7th International Conference on
"Energy, Sustainability and Climate Change" ESCC 2020

Quadratic and linear ED problems and their AMPL implementations

O. Pichugina, P. Stetsyuk, O. Butkevych, O. Lykhovyd

V. M. Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine,
Institute of Electrodynamics of National Academy of Sciences of Ukraine
National Aerospace University «Kharkiv Aviation Institute»
(Ukraine)

25.08.2020

Introduction

Economic dispatch problem (**EDP**) is one of the fundamental optimization problems in power system operation, which is used to determine the optimal combination of power outputs of all power units to minimize the total fuel cost while satisfying various constraints over the entire dispatch periods.

Notations

i	generator (power unit) number
t	time period
n	total number of generators
T	total number of time periods
$f_i(x)$	fuel cost function of a generator i
$f(x)$	total fuel cost function
E_t	total long demand during the time interval t
P_i^{low}, P_i^{up}	output limits of a generator i
u_i, d_i	ramp-up and ramp-down rate limits of a generator i
variable $x_{i,t}$	power output of a generator i in a time interval t

ED problem (EDP): mathematical model

Goal is to minimize total fuel cost:

$$f^* = f(x^*) = \min \left\{ f(x) = \sum_{t=1}^{T} \sum_{i=1}^{n} f_i\left(x_{i,t}\right) \right\} \tag{1}$$

subject to:

1. power balance constraints

$$\sum_{i=1}^{n} x_{i,t} = E_t, \quad t \in J_T; \tag{2}$$

2. power generating limits

$$P_i^{low} \le x_{i,t} \le P_i^{up}, \quad i \in J_n, \quad t \in J_T; \tag{3}$$

3. Generator's ramp-up and ramp-down rate constraints

$$x_{i,t+1} - x_{i,t} \le U_i,\ i \in N_R \subseteq J_n,\ t \in J_{T-1}; \tag{4}$$

$$x_{i,t} - x_{i,t+1} \le D_i,\ i \in N_R \subseteq J_n,\ t \in J_{T-1}. \tag{5}$$

EDP: typology

1. Depending on type of generator's cost functions $f_i(x)$, $i \in J_n$:

- The linear EDP (LEDP): $f_i\left(x\right) = b_i x + c_i,\ \ i \in J_n$;
- the quadratic EDP (QEDP): $f_i\left(x\right) = a_i x^2 + b_i x + c_i,\ \ i \in J_n,\ \sum_{i=1}^{n} |a_i| > 0$;
- the cubic EDP: $f_i\left(x\right) = a_i x^3 + b_i x + c_i,\ \ i \in J_n,\ \sum_{i=1}^{n} |a_i| > 0$;
- the EDP considering the valve-point effect[1]:

$$f_i\left(x\right) = a_i x^2 + b_i x + c_i + \left| d_i \sin\left(\omega_i\left(P_i^{low} - x_{i,t}\right)\right)\right|,\ \ i \in J_n.$$

2. Depending on type of the total fuel cost function $f(x)$:

- convex/nonconvex EDP;
- smooth/nonsmooth EDP, etc.;

3. Depending on the number of objective functions:

- single objective;
- multiobjective.

4. Depending on present constraints;

5. Depending on the presence of integer variables.

[1] a_i, b_i, c_i are the fuel cost coefficients of a generator i; d_i, ω_i – coefficients of a generator i reflecting valve-point effect

Methods to solve EDP

linear programming (LP)	improved differential evolution (IDE)
mixed integer programming (MIP)	quantum-inspired evolutionary algorithm
dynamic programming (DP)	artificial bee colony (ABC)
Lagrangian relaxation	integer coded genetic algorithm (ICGA)
Shor's r-algorithm	bacterial foraging (BF)
differential evolution (DE)	imperialistic competition algorithm (ICA)
simulated annealing (SA)	harmony search algorithm (HSA)
evolutionary programming (EP)	particle swarm optimization (LR-PSO)
artificial immune system (AIS)	straightforward (SF)
Tabu search (TS)	extended priority list (EPL)
colony optimization (ACO)	advanced three stage approach (ATHS)
improved particle swarm optimization	

Goals

Our problem is (**Problem A**):

$$f^* = f(x^*) = \min \left\{ f(x) = \sum_{t=1}^{T} \sum_{i=1}^{n} (a_i x_{i,t}^2 + b_i x_{i,t} + c_i) \right\} \qquad (6)$$

$$a \geq 0.$$

subject to (2)-(5).

