# Efficient computation of convex hull prices with level and subgradient methods: a computational comparison of dual methods

*Sofiane Tanji*[1], Yassine Kamri[1], François Glineur[1], Mehdi Madani[1,2]

[1]Université catholique de Louvain

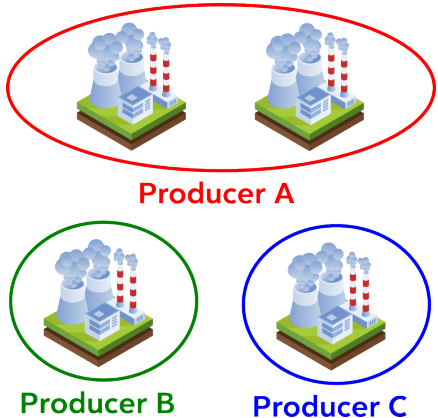
[2]N-SIDE

EUROPT 2024 | June 2024

# *Talk Outline*

1. Quick introduction to electricity markets and CH pricing
2. Formulation of the CH pricing problem
3. First-order methods to compute CH prices
4. Numerical experiments
5. Conclusion

# *Setting*




Electric demand
(Load)

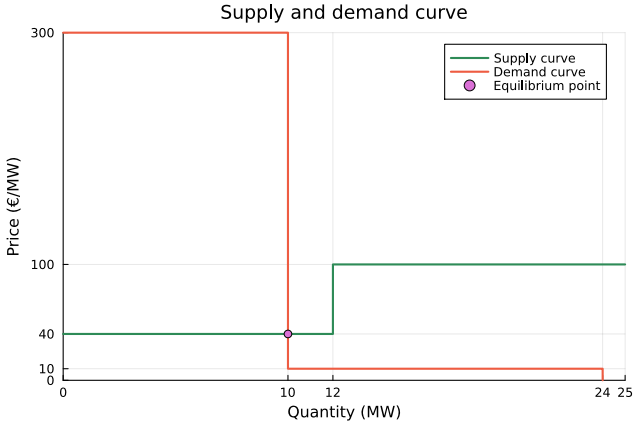
Multiple decision makers: each producer and buyer seeks to maximize its individual benefits.

# *Supply and demand example*

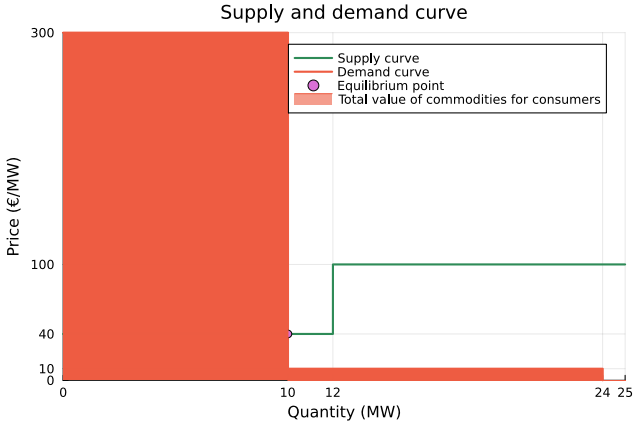
Consider a market with 2 producers and 2 buyers.

| | Quantity (MW) | Limit price (€/MW) | Startup cost (€) |
|---|---|---|---|
| Buyer 1 | 10 | 300 | - |
| Buyer 2 | 14 | 10 | - |
| Producer 1 | 12 | 40 | 200 |
| Producer 2 | 13 | 100 | - |

