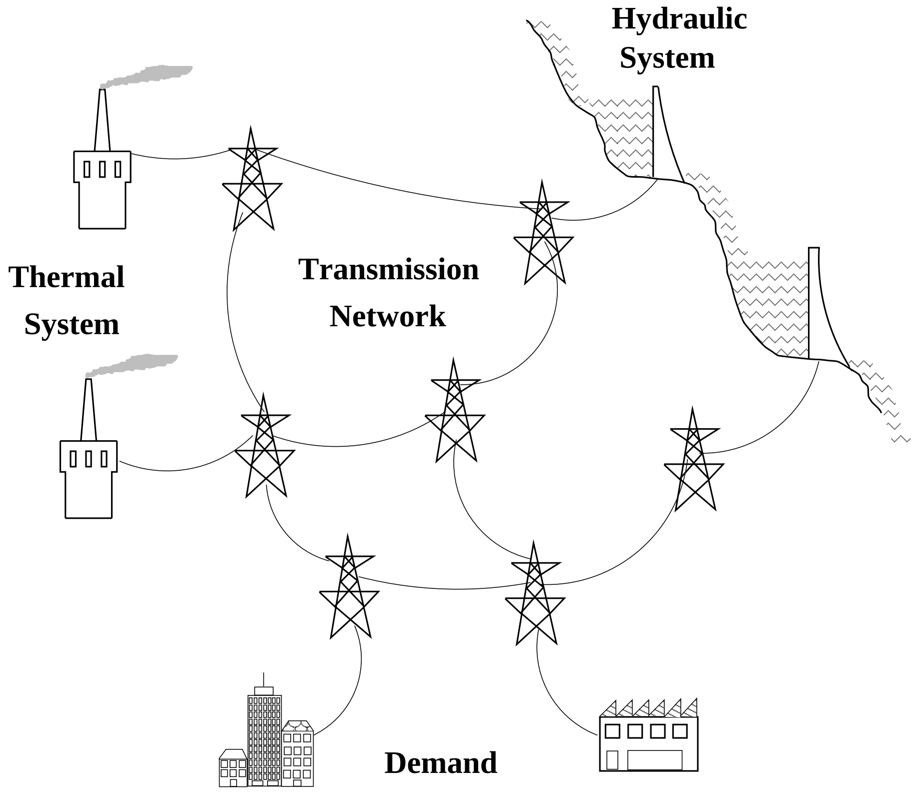
Generalized Unit Commitment

F. Javier Heredia
Dept. d'Estadística i Investigació Operativa
Universitat Poltècnica de Catalunya

César Beltran
Hautes Études Commerciales. HEC. Univ. de Genève

APMOD2004
Brunel University
London, june 2004

The generalized unit commitment problem



Given a forecasted power demand (1 up to 7 days), for each hour, we have to answer:

- Which thermal units will be generating power?
- How much power will they generate?
- How mach power will generate each reservoir?
- How will we distribute the power through the network?

Furthermore, we whish to generate at minimum cost and with a reliable service.

Formulation and variables

$$
(\text{GUC}) \begin{cases} \min & TC(p) + MC(p) + DC(q) \\ \text{s. t.} & (p, r_{_I}, r_{_D}, q, g, d, v, s) \in D_{htd} \\ & \qquad\qquad\qquad\qquad p \in D_m \end{cases}
$$

(Heredia and Nabona, 1995)

p_t^i : power generation of unit t during interval i.

$r_{_It}^{\,i}$: incremental spinning reserve of unit t at interval i.

$r_{_Dt}^{\,i}$: decremental spinning reserve of unit t at interval i.

q_l^i : power flow through line l during interval i.

g_r^i : hydro-generation of the reservoir r during interval i.

d_r^i : water discharge of reservoir r at interval i.

v_r^i : water volume of reservoir r at the end of interval i.

s_r^i : water spillage of reservoir r at interval i.

Objective function

$$TC(p) + MC(p) + DC(q)$$

- Thermal cost:

$$TC(p) := \sum_{t=1}^{n_t} \sum_{i=1}^{n_i} \left(c_{lt} p_t^i + c_{qt} (p_t^i)^2 \right). \qquad (5)$$

- Management cost:

$$MC(p) := \sum_{t=1}^{n_t} \sum_{i=1}^{n_i} MC_t(p_t^{i-1}, p_t^i), \qquad (6)$$

$$MC_t(p_t^{i-1}, p_t^i) := \begin{cases} C_{on\,t} + c_{b_t} & \text{if } p_t^{i-1} = 0 \text{ and } p_t^i > 0, \\ C_{off\,t} & \text{if } p_t^{i-1} > 0 \text{ and } p_t^i = 0, \\ c_{b_t} & \text{if } p_t^{i-1} > 0 \text{ and } p_t^i > 0, \\ 0 & \text{if } p_t^{i-1} = 0 \text{ and } p_t^i = 0. \end{cases} \qquad (7)$$

- Distribution cost.

$$DC(q) := \sum_{i=1}^{n_i} \pi^i \left(\sum_{l=1}^{n_l} r_l (q_l^i)^2 \right). \qquad (8)$$

Hydrothermal distribution domain D_{htd}

- Hydrothermal Transmission Extended (HTTE) network:

$$A_{\scriptscriptstyle HTTE} \cdot x \quad = \quad b_{\scriptscriptstyle HTTE}, \tag{9}$$

where $x = (p, r_{\scriptscriptstyle I}, r_{\scriptscriptstyle D}, q, d, v, s, g)$.

