R}^I} \sum_{i \in I} d_i u_i + \sum_{j \in J} f_j^*(v_j) \qquad \text{s.t. } v_j := \sum_{i \in I} -E_{ij} u_i \quad \forall j \in J$$

- *conjugate* f_j^* is convex lower semi-continuous: D is monotropic
- *biconjugate* $f_j = f_j^{**}$ (as f_j convex l.s.c.): dual(dual)=primal
- *Fenchel inequality*: $f_j(x_j) + f_j^*(v_j) \geq x_j v_j$ and equality holds iff $v_j \in \partial f_j(x_j)$
- strong duality and KKT conditions for $(x; u, v)$ a feasible primal-dual pair:
  $0 = \sum_j \left(f_j(x_j) + f_j^*(v_j)\right) + \sum_i d_i u_i = \sum_j \left(f_j(x_j) + f_j^*(v_j) - x_j v_j\right) \iff v_j \in \partial f_j(x_j) \forall j$

MONOTROPIC PROGRAMMING: EQUIVALENT CONDITIONS (FINITE OPTIMUM)

primal: x solves

$$(P) : \min_{x} \sum_{j} f_j(x_j)$$

$$s.t. \sum_{j} E_{ij} x_j = d_i \qquad \forall i$$

dual: u solves

$$(D) : \min_{u} \sum_{i} d_i u_i + \sum_{j} f_j^*(v_j)$$

$$s.t. \; v_j := \sum_{i} -E_{ij} u_i \qquad \forall j$$

equilibrium (KKT): (x, u) solves

$$(Eq) : \sum_{j} E_{ij} x_j = d_i \qquad \forall i$$

$$v_j := \sum_{i} -E_{ij} u_i \in \partial f_j(x_j) \quad \forall j$$

strong duality: (x, u) solves

$$(SD) : \sum_{j} E_{ij} x_j = d_i \qquad \forall i$$

$$\sum_{j} \left(f_j(x_j) + f_j^*(v_j) \right) + \sum_{i} d_i u_i \le 0.$$

MONOTROPIC PROGRAMMING: APPLICATIONS

1. f_j piecewise linear/quad-convex

$$(P): \min_x \sum_j f_j(x_j)$$

$$s.t. \sum_j E_{ij} x_j = d_i \qquad \forall i$$

no need to linearize to dualize

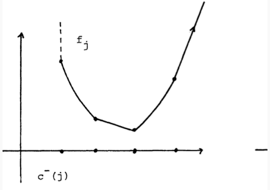


Figure 8.24

2. potential-flow network

$$(Eq): \sum_j E_{ji} x_j = d_i \qquad \forall i$$

$$v_j = \sum_i -E_{ji} u_i \in \partial f_j(x_j) \qquad \forall j$$

- E incidence matrix of graph G(I,J)
- x arc flows, u node potentials
- ∂f arc resistance/conductivity

POTENTIAL-FLOW NETWORK

- transportation of a commodity on a digraph $G = (N, A)$ of incidence matrix $E \in \{0, 1, -1\}^{A \times N}$
- flow $x_a \in \mathbb{R}$: volume/rate on arc $a \in A$, sign=direction

$$\sum_a E_{an} x_a = d_n \quad \text{flow conservation/demand at nodes } n \in N$$

- potential $u_n \geq 0$: energy at node $n \in N$
- potential loss $v_a := u_i - u_j = -\sum_n E_{an} u_n$ along $a = (i, j) \in A$ is related to flow x_a:

$$v_a = \phi_a(x_a) \quad \text{resistance/conductivity of arc } a = (i, j) \in A$$

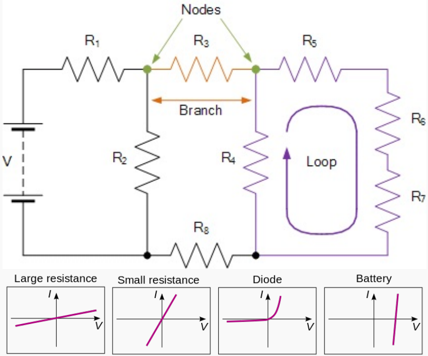
- model for many physical networks (of newtonian fluids): electricity, water, gas, heat, telecommunications, transportation, vascular, elastic/spring

STEADY-STATE NETWORK EQUILIBRIUM PROBLEM

Given boundary conditions (some fixed flow/potential values), compute overall arc flows x_A and node potentials u_N satisfying:

- flow conservation at nodes $E_{An}^{\top} x_A = d_n,\ \forall\, n \in N$
- resistance relation on arcs $v_a := -\sum_n E_{an} u_n = \phi_a(x_a),\ \forall\, a \in A$