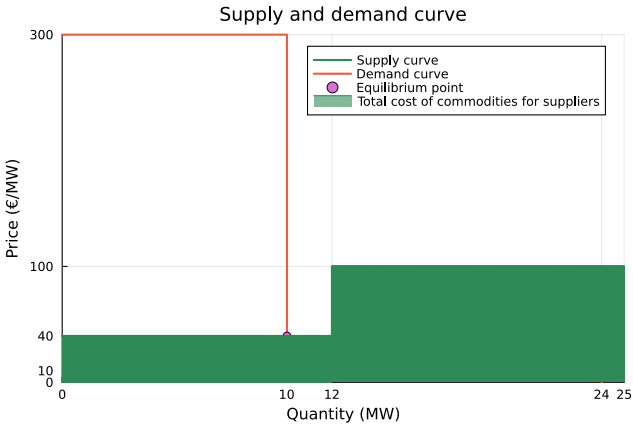
# *Supply and demand example*



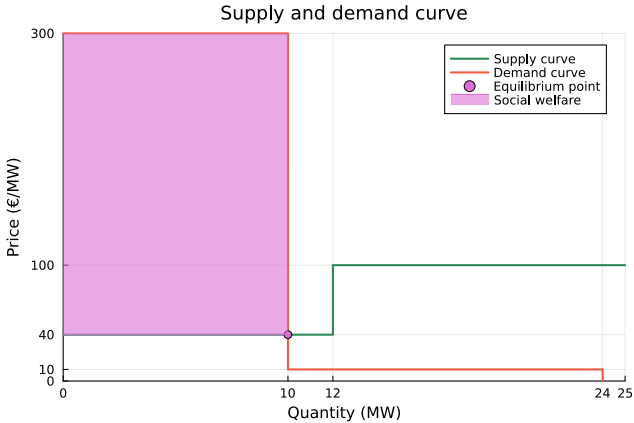
# *Supply and demand example*



# *Supply and demand example*



# *Supply and demand example*




Maximizing social welfare maximizes value for consumers while minimizing cost for producers: both producers and consumers are happy !

# *Supply and demand example*

At (40€/MW), Producer 1 would rather produce 12MW than 10MW: following market equilibrium leads to a lost opportunity cost (LOC).

- Maximizing social welfare with uniform pricing is not equivalent to having a market equilibrium: individual agents may have LOCs.
- In this example, the market operator should make a side payment to Producer 1 to compensate its LOC.

# *Summary and CH pricing*

The market operator is an independent entity making the following decisions:

- Goal: Maximize social welfare (with uniform prices) under operating constraints and power balance equality
- Side payments must be made whenever market equilibrium is not equivalent to maximizing social welfare.
- There are many pricing rules and each may lead to different side payments.
- The CH prices are defined as the prices minimize these side payments.

# *Talk Outline*

1. Quick introduction to electricity markets and CH pricing
2. Formulation of the CH pricing problem
3. First-order methods to compute CH prices
4. Numerical experiments
5. Conclusion

# *Primal formulation*

Producers must match generation and demand under technical constraints while minimizing the cost for each participant.
This leads to the following MILP, named "Unit Commitment problem":

$$\min_{\xi, l} \left[ \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} \text{cost}(\xi_t^g) - VOLL \sum_{t \in \mathcal{T}} l_t \right] \tag{1}$$

subject to:

$$0 \leq l_t \leq L_t \qquad \forall t \in \mathcal{T} \tag{2}$$

$$\xi_g \in \Pi^g \qquad \forall g \in \mathcal{G} \tag{3}$$

$$\left( \sum_{g \in \mathcal{G}} p_g^t \right) - l_t = 0 \qquad \forall t \in \mathcal{T} \quad [\lambda_t] \tag{4}$$

$$\min_{\xi, l} \left[ \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} \text{cost}(\xi_t^g) - VOLL \sum_{t \in \mathcal{T}} l_t \right] \tag{1}$$

subject to:

$$0 \leq l_t \leq L_t \qquad \forall t \in \mathcal{T} \tag{2}$$

$$\xi_g \in \Pi^g \qquad \forall g \in \mathcal{G} \tag{3}$$

$$\left( \sum_{g \in \mathcal{G}} p_g^t \right) - l_t = 0 \qquad \forall t \in \mathcal{T} \quad [\lambda_t] \tag{4}$$

Here, $\xi_g = \left(p_g, \bar{p}_g, u_g, v_g, w_g\right)$ is the state of a generator $g$, $\mathcal{T}$ is the time horizon, $\mathcal{G}$ the set of producers, $L_t$ is the demand of the consumer at time $t$ and $l_t$ the demand met by the market.