- Hydroelectric generation:

$$\sum_{r \in \mathcal{R}_j} h_r(d_r^i, v_r^{i-1}, v_r^i, s_r^i) = g_j^i \quad j = 1, \ldots, n_g\,,\; i = 1, \ldots, n_i. \tag{10}$$

- Incremental and decremental spinning reserve:

$$\sum_{t=1}^{n_t} {r_{\scriptscriptstyle I}}_t^i + \sum_{r=1}^{n_r} \left(\bar{g}_r^i - g_r^i \right) \quad \geq \quad {R_{\scriptscriptstyle I}}^i \quad i = 1, \ldots, n_i, \tag{11}$$

$$\sum_{t=1}^{n_t} {r_{\scriptscriptstyle D}}_t^i + \sum_{r=1}^{n_r} g_r^i \quad \geq \quad {R_{\scriptscriptstyle D}}^i \quad i = 1, \ldots, n_i. \tag{12}$$

- Kirchoff voltage law (d.c. model):

$$\sum_{l \in \mathcal{L}_j} x_l \cdot q_l^i = 0 \quad j = 1, \ldots, n_o\,,\; i = 1, \ldots, n_i. \tag{13}$$

- Upper and lower bounds.

Management domain D_m

- Minimum generating power:

$$p_t^i \in \{0\} \cup [\underline{p}_t^i, \overline{p}_t^i]. \tag{14}$$

- Minimum up time:

$$\begin{aligned} &\text{if } p_t^{i-1} = 0 \text{ and } p_t^i > 0, \text{ then } p_t^k > 0 \\ &(k = i, \ldots, i + min_{ont} - 1). \end{aligned} \tag{15}$$

- Minimum down time:

$$\begin{aligned} &\text{if } p_t^{i-1} > 0 \text{ and } p_t^i = 0, \text{ then } p_t^k = 0 \\ &(k = i, \ldots, i + min_{off_t} - 1). \end{aligned} \tag{16}$$

VD formulation of the GUC problem

- The primal GUC problem (VD version):

$$\left.\begin{array}{ll} \min & GC(p,\widetilde{p},q) \\ & := \frac{1}{2}TC(p) + DC(q) + \frac{1}{2}TC(\widetilde{p}) + MC(\widetilde{p}) \\ \text{s. t.} & (p,q,x) \in \mathcal{D}_{htd}, \quad \widetilde{p} \in \mathcal{D}_m, \quad p = \widetilde{p}. \end{array}\right\} \tag{22}$$

- The dual GUC problem (VD version):

$$\max_{\lambda \in R^{n_i \times n_t}} \left\{ \begin{array}{ll} \min & GC(p,\widetilde{p},q) + \lambda'(p - \widetilde{p}) \\ \text{s. t.} & (p,q,x) \in D_{htd}, \quad \widetilde{p} \in D_m. \end{array} \right\} \tag{23}$$

- The thermal subproblem for λ_n

$$\left.\begin{array}{ll} \min & \frac{1}{2}TC(\widetilde{p}) + MC(\widetilde{p}) - \lambda_n'\widetilde{p} \\ \text{s. t.} & \widetilde{p} \in D_m. \end{array}\right\} \tag{24}$$

- The hydrothermal subproblem for λ_n

$$\left.\begin{array}{ll} \min & \frac{1}{2}TC(p) + DC(q) + \lambda_n' p \\ \text{s. t.} & (p,q,x) \in D_{htd}. \end{array}\right\} \tag{25}$$

Thermal and hydrothermal subproblems

- The thermal subproblem for λ_n

$$\left.\begin{array}{ll} \min & \sum_{t=1}^{n_t} \left(\frac{1}{2}TC_t(\widetilde{p}_t) + MC_t(\widetilde{p}_t) - \lambda'_{n,t}\widetilde{p}_t\right) \\ \text{s. t.} & \widetilde{p}_t \in D_{m,t}. \end{array}\right\} \quad (26)$$

- n_t quadratic mixed integer programming problems of dimension n_i.

- Solved by forward dynamic programming.

- The hydrothermal subproblem for λ_n

$$\left.\begin{array}{ll} \min & \frac{1}{2}TC(p) + DC(q) + \lambda'_n p \\ \text{s. t.} & (p, q, x) \in D_{htd}. \end{array}\right\} \quad (27)$$

- OPF problem: large-scale quadratic network flow problem with side constraints.