EX 1: ELECTRIC CIRCUIT



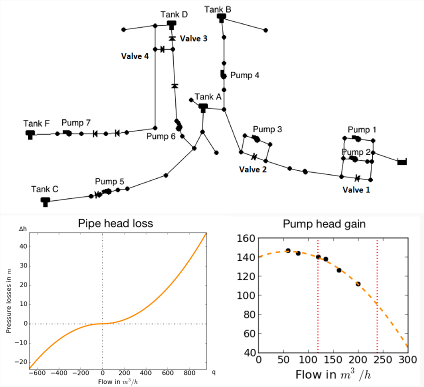
- A: conductors (resistors, batteries,...)
- x: current (I)
- v: voltage (V)
- flow conservation = Kirchhoff's current law
- linear resistance $R = V/I$ (Ohm's law)

CASE 1: EQUILIBRIUM WITH LINEAR RESISTANCE

$$\phi_a(x_a) = r_a x_a$$

- laws of Ohm (electric), Fourier (thermal), Poiseulle (viscous fluids)
- well studied in the electric context: existence, unicity, reduction
- equilibrium solution minimizes energy dissipation:

$$(P) : \min_{x, Ex=d} \sum_{a \in A} \frac{r_a}{2} x_a^2 = \sum_{a \in A} f_a(x_a) \text{ with } f'_a(x_a) = \phi_a(x_a).$$



- A = { pipes, pumps, valves }
- $N = S \cup R$: Service nodes (junctions) and Reservoir (tanks, sources)
- x = water flow rate
- u = hydraulic head = pressure + elevation
- nonlinear resistance: friction in pipes (Darcy-Weisbach's law), charge gain in pumps

- boundary conditions: known demand d_n at service nodes $n \in S$, known potential/level u_n at reservoirs $n \in R$

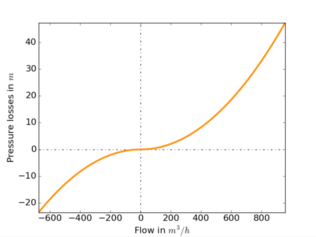
equilibrium problem

$$
\begin{aligned}
NAP(E, d_S, u_R) = \{ &(x_A, u_S) \in \mathbb{R}^A \times \mathbb{R}^S, && && \text{(flows, potentials)} \\
& x_s := \sum_a E_{as} x_a = d_s && \forall\, s \in S, && \text{(flow conservation)} \\
& v_a := \sum_n -E_{an} u_n = \phi_a(x_a) && \forall\, a \in A \} && \text{(resistance)}
\end{aligned}
$$

In practice: a system of equations solved by the Newton-Raphson algorithm (e.g. Epanet). The boundary conditions ensure a solution exists and is unique.

COMFY ASSUMPTIONS FOR THE NONLINEAR CASE

the resistance function ϕ_a is *continuous*, *strictly increasing*, and bijective on $\mathbb{R}$
$\Rightarrow$ the integral $f_a(x) = \int_0^x \phi_a(t)dt$ is smooth, strictly convex, and coercive
$\Rightarrow$ the same for the conductivity function $\psi_a = \phi_a^{-1}$ and its integrals



Examples:

- friction in pipes $\phi_a(x) = sgn(x)\alpha_a|x|^p$ with $p = 2$ (water) or $p = 1.852$ (gas)
- discharge pressure in pumps $\phi_a(x) = \alpha_a x|x| + \beta_a x + \kappa_a$ with $\alpha_a > 0$

REFORMULATION OF THE NETWORK ANALYSIS PROBLEM: PRIMAL

equilibrium problem

$$NAP(E, d_S, u_R) = \{(x_A, u_S) \mid x_S = d_S, v_a = \phi_a(x_a) \forall\, a \in A\}$$

$(x_A, u_S) \in NAP(E, d_S, u_R)$ for some u_S if and only if x_A solves

primal distribution problem

$$P(E, d_S, u_R) : \min_{x_A} \{f(x_A) = \sum_{a \in A} f_a(x_a) + u_R^\top x_R \mid x_S = d_S\}$$

where $f_a = \int \phi_a$, $f_a(0) = 0$ is l.s.c. convex

$$L(x_A, u_S) = f(x_A) + u_S^\top (x_S - d_S) = \textstyle\sum_{a \in A} f_a(x_a) - v_A^\top x_A - u_S^\top d_S.$$

NAP: the stationary points $\left(\phi_a(x_a) - v_a = \frac{\partial L}{\partial x_a} = 0, x_s - d_s = \frac{\partial L}{\partial u_s} = 0\right)$ of lagrangian L $\implies$ the primal-dual optimizers of P (convex+LCQ).