# *Partial Lagrangian dual*

Applying the Lagrangian relaxation technique on the power balance constraint, we get the following Lagrangian function:

$$L(\xi, l, \lambda) := \sum_{t \in \mathcal{T}} \sum_{g \in \mathcal{G}} \text{cost}(\xi_t^g) - VOLL \sum_{t \in \mathcal{T}} l_t - \sum_{t \in \mathcal{T}} \lambda_t \left[ \sum_{g \in \mathcal{G}} p_g^t - l_t \right] \tag{5}$$

The dual problem is then:

$$\max_{\lambda} \mathcal{L}(\lambda) = \min_{\xi, l} L(\xi, l, \lambda) \tag{6}$$

subject to:

$$0 \leq l_t \leq L_t \; \forall t \in \mathcal{T}$$

$$\xi_g \in \Pi^g \; \forall g \in \mathcal{G}$$

# *CH prices*

Remember that CH prices are defined as the prices minimizing the side payments done by the market operator. Authors in Gribik et al. (2007) show that:

$$\min_{\lambda} \sum_{p \in \text{participants}} \text{uplifts} = \min_{\lambda}\{\text{duality gap}\} = \min_{\lambda}\{UC - \mathcal{L}(\lambda)\} = UC - \max_{\lambda} \mathcal{L}(\lambda)$$

This leads to an alternative definition of CH prices:

## Definition

The convex hull prices are the optimal variables of the Lagrangian dual associated to the unit commitment problem in which one dualizes the balance condition.

Note: This is equivalent (see Hua and Baldick (2016)) to querying the dual variables associated to the balance condition in the primal problem where for all generators $g$, $\mathcal{P}^g$ have been replaced by $\text{conv}(\mathcal{P}^g)$.

# *Overview of methods*

Two types of methods to compute CH prices:

1. So-called primal methods which directly tackle the primal CH problem (with the $\mathcal{P}^g$ being replaced by $\text{conv}(\mathcal{P}^g)$) and obtain prices by querying the dual value associated with the power balance constraint
2. Our focus in this talk: dual methods which focus on the maximization of the Lagrangian, a piece-wise linear concave function using methods from nonsmooth optimization.

# *Talk Outline*

1. Quick introduction to electricity markets and CH pricing
2. Formulation of the CH pricing problem
3. First-order methods to compute CH prices
4. Numerical experiments
5. Conclusion

# *First-order oracle*

$$\pi^* = \arg\max_{\pi \in Q} \left[ \mathcal{L}(\pi) = \mathcal{L}_0(\pi) + \sum_{g \in \mathcal{G}} \mathcal{L}_g(\pi) \right] \tag{7}$$

where:

$$\mathcal{L}_0(\pi) = \min_{\mathbf{l}} \sum_{t \in \mathcal{T}} [C_{VOLL}(L_t - l_t) + \pi_t l_t] \tag{8}$$

subject to: $l_t \leq L_t \quad \forall t \in \mathcal{T}$.

and:

$$\mathcal{L}_g(\pi) = \min_{(\mathbf{p}, \bar{\mathbf{p}}, \mathbf{u}, \mathbf{v}, \mathbf{w})} \sum_{t \in \mathcal{T}} [C(u_t^g, v_t^g, p_t^g) - \pi_t p_t^g] \tag{9}$$

subject to: $(p^g, \bar{p}^g, u^g, v^g, w^g) \in \mathcal{P}^g \qquad \forall t \in \mathcal{T}$.

# First-order oracle

$$\pi^* = \arg\max_{\pi \in Q} \left[ \mathcal{L}(\pi) = \mathcal{L}_0(\pi) + \sum_{g \in \mathcal{G}} \mathcal{L}_g(\pi) \right] \tag{7}$$

## Lemma

*First-order oracle $\mathcal{L}$ is nonsmooth, concave and piece-wise linear with supgradient:*

$$\hat{\partial}\mathcal{L}(\pi) = \left[ \hat{\partial}\mathcal{L}_0(\pi) + \sum_{g \in \mathcal{G}} \hat{\partial}\mathcal{L}_g(\pi) \right] \ni \left[ l^* - \sum_{g \in \mathcal{G}} p_*^g \right] \tag{8}$$

*where $(\mathbf{l}^*, \mathbf{p}_*^g)$ are the optimal values of respectively (8) and (9).*

## Proof.

Proof is standard and may be found in (Conforti et al., 2014, Corollary 8.3) □

# *Subgradient-based methods*

**Algorithm 1** Subgradient method (vanishing step lengths) | Boyd et al. (2003)

**Parameters:** initial step length $\eta$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Select a subgradient $g^k \in \partial \bar{\mathcal{L}}(x^k)$.
2. Compute $x^{k+1} = P_X\left(x^k - t_k \frac{g^k}{\|g^k\|}\right)$ with $t_k = \frac{\eta}{k}$ or $t_k = \frac{\eta}{\sqrt{k}}$.