- Solved by the specialized nonlinear network flow code NOXCB *(Heredia 1995)*.

Separating the augmented Lagrangian
(*Beltran and Heredia, JOTA 2002*)

- Primal VD problem:

$$\left.\begin{array}{ll} \text{min} & f(x) + \widetilde{f}(\widetilde{x}) \\ \text{s. t.} & x \in \mathcal{D},\ \ \widetilde{x} \in \widetilde{\mathcal{D}}, \\ & x - \widetilde{x} = 0. \end{array}\right\} \tag{32}$$

- (Augmented) VD dual problem:

$$\max_{\lambda \in R^n} \left\{ \min_{\substack{x \in \mathcal{D} \\ \widetilde{x} \in \widetilde{\mathcal{D}}}} f(x) + \widetilde{f}(\widetilde{x}) + \lambda'(x - \widetilde{x}) + \tfrac{c}{2}\|x - \widetilde{x}\|^2 \right\}. \tag{33}$$

- The <u>Auxiliary Problem Principle</u> (APP) *(Cohen, 1990)*, first time used to solve the GUC problem in *Batut and Renaud, 1992*:

$$\frac{c}{2}\|x-\widetilde{x}\|^2 \quad \rightsquigarrow \quad c(\mathbf{x_n}-\widetilde{\mathbf{x}}_\mathbf{n})'(x-\widetilde{x})+\frac{b}{2}\|(x,\widetilde{x})'-(\mathbf{x_n},\widetilde{\mathbf{x}}_\mathbf{n})'\|^2. \tag{34}$$

- The <u>Block Coordinate Descent</u> (BCD) or Non-linear Gauss-Seidel method, *(Bertsekas, 1995)*:

$$\min \frac{c}{2}\|x-\widetilde{x}\|^2 \quad \rightsquigarrow \quad \min \frac{c}{2}\|x-\widetilde{\mathbf{x}}_\mathbf{n}\|^2, \quad \min \frac{c}{2}\|\mathbf{x_{n+1}}-\widetilde{x}\|^2. \tag{35}$$

  Used to solve the GUC problem by Beltran and Heredia.

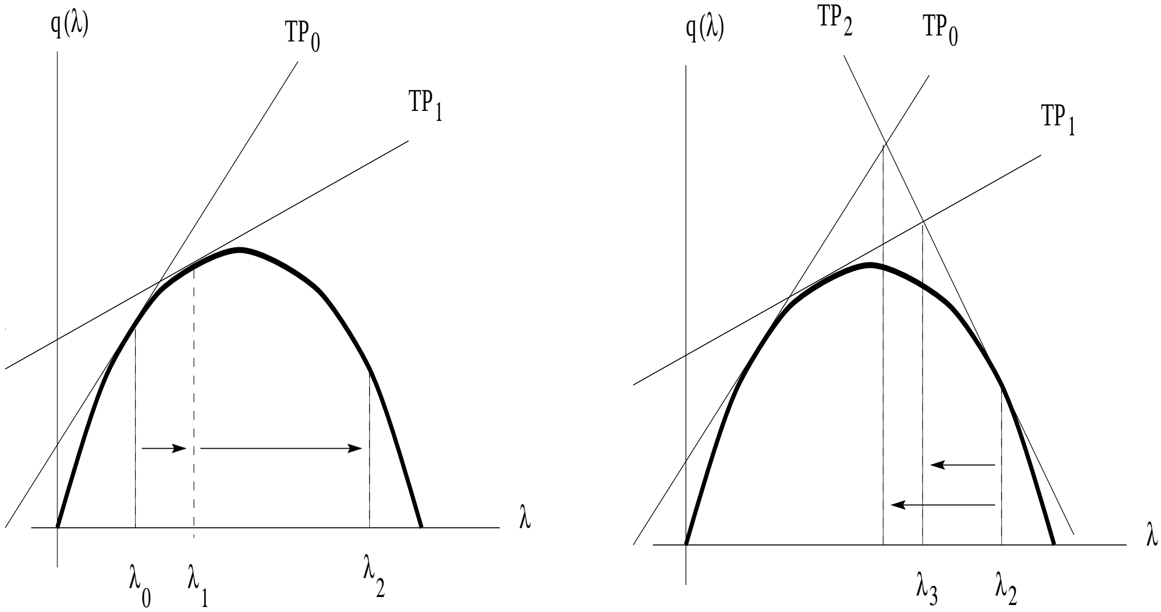
The radar subgradient method
(*Beltran and Heredia, JOTA 2005*)

- Subgradient method:

$$\lambda_{n+1} = \lambda_n + \alpha_n \cdot \frac{h(x_n)}{\|h(x_n)\|}, \tag{41}$$