Problem A covers:

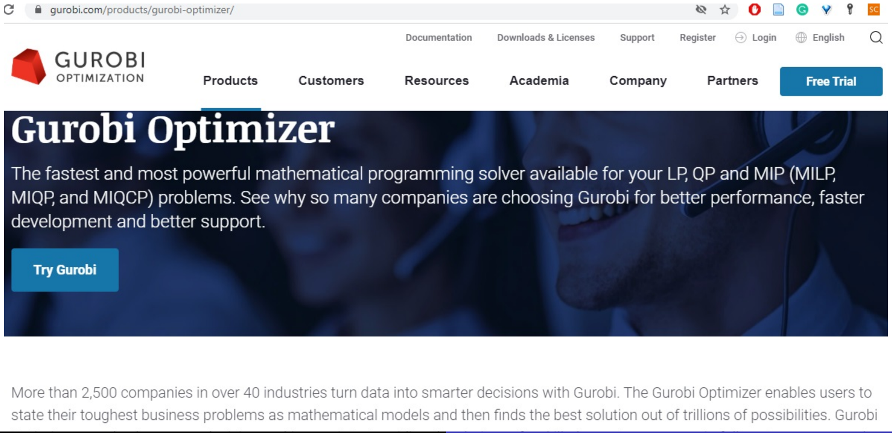
- the LEDP if $a = 0$;
- the convex QEPD if $a \geq 0$, $a \neq 0$.

Main tasks:

- exploration of the uniqueness of EDP-solution and dealing with multiple EDP-solutions;
- implementing Gurobi (gurobi.com) to EDP;
- implementing IMPL (https://ampl.com/) to EDP.

Gurobi

The **Gurobi** Optimizer is optimization solver for continuous optimization (**linear programming (LP), quadratic programming (QP)**, quadratically constrained programming (QCP)) for mixed integer optimization (mixed integer linear programming (MILP), mixed-integer quadratic programming (MIQP), and mixed-integer quadratically constrained programming (MIQCP)).



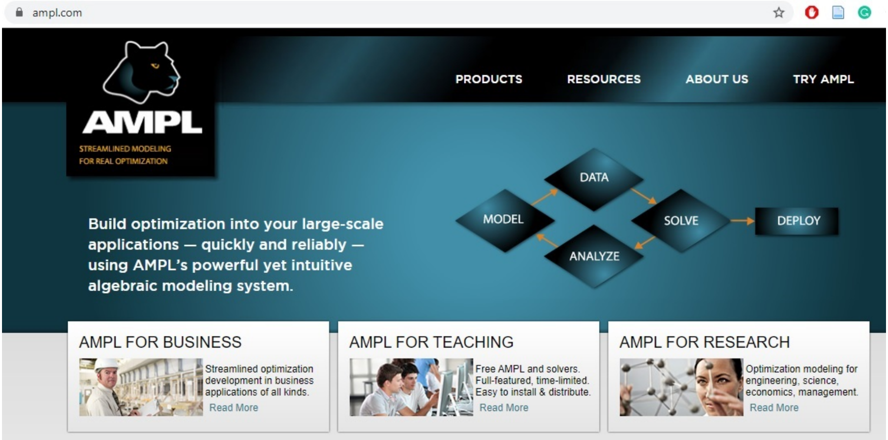
Gurobi Optimizer

The fastest and most powerful mathematical programming solver available for your LP, QP and MIP (MILP, MIQP, and MIQCP) problems. See why so many companies are choosing Gurobi for better performance, faster development and better support.

More than 2,500 companies in over 40 industries turn data into smarter decisions with Gurobi. The Gurobi Optimizer enables users to state their toughest business problems as mathematical models and then finds the best solution out of trillions of possibilities. Gurobi

AMPL

AMPL (A Mathematical Programming Language) is an algebraic modeling language.



ampl.com

Build optimization into your large-scale applications — quickly and reliably — using AMPL's powerful yet intuitive algebraic modeling system.

AMPL FOR BUSINESS

Streamlined optimization development in business applications of all kinds.

Read More

AMPL FOR TEACHING

Free AMPL and solvers. Full-featured, time-limited. Easy to install & distribute.

Read More

AMPL FOR RESEARCH

Optimization modeling for engineering, science, economics, management.

Read More

Problem A investigation: QEDP

Let

$$a_i > 0, i \in J_n. \tag{7}$$

To the QEDP (2)-(7), we will refer to as **Problem A1**.

Theorem 1.1.

If constraints (2)-(5) *are consistent, then Problem A1 has a unique solution* x^*.

Remark 1.

If $\exists\, i \in J_n :\ a_i \not> 0$, *the target function is not strictly convex and uniqueness of solution to Problem A,* (6) *is not guaranteed.*

Problem A investigation: LEDP

$$f^* = f(x^*) = \min \left\{ f(x) = \sum_{t=1}^{T} \sum_{i=1}^{n} (b_i x_{i,t} + c_i) \right\}. \tag{8}$$

Consider the general linear program

$$z^* = h(x^*) = \min \left\{ h(x) = c^T x \right\} \tag{9}$$

subject to

$$\begin{aligned} Ax &= b, \\ Cx &\geq d, \end{aligned} \tag{10}$$

where $c, x \in R^n, A \in R^{m \times n}, C = (c_i)_{i \in J_k} \in R^{k \times n}, b \in R^m, d \in R^k$.