**Output:** best iterate $x_{best}$

# *Subgradient-based methods*

**Algorithm 2** Subgradient method (estimated Polyak step lengths) | Boyd et al. (2003)

**Parameters:** initial suboptimality estimate $\alpha > 0$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Select a subgradient $g^k \in \partial \bar{\mathcal{L}}(x^k)$.
2. Compute $x^{k+1} = P_X\left(x^k - t_k \frac{g^k}{\|g^k\|^2}\right)$ with $t_k = \bar{\mathcal{L}}^k - (\bar{\mathcal{L}}^k_{best} - \frac{\alpha}{k})$.

**Output:** best iterate $x_{best}$

Polyak steplength is $\bar{\mathcal{L}}^k - \bar{\mathcal{L}}^*$.
When $\bar{\mathcal{L}}^*$ is unknown, it is approximated by $\bar{\mathcal{L}}^k_{best} - \frac{\alpha}{k}$.

# *Subgradient-based methods*

**Algorithm 3** Last-iterate optimal subgradient method | Zamani and Glineur (2023)

**Parameters:** number of iterations $N$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$ satisfying $\|x^1 - x^*\| \leq R$ for some minimizer $x^*$.

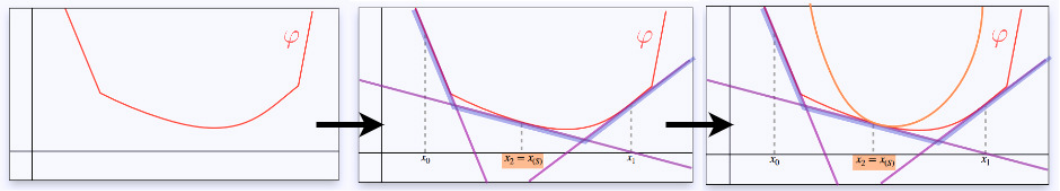
For $k = 1, 2, \ldots, N$ perform the following steps:

1. Select a subgradient $g^k \in \partial\bar{\mathcal{L}}(x^k)$.
2. Compute $x^{k+1} = P_X\left(x^k - t_k \frac{g^k}{\|g^k\|}\right)$ with $t_k = \frac{R(N+1-k)}{\sqrt{(N+1)^3}}$.

**Output:** last iterate $x_{N+1}$

# *Bundle methods*

## Lemaréchal et al. (1995)




Three steps:

1. Bundle information:
$$x \mapsto \varphi(x_i) + g_\varphi^\top (x - x_i)$$
2. Polyhedral approximation:
$$\hat{\varphi}(x) = \max_{i \in \text{Bundle}} \varphi(x_i) + g_\varphi^\top (x - x_i)$$
3. Stability center in the bundle ← this is where bundle methods vary

# *Bundle Level Method*

**Algorithm 4** Bundle Level Method (BLM) | Lemaréchal et al. (1995)

**Parameters:** level set parameter $\alpha \in (0, 1)$.

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$

Initialize $UB = +\infty$, $LB = -\infty$.

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Compute first-order oracle $(\bar{\mathcal{L}}(x^k), g^k \in \partial\bar{\mathcal{L}}(x^k))$.
2. Update upper bound $UB = \min\{UB, \bar{\mathcal{L}}(x^k)\}$.
3. Update polyhedral model
   $LB = \min\{\ t \text{ s.t. } x \in X \text{ and } \bar{\mathcal{L}}(x^i) + \langle \partial\bar{\mathcal{L}}(x^i), x - x^i \rangle \leq t,\ \forall i \leq k\}$.
4. Update level set $\mathcal{S} = LB + \alpha(UB - LB)$.
5. Update iterate
   $x^{k+1} = \min\{\ \|x - x^k\|^2 \text{ s.t. } x \in X \text{ and } \bar{\mathcal{L}}(x^i) + \langle \partial\bar{\mathcal{L}}(x^i), x - x^i \rangle \leq \mathcal{S},\ \forall i \leq k\}$.