$\lim_{n\to\infty} \alpha_n = 0, \quad \sum_{n=0}^{\infty} \alpha_n = +\infty.$

- Radar subgradient method (motivation):

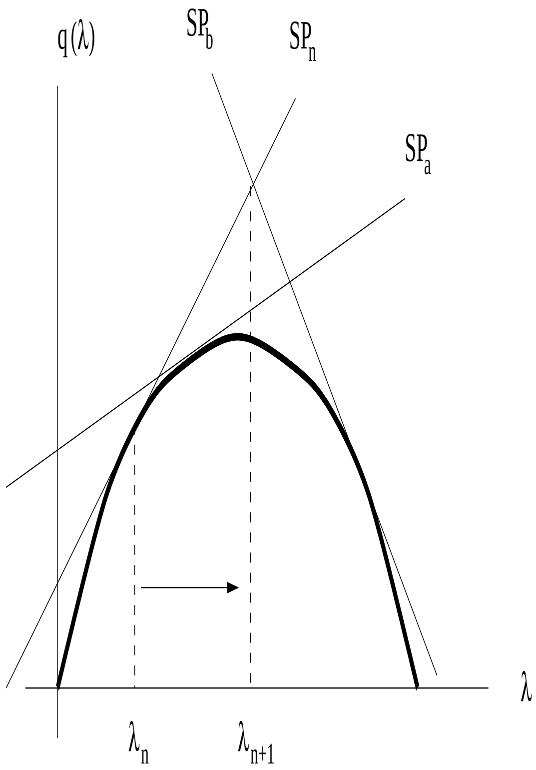


- Proposition 5.1 The step length β_n of the radar subgradient method can be computed as follows:

$$\beta_{n,k} := \frac{q_k - q_n + (\lambda_n - \lambda_k)' s_k}{(s_n - s_k)' s_n} \quad k = 0, \ldots, n-1, \tag{42}$$

$$\beta_n := \min\{\beta_{n,k} : \ \beta_{n,k} > 0 \ k = 0, \ldots, n-1\}. \tag{43}$$

The radar subgradient method (2)



Proposition 5.2 Let us define ($k = 0, \ldots, n-1$):

$SP_k \quad \equiv y_k(\lambda) = q_k + s_k'(\lambda - \lambda_k)$ supporting planes,

$\lambda_{n+1}(\beta) \quad := \lambda_n + \beta \cdot \frac{s_n}{\|s_n\|}$ line defined by the point λ_n and the subgradient s_n,

$y_k(\beta) \quad := q_k + s_k'(\lambda_{n+1}(\beta) - \lambda_k)$ line defined on the supporting plane SP_k when we move along the line $\lambda_{n+1}(\beta)$.

The slope of any supporting plane SP_k ($k = 0, \ldots, n-1$) along the line $\lambda_{n+1}(\beta)$ i.e. the slope of $y_k(\beta)$ is

$$m_k := \frac{s_k' s_n}{\|s_n\|} \tag{46}$$

The radar subgradient algorithm

Step 1 [Compute the subgradient s_n.]

Step 2 [Check the stopping criterion.]

Step 3 [Compute the radar step length.]

- Compute $s_n' s_n$
- For $k = 0, \ldots, n-1$
  * Compute $s_k' s_n$
  * If $s_k' s_n > 0$ reject the *unsuitable* plane SP_k. Otherwise compute

$$\beta_{n,k} := \frac{q_k - q_n + (\lambda_n - \lambda_k)' s_k}{s_n' s_n - s_k' s_n} \tag{47}$$

There are two cases depending on

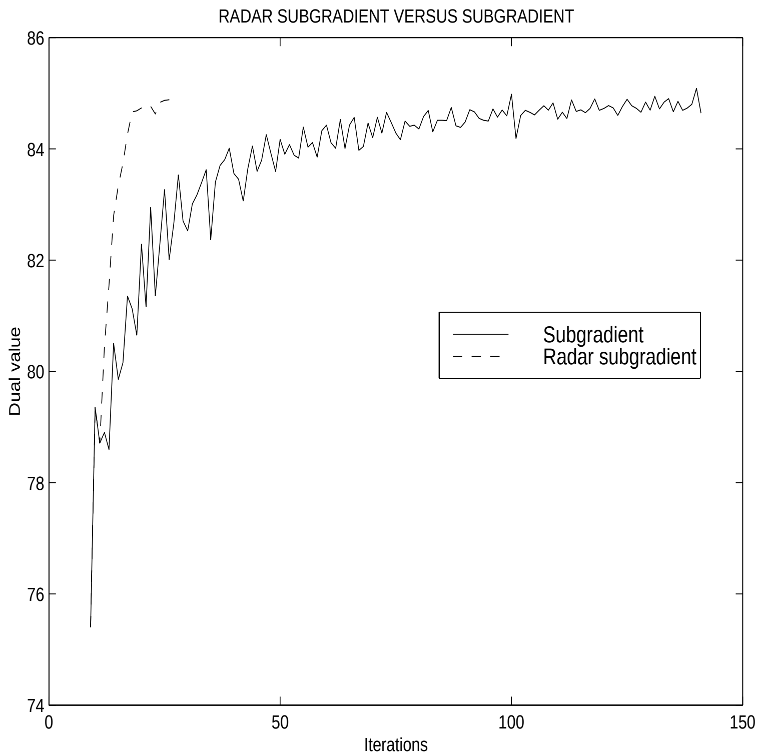
$$\Omega := \{\beta_{n,k} : \ \beta_{n,k} > 0 \ \ k = 0, \ldots, n-1\}. \tag{48}$$

a) If $\Omega \neq \emptyset$, then set $\beta_n := \min \Omega$.