Strong duality reformulation of $NAP(E, d_S, u_R)$

$$SDNAP(E, d_S, u_R) = \{(x_A, u_S) \in \mathbb{R}^A \times \mathbb{R}^S, x_S = d_S, \sum_{a \in A} \left(f_a(x_a) + f_a^*(v_a)\right) + u_R^\top x_R + d_S^\top u_S \leq 0\} \qquad (SD)$$

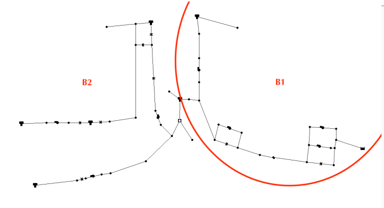
with $f_a(x_a) = \int_0^{x_a} \phi_a(t)dt$ and $f_a^*(v_a) = -f_a(\phi_a^{-1}(v_a)) + v_a \phi_a^{-1}(v_a)$ convex.

- (SD) integrates and aggregates the flow-potential equations:

$$\begin{aligned} (SD) \quad &\iff f_a(x_a) + f_a^*(v_a) = x_a v_a, \forall a \qquad \text{(addends are non-negative)} \\ &\iff f_a(x_a) = f_a(\phi_a^{-1}(v_a)) + f_a'(\phi_a^{-1}(v_a))(x_a - \phi_a^{-1}(v_a)) \forall a \\ &\iff \phi_a^{-1}(v_a) = x_a \forall a. \qquad (f_a \text{strictly convex}) \end{aligned}$$

APARTÉ: SPATIAL DECOMPOSITION OF NETWORK ANALYSIS PROBLEM

Let $G = \cup_{b \in B}(N_b, A_b)$ a graph partition along some R nodes, then



equilibrium problem

$$NAP(E, d_S, u_R) = \bigcup_{b \in B} NAP(E_b, d_{S_b}, u_R)$$

$$= \bigcup_{b \in B} \{(x, u) \in \mathbb{R}^{A_b} \times \mathbb{R}^{S_b},$$

$$x_s := \sum_{a \in A_b} E_{as} x_a = d_s \qquad \forall s \in S_b$$

$$v_a := \sum_{n \in N_b} -E_{an} u_n = \phi_a(x_a) \qquad \forall a \in A_b\}$$

discrete bilevel models
for
network optimization

APPLICATION TO NETWORK OPTIMIZATION

- network design: select the element to install to satisfy a worst-case demand scenario and minimize installation costs
- network operation: reconfigure the network dynamically on a discrete horizon to meet the demand profiles and minimize operation costs

bilevel structure:

1. MILP: get a layout $E(y_A)$, $y_a \in \{0,1\}$ and boundary conditions (d_S^0, u_R)
2. NLP: solve equilibrium $(x_A, u_S) \in NAP(E(y_A), d_S, u_R)$
3. repeat in the dynamic case, with u_R^t in (1) depends on x_A^{t-1} from (2)

PIPE LAYOUT IN GRAVITY-FED WATER NETWORKS (STATIC CASE)

- every node has a fixed demand D_S or a fixed potential U_R (sources)
- arcs are pipes (no pumps) to select in a discrete set K:

$$y_{ak} \in \{0,1\} \text{ select pipe of type } k \text{ on arc } a \in A?$$

- model on graph $G = (N, A^K)$ with replicated arcs:

$$\min_{y,x,u} \sum_a \sum_k c_{ak} y_{ak}$$

$$s.t. (x_{AK}, u_S) \in NAP(E(y_{AK}), D_S, U_R)$$

$$y_{ak} = 0 \implies x_{ak} = v_{ak} = 0 \qquad \forall a \in A, k \in K$$

$$\sum_{k \in K} y_{ak} = 1, u_i - u_j = \sum_k v_{ak} \qquad \forall a = (i,j) \in A.$$

PIPE LAYOUT: EXACT CONVEX MINLP REFORMULATION

$$
\begin{aligned}
\min \sum_{a} \sum_{k} & c_{ak} y_{ak} \\
s.t. \sum_{a \in A} \sum_{k \in K} & \left(f_{ak}(x_{ak}) + f^*_{ak}(v_{ak}) \right) + U_R^\top x_R + D_S^\top u_S \leq 0 && (SD) \\
& y_{ak} = 0 \implies x_{ak} = v_{ak} = 0 && \forall a \in A, k \in K \\
& \sum_{k \in K} y_{ak} = 1, u_i - u_j = \sum_{k} v_{ak} && \forall a = (i,j) \in A.
\end{aligned}
$$

- $f_{ak}(0) = f^*_{ak}(0) = 0 \implies f_{ak}(x_{ak}) + f^*_{ak}(v_{ak}) = \left(f_{ak}(x_{ak}) + f^*_{ak}(v_{ak}) \right) y_{ak}$
- convex but not polynomial[†]

[†]Tasseff et al. (2020) Exact MICP Formulation for Optimal Water Network Design.