**Output:** best iterate $x_{best}$

# *Bundle Proximal Level Method*

**Algorithm 5** Bundle Proximal Level Method (BPLM) | Lemaréchal et al. (1995)

**Parameters:** level set parameter $\alpha \in (0, 1)$.

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, $x^1 \in X$, $UB = +\infty$, $LB = -\infty$, $\Delta = +\infty$, $\mathcal{S}' = +\infty$.

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Compute oracle, update upper bound, polyhedral model and level set.
2. **if** $UB - LB \geq (1 - \alpha)\Delta$ **then**
3. Update proximal level set $\mathcal{S}' = \min\{\mathcal{S}, \mathcal{S}'\}$.
4. **else** ▷ Regular level set provides sufficient decrease
5. Update proximal level set $\mathcal{S}' = \mathcal{S}$ and proximal gap $\Delta = UB - LB$
6. **end if**
7. Update iterate
   $x^{k+1} = \min\{ \|x - x^k\|^2 \text{ s.t. } x \in X \text{ and } \bar{\mathcal{L}}(x^i) + \langle \partial\bar{\mathcal{L}}(x^i), x - x^i \rangle \leq \mathcal{S}',\ \forall i \leq k\}$.

**Output:** best iterate $x_{best}$

# *Parameter-free methods*

Parameter-free: do not require the input of any problem parameters and/or problem classes
We considered two methods: D-Adaptation Defazio and Mishchenko (2023) and DowG Khaled et al. (2023).
Key ideas:

- For subgradient-based methods, the optimal stepsize depends on unknown quantities: $\|x_0 - x^*\|$ and the Lipschitz constant of the objective
- Normalizing the stepsize (divide by $\sqrt{\sum_{i=0}^{k} \|g^i\|^2}$) removes the dependency on the Lipschitz constant
- Parameter-free methods maintain and update a lower bound on the initial distance to circumvent the need for the exact quantity.
- Two regimens in practice: one where the distance estimator is tuning itself automatically and a faster one when an appropriate value has been found.

# *Parameter-free methods*

**Algorithm 6** D-Adaptation (DA)

**Parameters:** initial lower bound $D_1$ on $\|x^1 - x^*\|$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$

Initialize $s^1 = 0, \gamma^1 = \|g^1\|^{-1}$.

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Select a subgradient $g^k \in \partial\bar{\mathcal{L}}(x^k)$.
2. Compute $s^{k+1} = s^k + D_k g^k$.
3. Compute $\gamma^{k+1} = (\sum_{i=1}^k \|g^i\|^2)^{-\frac{1}{2}}$.
4. $D_{k+1} = \max\left(D_k, \frac{\gamma^{k+1}\|s^{k+1}\|^2 - \sum_{i\le k}\gamma^i D_i^2 \|g^i\|^2}{2\|s^{k+1}\|}\right)$.
5. Compute $x^{k+1} = P_X\left(x^k - \gamma^{k+1} s^{k+1}\right)$.

**Output:** best iterate $x_{best}$

# *Parameter-free methods*

**Algorithm 7** Distance over Weighted Gradients (DoWG)

**Parameters:** initial lower bound $d_0$ on $\|x^1 - x^*\|$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$

Initialize $v^0 = 0$.

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Select a subgradient $g^k \in \partial\bar{\mathcal{L}}(x^k)$.
2. Compute distance estimator $d_k = \max(d^{k-1}, \|x^k - x^1\|)$.
3. Compute weighted gradient sum $v^k = v^{k-1} + d_k^2\|g^k\|^2$.
4. Compute step size $\eta_k = \frac{d_k^2}{\sqrt{v^k}}$.
5. Compute step $x^{k+1} = P_X\left(x^k - \eta_k g^k\right)$.