b) If $\Omega = \emptyset$, then set $\beta_n = \frac{\alpha_n}{\|s_n\|}$, for a prefixed sequence $\{\alpha_n\}$ that guarantees subgradient convergence.

Step 4 Compute $\lambda_{n+1} = \lambda_n + \beta_n \cdot s_n$ and back to Step 1.

Radar subgradient vs. subgradient (1)



On average, the subgradient (SG) method performed 4.2 times slower than the radar subgradient (RS) method:

Case	Iterations		CPU Time (seconds)			Optimum	
	SG	RS	SG	RS	R.	SG	RS
10	141	28	3.0	0.9	3.3	84.6	84.9
20	154	28	6.0	1.4	4.3	170.3	169.8
30	179	38	10.7	2.8	3.8	255.6	256.2
40	189	31	16.8	3.4	4.9	341.0	340.8
50	178	34	21.8	5.2	4.2	426.3	427.1
60	190	32	32.2	7.1	4.5	511.7	512.0
70	198	36	45.3	10.5	4.3	597.3	597.7
80	198	33	59.6	13.1	4.5	682.3	682.6
90	198	32	77.9	17.1	4.6	767.6	767.9
100	201	42	103.2	27.3	3.8	853.4	854.1
Average	**183**	**33**	**37.7**	**8.9**	**4.2**	**469.0**	**469.3**

Radar subgradient vs. subgradient (2)

- Description of the UC instances:

Case	Int.	Res.	Thermal units	Cont. var.	Binary var.
1	2	0	2	16	4
2	6	2	4	138	24
3	48	2	4	1104	192
4	48	4	7	1920	336
5	48	2	7	1680	336
6	168	4	2	3360	336
7	168	4	7	6720	1176
8	168	4	11	9408	1848

- On average, the Subgradient (SG) method performed 2.17 times slower than the Radar Subgradient (RS) method:

Case	Iterations		CPU Time (seconds)			Cost ($\times 10^6$ PTA)	
	SG	RS	SG	RS	Ratio	SG	RS
1	15	39	6.5	16.0	0.41	0.1	0.1
2	200	30	60.0	9.8	6.12	5.7	5.8
3	40	25	25.7	16.3	1.58	0.9	0.9
4	200	37	169.7	72.4	2.34	6.3	6.3
5	42	18	31.5	15.7	2.01	1.0	1.0
6	88	28	150.0	78.6	1.91	4.4	4.3
7	49	28	632.0	467.0	1.35	2.5	2.5
8	100	40	25810.0	15479.0	1.67	84.8	84.8
Av.	**92**	**31**	**3360.7**	**2019.4**	**2.17**	**13.2**	**13.2**

Motivation of the radar multiplier (RM) method

- Problem to be solved:

$$\left.\begin{array}{ll} \text{min} & f(x) + \widetilde{f}(\widetilde{x}) \\ \text{s. t.} & x \in \mathcal{D},\ \widetilde{x} \in \widetilde{\mathcal{D}}, \\ & x - \widetilde{x} = 0. \end{array}\right\} \tag{49}$$

- In nonconvex optimization, the Classical Lagrangian Relaxation (CLR) may obtain primal infeasible solutions ($x - \widetilde{x} \neq 0$). By augmenting the Lagrangian, we can obtain a feasible solution ($x - \widetilde{x} = 0$).

- In nonconvex optimization, the Augmented Lagrangian Relaxation (ALR) may obtain a local optimizer.
  - The dual optimum obtained by the CLR method can be used to asses the ALR solution.
  - We propose a heuristic procedure to update the penalty parameter c_n for the augmented Lagrangian, which produces high quality local optimizers.

- We use the CLR combined with the ALR (RM method). In the first phase the CLR computes a dual cost bound and in the second phase the ALR computes a high quality local optimizer.

The radar multiplier algorithm

* [Objectives.] (1) To obtain a local optimizer of the following problem:

$$\left.\begin{array}{ll} \text{min} & f(x) + \widetilde{f}(\widetilde{x}) \\ \text{s. t.} & x \in \mathcal{D},\ \widetilde{x} \in \widetilde{\mathcal{D}}, \\ & x - \widetilde{x} = 0, \end{array}\right\} \tag{50}$$

and (2) to measure the optimizer quality through a dual lower bound.