PUMP SCHEDULING IN PRESSURIZED NETWORKS (DYNAMIC CASE)

- controllable arcs (pumps, valves) are switch on/off on a discrete horizon T:

$$y_{at} \in \{0,1\} \text{ active arc } a \in A \text{ on time } t \in T?$$

- fixed demand D_{St} at service nodes $\forall t \in T$
- tank level: fixed **only at $t = 0$**, then $u_{r(t+1)}$ depends on x_{rt} (residual inflow)
- a sequence-dependent sequence of NAPs:

$$\min \sum_a \sum_t c^0_{at} y_{at} + c^1_{at} x_{at}$$

$$s.t. (x_{At}, u_{St}) \in NAP(E(y_{At}), D_{St}, u_{Rt}) \qquad \forall t \in T$$

$$y_{at} = 0 \implies x_{at} = 0 \qquad \forall a \in A, t \in T$$

$$u_{R(t+1)} = u_{Rt} + s_R^\top x_{Rt} \qquad \forall t \in T$$

$$\underline{U}_{Rt} \leq u_{Rt} \leq \overline{U}_{Rt} \qquad \forall t \in T.$$

strong duality constraints are nonconvex

$$\sum_{a \in A} \left(f_a(x_{at}) + f_a^*(v'_{at}) \right) + u_{Rt}^\top x_{Rt} + D_{St}^\top u_{St} \leq 0 \quad \forall t$$

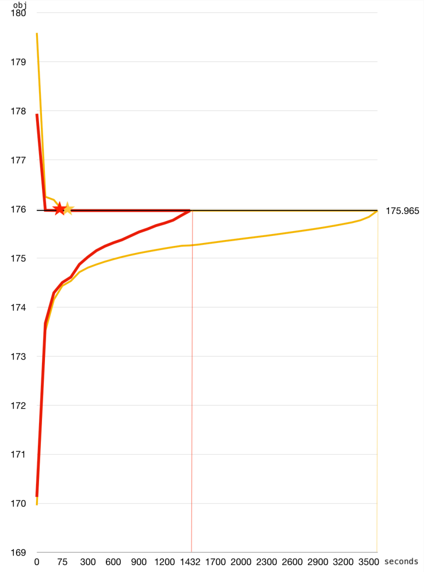
with $y_{at} = 1 \implies v'_{at} = v_{at}$ and $y_{at} = 0 \implies v'_{at} = (f_a^*)^{-1}(0)$

First application: relax and generate cuts:

- $f_a(x_{at}) + f_a^*(v'_{at}) = \left(f_a(x_{at}) + f_a(v_{at}) \right) y_{at}$ is convex $\implies$ linearize at trial points
- bad news: a loose relaxation of the bilinear terms may *absorb* the duality gap
- good news: tank capacities provide exogenous bounds on u_{Rt}, $u_{R(t+1)}$ and x_{Rt} to tighten McCormick's relaxation

1ST APPLICATION OF SD REFORMULATION: CUT GENERATION

- evolution of the primal/dual bounds in
- Branch-and-Check [Bonvin, Demassey, Lodi 2020]
- with or without duality cuts



2ND APPLICATION: DECOMPOSITION OF PUMP SCHEDULING

$$\min_{y,x,u} \sum_a \sum_t (c^0_{at} y_{at} + c^1_{at} x_{at})$$

$$s.t. (x_{At}, u_{St}) \in NAP(E(y_{At}), D_{St}, u_{Rt}) \qquad \forall t \in T$$

$$u_{R(t+1)} = u_{Rt} + s_R^\top x_{Rt} \qquad \forall t \in T$$

$$\underline{U}_{Rt} \leq u_{Rt} \leq \overline{U}_{Rt} \qquad \forall t \in T.$$

- complexity comes less from the nonconvex constraints $v_a = \phi_a(x_a)$ than from the inter-dependency $x_t = F(y_t, u_t)$, and $u_{t+1} = G(x_t)$
- $\implies$ bilinear terms $u_r x_r$ in the dual formulation
- dualizing the time-coupling constraints does not change this complexity
- fixing the time-coupling variables u_{Rt} $\implies$ decompose & enumerate