**Output:** best iterate $x_{best}$

# *Smoothing technique*

Nonsmooth minimization with a subgradient: necessarily slow. Idea:

- Compute a smooth approximation of the problem (with Nesterov's technique)
- Apply accelerated gradient method to solve the approximation

With smoothing parameter $\varsigma$, we have the following approximations:

$$\mathcal{L}_{0,\varsigma}(\pi) = \min_{\mathbf{l}} \sum_{t\in\mathcal{T}} [C_{VOLL}(L_t - l_t) + \pi_t l_t + \frac{\varsigma}{2}\|l_t\|^2] \tag{9}$$

$$\text{subject to: } l_t \leq L_t \quad \forall t \in \mathcal{T}.$$

$$\mathcal{L}_{g,\varsigma}(\pi) = \min_{(\mathbf{p},\bar{\mathbf{p}},\mathbf{u},\mathbf{v},\mathbf{w})} \sum_{t\in\mathcal{T}} [C(u_t^g, v_t^g, p_t^g) - \pi_t p_t^g + \frac{\varsigma}{2}\|(\mathbf{p}, \bar{\mathbf{p}}, \mathbf{u}, \mathbf{v}, \mathbf{w})\|^2] \tag{10}$$

$$\text{subject to: } (p^g, \bar{p}^g, u^g, v^g, w^g) \in \text{conv}(\mathcal{P}^g) \qquad \forall t \in \mathcal{T}.$$

# *Smoothing technique*

## Lemma

*We have :*

$$\nabla \tilde{\mathcal{L}}_\varsigma = \left[ \sum_{g \in \mathcal{G}} (p^g_{\varsigma,*}) - l^*_\varsigma \right] \tag{9}$$

*where $p^g_{\varsigma,*}$ and $l^*_\varsigma$ are respectively the optimal values of the minimization problems (10) and (9), both unique by strong convexity of the objective functions.*

# *Smoothing-based method*

**Algorithm 8** Fast (Projected) Gradient Method (FGM) | Nesterov (2005)

**Parameters:** initial step length $\eta$, smoothing parameter $\varsigma$

**Inputs:** box $X = [\pi_{min}, \pi_{max}]^T$, initial iterate $x^1 \in X$ Initialize $y^1 = x^1$.

For $k = 1, 2, \ldots, N$ perform the following steps:

1. Compute $\nabla \tilde{\mathcal{L}}_\varsigma(y^k)$.
2. Compute $x^{k+1} = P_X\left(y^k - \eta \nabla \tilde{\mathcal{L}}_\varsigma(y^k)\right)$.
3. Compute $y^{k+1} = x^{k+1} + \frac{k-1}{k+2}(x^{k+1} - x^k)$.

**Output:** best iterate $x_{best}$

# *Talk Outline*

1. Quick introduction to electricity markets and CH pricing
2. Formulation of the CH pricing problem
3. First-order methods to compute CH prices
4. Numerical experiments
5. Conclusion

# *Experimental setup*

- Full Julia code using JuMP/Gurobi.
- 24 real-world instances:

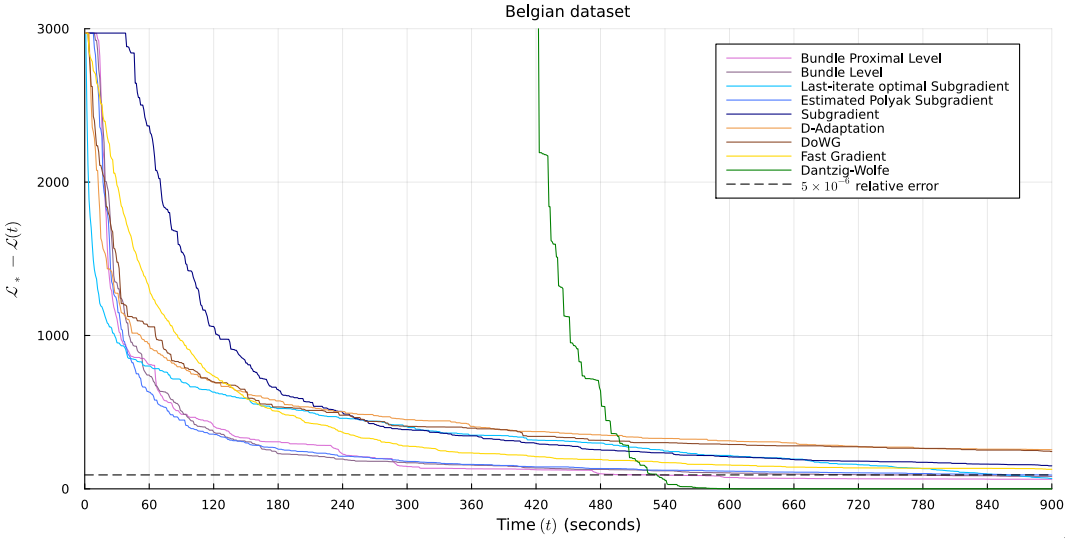
| Dataset name | # instances | # generators | # time periods |
|---|---|---|---|
| Belgian | 8 | 68 | 96 |
| Californian | 16 | 610 | 48 |