Phase 1 [Compute a dual lower bound $\underline{f}^*$.]

Using the <u>radar</u> subgradient method solve the dual of problem (50),

$$\max_{\lambda \in R^n} \left\{ \min_{\substack{x \in \mathcal{D} \\ \widetilde{x} \in \widetilde{\mathcal{D}}}} f(x) + \widetilde{f}(\widetilde{x}) + \lambda'(x - \widetilde{x}) \right\}. \tag{51}$$

Phase 2 [Compute the local optimizer $(x^*, \widetilde{x}^*)$.]

Using the <u>multiplier</u> method (block coordinate descent version) solve the augmented dual of problem (50),

$$\max_{\lambda \in R^n} \left\{ \min_{\substack{x \in \mathcal{D} \\ \widetilde{x} \in \widetilde{\mathcal{D}}}} f(x) + \widetilde{f}(\widetilde{x}) + \lambda'(x - \widetilde{x}) + \frac{c}{2}\|x - \widetilde{x}\|^2 \right\}. \tag{52}$$

Radar multiplier vs. multiplier

- Description of the UC instances:

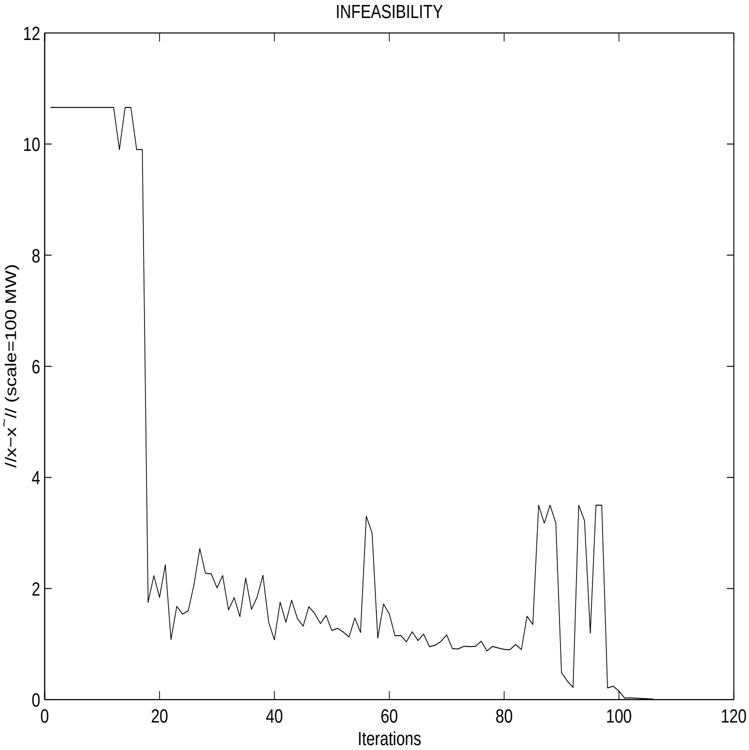
Case	Number interv.	Number reserv.	Thermal units	Cont. variables	Binary variables
1	2	0	2	16	4
2	6	2	4	138	24
3	48	2	4	1104	192
4	48	4	7	1920	336
5	48	2	7	1680	336
6	168	4	2	3360	336
7	168	4	7	6720	1176
8	168	4	11	9408	1848

- Unlike the Multiplier (M) method, the Radar Multiplier (RM) method computes a dual cost bound:

Case	CPU time (seconds)			Optimal cost ($\times 10^6 PTA$)		Cost bound
	M	RM	Ratio	M	RM	RM
1	5	16	2.8	0.004	0.004	0.004
2	28	31	1.1	7.015	6.776	5.826
3	29	45	1.5	0.991	0.990	0.961
4	63	98	1.5	7.134	6.358	6.324
5	34	42	1.2	1.028	1.028	1.004
6	71	120	1.6	4.471	4.467	4.330
7	553	627	1.1	3.223	2.653	2.530
8	699	15836	22.6	89.122	85.896	84.865
Av.	**185**	**2102**	**4.2**	**14.122**	**13.522**	**13.231**

Radar multiplier vs. radar subgradient

- Evolution of the infeasibility $\|x - \widetilde{x}\|_\infty$ (RM method):



- Unlike the Radar Subgradient (RS) method, the Radar Multiplier (RM) method computed feasible solutions ($x - \widetilde{x} \approx 0$):

Case	CPU time (seconds)			Infeasibility $\|x - \widetilde{x}\|_\infty$ (MW)	
	RS	RM	Ratio	RS	RM
1	16	16	1.01	0	0.0065
2	9	31	3.19	230	0.0004
3	16	45	2.76	360	0.0072
4	72	98	1.36	1180	0.0032
5	15	42	2.69	470	0.0077
6	78	120	1.53	1390	0.0084
7	467	627	1.34	530	0.0067
8	15479	16019	1.03	3350	0.0040
Av.	**2019**	**2124**	**1.86**	**939**	**0.0055**

The MACH code

Phase 1 [Compute a dual lower bound $\underline{f}^*$.]