ALTERNATE DIRECTION METHOD 1 (DOUGLAS-RACHFORD PRINCIPLE)

$$\min_{y,x,u,X,U} \sum_a \sum_t (c^0_{at} y_{at} + c^1_{at} x_{at}) + \sum_r \sum_t \left(\sum_b \mu^b_{rt} \| x^b_{rt} - X^b_{rt} \|^2 + \nu_{rt} \| u_{rt} - U_{rt} \|^2 \right)$$

$$s.t.\ (x_{A_b t}, u_{S_b t}) \in NAP_b(E_b(y_{A_b t}), D_{S_b t}, U_{R_b t}) \qquad \forall t \in T, b \in B$$

$$U_{R(t+1)} = U_{Rt} + s_R^\top \sum_b X^b_{Rt},\ \underline{U}_{Rt} \le U_{Rt} \le \overline{U}_{Rt} \qquad \forall t \in T.$$

$(P1)$: fix U_{RT} (test y_{AT}) get x_{AT} $(P2)$: fix x_{AT} get U_{RT} 3: update μ, ν ‡

- $(P1)$ becomes decomposable both in time and space, thus enumerable
- relax NAP in $(P2)$; unlikely convergent bc not linearly separable: $U_{rt} x_{rt}$

‡ongoing work with Valentina Sessa and Amir Tavakoli with U^0 generated by ML

ADM 2 APPLIED TO THE (SD) REFORMULATION (WORK IN PROGRESS)

First, dualize (SD), for any $\lambda_{bt} \geq 0$:

$$\min \sum_a \sum_t (c^0_{at} y_{at} + c^1_{at} x_{at}) + \sum_r \sum_t \left(\sum_b \mu^b_{rt} \|x^b_{rt} - X^b_{rt}\|^2 + \nu_{rt} \|u_{rt} - U_{rt}\|^2 \right)$$

$$+ \sum_b \sum_t \lambda_{bt} \left(\sum_{a \in A_b} (f_a(x_{at}) + f^*_a(u_{at})) + \sum_{r \in R_b} U_{rt} x^b_{rt} + \sum_{s \in S_b} D_{st} u_{st} \right)$$

$$s.t.\ x_{St} = D_{St},\ U_{R(t+1)} = U_{Rt} + s^\top_R \sum_b X^b_{Rt},\ \underline{U}_{Rt} \leq U_{Rt} \leq \overline{U}_{Rt} \qquad \forall t \in T.$$

$(P1)$: fix U_{RT} (test y_{AT}) get x_{AT}	$(P2)$: fix x_{AT} get U_{RT}	3: update μ, ν

No need to relax $(P2)$ anymore, and $(P1)$ becomes separable as independent penalized NAPs

REPORT COSTS/PENALTIES TO THE LOWER (NAP) LEVEL

$$(P1) = \min_{y,x,u|x_{ST}=D_{ST}} \sum_a \sum_t (c^0_{at} y_{at} + c^1_{at} x_{at}) + \sum_r \sum_t \left(\sum_b \mu^b_{rt} \|x^b_{rt} - X^b_{rt}\|^2 + \nu_{rt} \|u_{rt} - U_{rt}\|^2 \right)$$

$$+ \sum_b \sum_t \lambda_{bt} \left(\sum_{a \in A_b} (f_a(x_{at}) + f^*_a(v_{at})) + \sum_{r \in R_b} U_{rt} x^b_{rt} + \sum_{s \in S_b} D_{st} u_{st} \right)$$

$$= \sum_t \sum_b \min_{y_t} \quad f^\lambda_{bt}(y) + g^\lambda_{bt}(y) + \sum_{a \in A_b} c^0_{at} y_{at}.$$

$$\text{primal/dual } \textit{penalized} \text{ NAPs} = \begin{cases} f^\lambda_{bt}(y) = \min_{x, x_S = D_{St}} \sum_{a \in A^b x_a = 1} f^\lambda_a(x_a) + \sum_{r \in R^b} U^\lambda_r x_r \\ g^\lambda_{bt}(y) = \min_u \sum_{a \in A^b, x_a = 1} g^\lambda_a(v_a) + \sum_{s \in R^b} D^\lambda_s u_s \end{cases}$$

CONCLUSION

- reveal the bilevel structure of some nonconvex MINLP to derive convex MINLP reformulation or cut families
- flow networks and monotropic optimization at the inner level of many practical problems: exploit the special duality and variational characteristics

REFERENCES

- our papers on the pump scheduling problem are available on https://sofdem.github.io/
- code available on: https://github.com/sofdem/gopslpnlpbb (find the right branch!)