Its dual is

$$z^* = g(u^*, v^*) = \max \left\{ g(u, v) = b^T u + d^T v \right\} \tag{11}$$

subject to

$$\begin{aligned} & A^T u + B^T v = c \\ & v \geq 0. \end{aligned} \tag{12}$$

Problem A investigation: LEDP

Theorem 1.2.

[a] *The linear program* (9), (10) *has a unique solution* x^* *iff:*

- *a matrix* $\left[A^T \overline{C}^T\right]$ *has no linearly dependent rows, where*

$$\overline{C} = (c_i)_{i \in I}\,,\ I{=}\{i \in J_k : c_i x^* = d_i\}\,;$$

- *there is no* x *satisfying*

$$\begin{array}{rl} & Ax = 0,\, C'x = 0,\, C''x \geq 0, \\ \textit{where} & C' = (c_i)_{i \in I'}\,,\ I'{=}\{i \in I : v_i > 0\}\,; \\ & C'' = (c_i)_{i \in I'}\,,\ I''{=}\{i \in I : v_i = 0\}\,. \end{array} \tag{13}$$

[a] Mangasarian, O.L.: Uniqueness of solution in linear programming. Linear Algebra and its Applications. 25, 151–162 (1979).

This means that the linear program (9),(10) has a unique solution x^* iff

$$rank\left[A^T \overline{C}^T\right] = n$$

(**Condition 1**), and the constraints (13) are inconsistent (**Condition 2**) .

Dealing with LEDP multiple solutions

LEDP is replaced by the following convex optimization problem (**Problem A2.p**):

$$q^*(p) = q(y^*, p) = \min\left\{ q(x,p) = \sum_{t=1}^{T}\sum_{i=1}^{N} |x_{i,t} - x_{i,t}^0|^p \right\}$$

subject to the standard EDP-constraints (2)-(5) and an additional constraint

$$\sum_{t=1}^{T}\sum_{i=1}^{N}(b_i x_{i,t} + c_i) \le f^*, \tag{14}$$

where $1 \le p \le 2$, $\{x_{i,t}^0\}_{i,t}$ is a target solution to the LEDP.

Theorem 1.3.

1. For $1 \le p \le 2$, the Problem A2.p has a solution iff the correspondent LEDP has a solution.
2. If $1 < p \le 2$, the Problem A2.p has a unique solution $y^* = x^*$.

Remark 2.

Problem A 2.p is intended to single out a solution other than x^* from LEDP-multiple solutions.

Solving Problem A 2.p by Gurobi

Problems A 2.1 and A 2.2 are solvable by Gurobi. For instance, the objective function in Problem A 2.2

$$q(x,2) = \sum_{t=1}^{T} \sum_{i=1}^{N} (x_{i,t} - x_{i,t}^{0})^2$$

is quadratic and additional constraints are linear.

In Problem A 2.1, the objective function is

$$q(x,1) = \sum_{t=1}^{T} \sum_{i=1}^{N} |x_{i,t} - x_{i,t}^{0}| \; - \; \text{piece-wise linear.}$$

To apply Gurobi, a lifting into a higher dimensional space can be applied. For that, each expression $|x_{i,t} - x_{i,t}^{0}|$ is replaced by $z_{i,t} \geq 0$, where $-z_{i,t} \leq x_{i,t} - x_{i,t}^{0} \leq z_{i,t}$. As a result, we come to a new linear program solvable by Gurobi:

$$q^{*}(1) = \min \left\{ \sum_{t=1}^{T} \sum_{i=1}^{N} z_{i,t} \right\}$$

subject to constraints (2)-(5) and

$$z_{i,t} \geq 0, \; -z_{i,t} \leq x_{i,t} - x_{i,t}^{0} \leq z_{i,t}, \; i \in J_n, \; t \in J_T.$$

* We solve Problem A 2.2 in order to single out a single EDP-solution.