Step 1.1 [Hydrothermal distribution subproblem.] (NOXCB, nonlinear network flow solver).

$$\min_{x \in \mathcal{D}} \ f(x) + \lambda_n' x.$$

Step 1.2 [Thermal subproblem.] (Dynamic programming).

$$\min_{x \in \widetilde{\mathcal{D}}} \ \widetilde{f}(\widetilde{x}) - \lambda_n' \widetilde{x}.$$

Step 1.3 [Dual variable updating.] (Radar subgradient). If the stopping criterion is not fulfilled, back to step 1.1.

Phase 2 [Compute $(x^*, \widetilde{x}^*)$ a local LPF optimizer.]

Step 2.1 [Hydrothermal distribution subproblem.] (NOXCB, nonlinear network flow solver).

$$\min_{x \in \mathcal{D}} \ f(x) + \lambda_n' x + \tfrac{c_n}{2} \|x - \widetilde{x}_n\|^2.$$

Step 2.2 [Thermal subproblem.] (Dynamic programming).

$$\min_{x \in \widetilde{\mathcal{D}}} \ \widetilde{f}(\widetilde{x}) - \lambda_n' \widetilde{x} + \tfrac{c_n}{2} \|x_{n+1} - \widetilde{x}\|^2.$$

Step 2.3 [Dual variable updating.] (Multiplier method). If the stopping criterion is not fulfilled, back to step 2.1.

Testing MACH

- Description of the GUC instances:

Case	Int.	Res.	Units	Buses	Lines	Cont.	Bin.
1	2	0	2	3	3	28	4
2	6	2	4	3	6	192	24
3	24	20	70	6	10	8568	1680
$*4$	48	10	70	6	10	15696	3360
5	48	0	27	6	21	6528	1296
6	48	2	4	3	6	1536	192
7	48	4	4	2	3	1632	192
8	48	2	7	2	3	1920	336

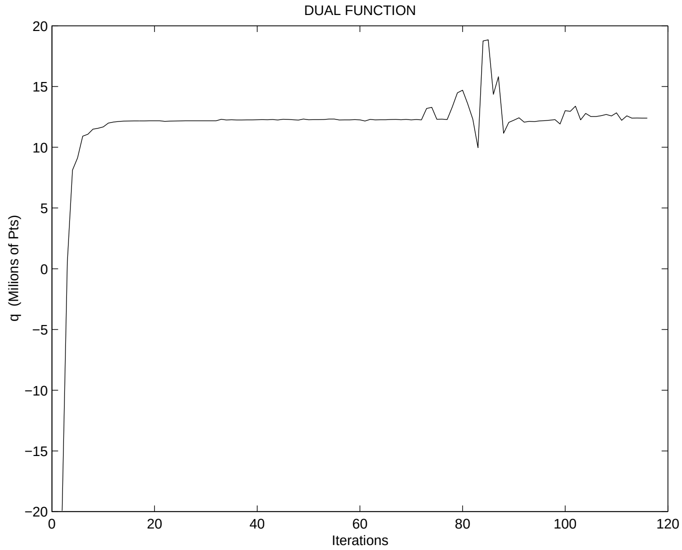
- Computational results:

Case	Iterations		Time (sec.)	Dual bound	Opt. cost ($\times 10^6$ PTA)	Duality gap (%)
	I	II				
1	41	0	19	0.005	0.005	0.00
2	31	280	134	9.038	10.006	10.71
3	32	64	3604	5.522	5.590	1.23
$*4$	32	84	6282	12.183	12.397	1.76
5	32	167	3076	8.713	8.890	2.03
6	17	185	161	1.396	1.404	0.57
7	34	9	163	7.181	7.745	7.85
8	17	77	67	1.193	1.232	3.27
Av.	**30**	**108**	**1688**	**5.654**	**5.909**	**3.43**

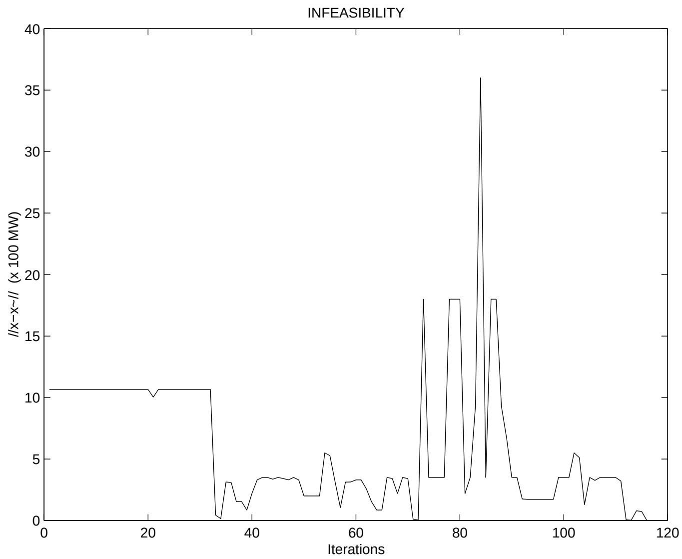
A real-life large-scale GUC instance (1)