- Hyperparameter selection: all methods have 1 (critical) hyperparameter to tune. Benchmarks are computed after tuning all methods on one instance per dataset.
- We compute optimal solutions with $10^{-9}$ relative error using the Bundle Level Method.
- Stopping criterion is 15 minutes (time constraint for day-ahead markets)

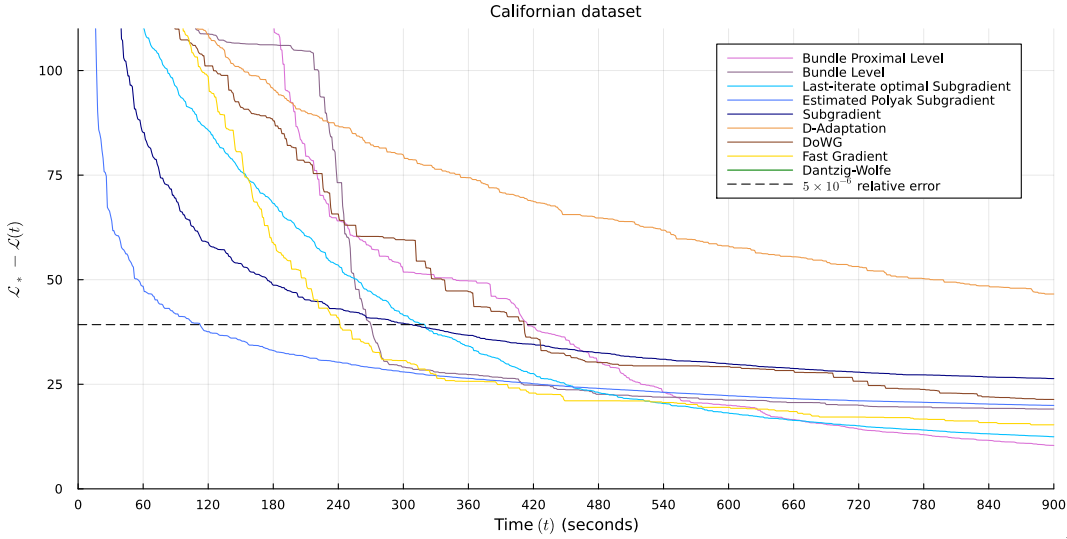
# *Simple heuristics*

- Warm start: we set the first iterate to be the dual variables of the continuous relaxation of the unit commitment problem.
  - Cheap initialization ($\sim$ one oracle call)
  - Interpretable: exactly the CH prices in the case where we only have minimum up/down times and constant start-up/shut-down costs.
- Averaging: we return the best iterate between (1) the price iterate which yielded the lowest error during the run and (2) the average of the 10% last iterates.
  - Inexpensive (one oracle call)
  - Can only decrease the objective function
  - In our benchmarks, we typically observe a relative error decrease in the order of $2 \times 10^{-6}$.