Case	Int.	Res.	Units	Buses	Lines	Cont.	Bin.
$*4$	48	10	70	6	10	15696	3360

- Evolution of the dual function:

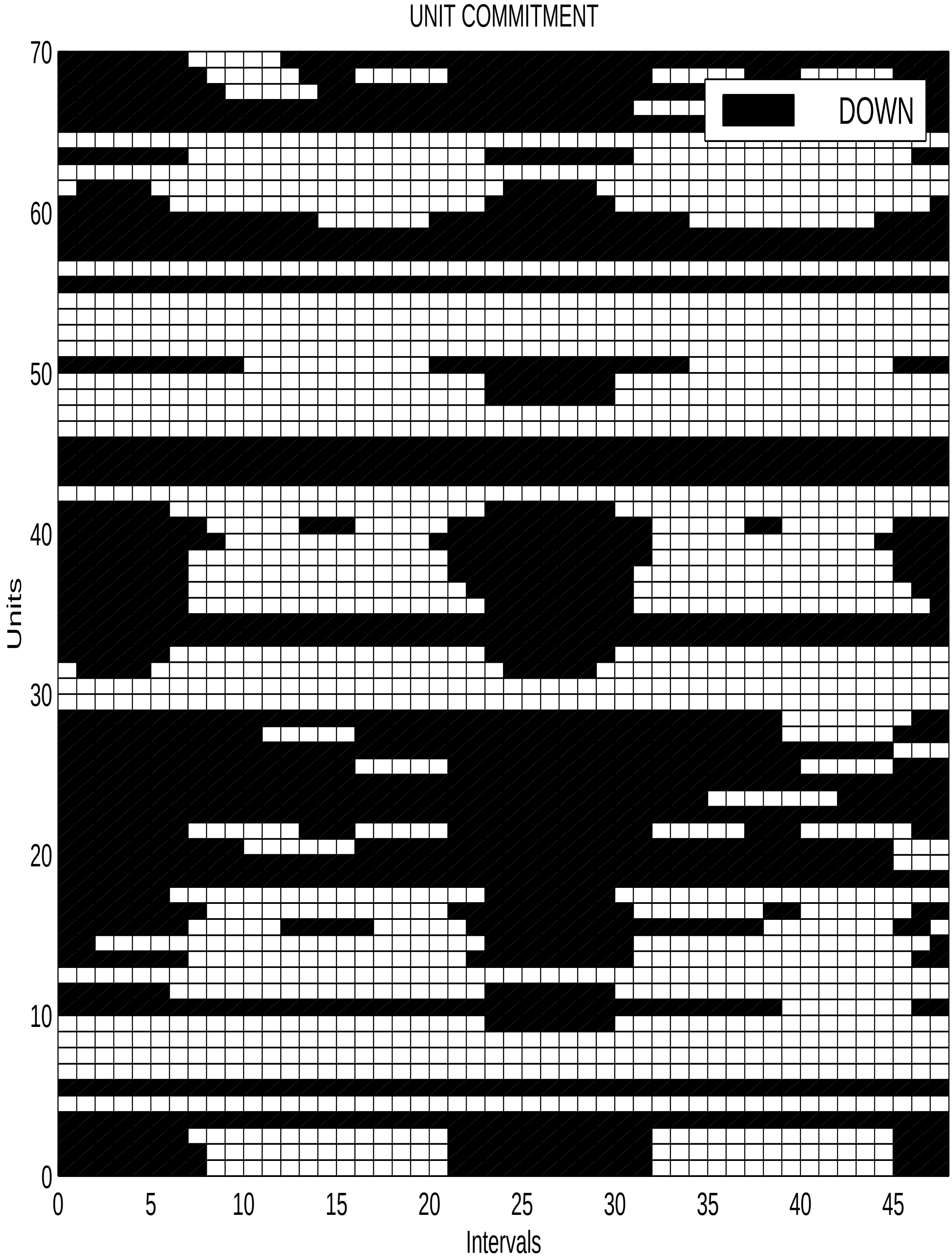


- Evolution of the infeasibility:



Iter.	Time (h.)	Dual bound	Opt. cost ($\times 10^6$ PTA)	Duality gap (%)	$\|x^* - \widetilde{x}^*\|_\infty$ (MW)
32+84	1.75	12.183	12.397	1.76	0.86

A real-life large-scale GUC instance (2)



A real-life large-scale GUC instance (3)