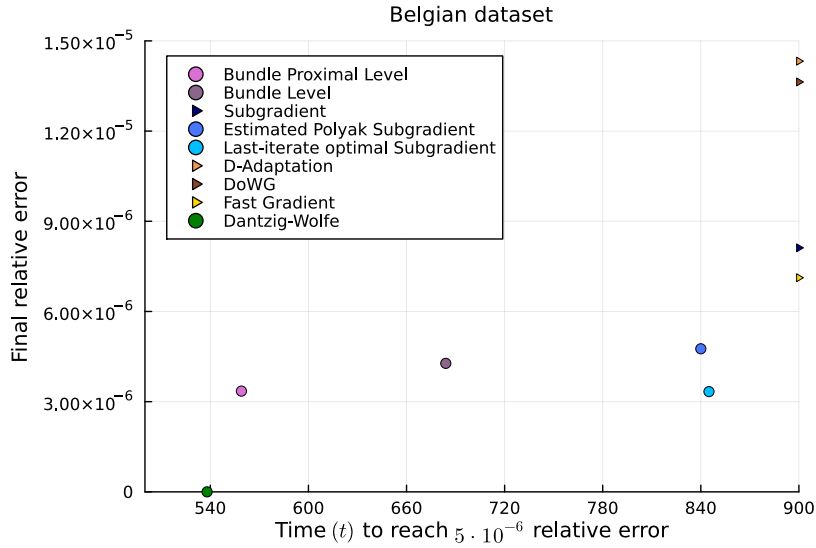
# *Benchmark on real-world datasets*



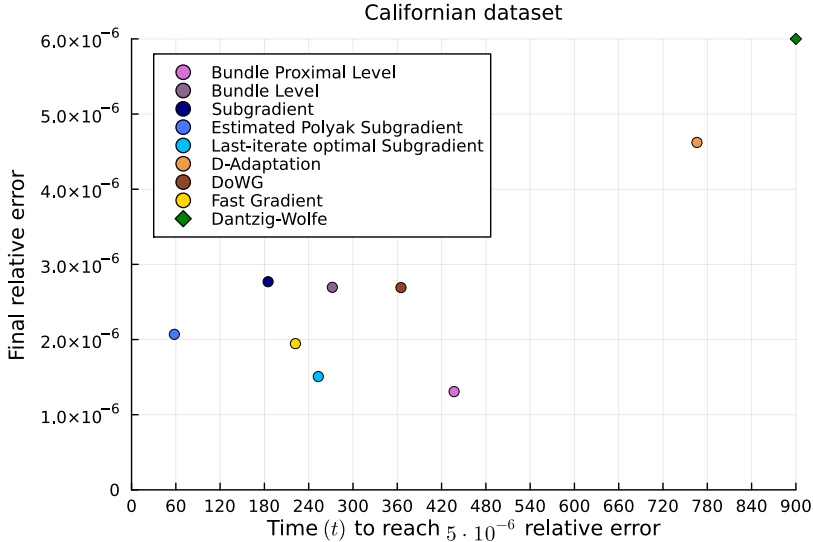
# Benchmark on real-world datasets



# *Benchmark on real-world datasets*



# *Benchmark on real-world datasets*



# *Talk Outline*

1. Quick introduction to electricity markets and CH pricing
2. Formulation of the CH pricing problem
3. First-order methods to compute CH prices
4. Numerical experiments
5. Conclusion

# *Conclusion*

Context:

- Computing CH prices is expensive (large MIP)
- Access to a first-order oracle is possible by solving many small MIPs
- We investigate the efficiency of (known) first-order methods for solving the CHP Lagrangian relaxation.

Contributions:

- Clear view on methods traditionaly less used for CH pricing
- Simple heuristics to improve accuracy for practicioners
- Open source Julia toolbox for practicioners and researchers to test methods and investigate a promising pricing rule for electricity markets

Code is available on sofianetanji.com/software

# *Related publications*


Boyd, S., Xiao, L., and Mutapcic, A. (2003). Subgradient methods. *lecture notes of EE392o, Stanford University, Autumn Quarter*, 2004:2004–2005.

Conforti, M., Cornuéjols, G., Zambelli, G., Conforti, M., Cornuéjols, G., and Zambelli, G. (2014). *Integer programming models*. Springer.

Defazio, A. and Mishchenko, K. (2023). Learning-rate-free learning by d-adaptation. In *Proceedings of the 40th International Conference on Machine Learning*.

Gribik, P. R., Hogan, W. W., Pope, S. L., et al. (2007). Market-clearing electricity prices and energy uplift. *Cambridge, MA*.

Hua, B. and Baldick, R. (2016). A convex primal formulation for convex hull pricing. *IEEE Transactions on Power Systems*, 32(5):3814–3823.

Khaled, A., Mishchenko, K., and Jin, C. (2023). Dowg unleashed: An efficient universal parameter-free gradient descent method.

Lemaréchal, C., Nemirovskii, A., and Nesterov, Y. (1995). New variants of bundle methods. *Mathematical Programming*.