Data: Example 1. Problem A.1; Example 2. LEDP, Problem A2.2 ($a = 0$, $x^0 = 0$)

Input: $n = 10$, $T = 24$

unit	a_i $\$/\mathrm{MW}^2\mathrm{h}$	b_i $\$/\mathrm{MWh}$	c_i $\$/\mathrm{h}$	P_i^{low} MW	P_i^{up} MW	U_i MW	D_i MW
1	0.00043	21.6	958.2	150	470	80	80
2	0.00063	21.05	1313.6	135	460	80	80
3	0.00039	20.81	604.97	73	340	80	80
4	0.0007	23.9	471.6	60	300	50	50
5	0.00079	21.62	480.29	73	243	50	50
6	0.00056	17.87	601.75	57	160	50	50
7	0.00211	16.51	502.7	20	130	30	30
8	0.0048	23.23	639.4	47	120	30	30
9	0.10908	19.58	455.6	20	80	30	30
10	0.00951	22.54	692.4	55	55	30	30

t	1	2	3	4	5	6	7	8	9	10	11	12
E_t	1036	1110	1258	1406	1480	1628	1702	1776	1924	2072	2146	2220
t	13	14	15	16	17	18	19	20	21	22	23	24
E_t	2072	1924	1776	1554	1480	1628	1776	2072	1924	1628	1332	1184

Solution: Example 1

Problem A1 solution:

t	$x^*_{\mathbf{1},t}$	$x^*_{\mathbf{2},t}$	$x^*_{\mathbf{3},t}$	$x^*_{\mathbf{4},t}$	$x^*_{\mathbf{5},t}$	$x^*_{\mathbf{6},t}$	$x^*_{\mathbf{7},t}$	$x^*_{\mathbf{8},t}$	$x^*_{\mathbf{9},t}$	$x^*_{\mathbf{10},t}$
1	150.0	135.0	206.0	60.0	73.0	160.0	130.0	47.0	20.0	55.0
2	150.0	156.8	258.2	60.0	73.0	160.0	130.0	47.0	20.0	55.0
3	150.0	236.8	326.2	60.0	73.0	160.0	130.0	47.0	20.0	55.0
4	187.7	316.8	340.0	60.0	89.5	160.0	130.0	47.0	20.0	55.0
5	183.8	396.8	340.0	60.0	87.4	160.0	130.0	47.0	20.0	55.0
6	238.7	460.0	340.0	60.0	117.3	160.0	130.0	47.0	20.0	55.0
7	286.6	460.0	340.0	60.0	143.4	160.0	130.0	47.0	20.0	55.0
8	348.1	442.3	340.0	60.0	173.6	160.0	130.0	47.0	20.0	55.0
9	428.1	460.0	340.0	60.0	223.6	160.0	130.0	47.3	20.0	55.0
10	470.0	460.0	340.0	110.0	243.0	160.0	130.0	77.3	26.7	55.0
11	470.0	460.0	340.0	160.0	243.0	160.0	130.0	106.6	21.4	55.0
12	470.0	460.0	340.0	201.5	243.0	160.0	130.0	120.0	40.5	55.0
13	439.1	460.0	340.0	151.5	226.4	160.0	130.0	90.0	20.0	55.0
14	395.1	460.0	340.0	101.5	202.4	160.0	130.0	60.0	20.0	55.0
15	332.5	460.0	340.0	60.0	171.5	160.0	130.0	47.0	20.0	55.0
16	252.5	380.0	328.0	60.0	121.5	160.0	130.0	47.0	20.0	55.0
17	230.0	345.0	340.0	60.0	93.0	160.0	130.0	47.0	20.0	55.0
18	310.0	363.0	340.0	60.0	143.0	160.0	130.0	47.0	20.0	55.0
19	390.0	381.0	340.0	60.0	193.0	160.0	130.0	47.0	20.0	55.0
20	470.0	460.0	340.0	110.0	243.0	160.0	130.0	77.0	27.0	55.0
21	390.0	460.0	340.0	66.3	223.0	160.0	130.0	79.5	20.2	55.0
22	310.0	380.0	287.0	60.0	173.0	160.0	130.0	53.0	20.0	55.0
23	230.0	300.0	207.0	60.0	123.0	160.0	130.0	47.0	20.0	55.0
24	150.0	220.0	269.0	60.0	73.0	160.0	130.0	47.0	20.0	55.0

The minimum total fuel cost $f^* = 1002055.51$

Elapsed time $\tau = 0.030812$ *sec*

Solution: Example 2

LEDP solution:

t	$x^*_{1,t}$	$x^*_{2,t}$	$x^*_{3,t}$	$x^*_{4,t}$	$x^*_{5,t}$	$x^*_{6,t}$	$x^*_{7,t}$	$x^*_{8,t}$	$x^*_{9,t}$	$x^*_{10,t}$
1	150	135	146	60	73	160	130	47	80	55
2	150	151	204	60	73	160	130	47	80	55
3	150	231	272	60	73	160	130	47	80	55
4	150	311	340	60	73	160	130	47	80	55
5	150	385	340	60	73	160	130	47	80	55
6	223	460	340	60	73	160	130	47	80	55
7	277	460	340	60	93	160	130	47	80	55
8	319	442	340	60	143	160	130	47	80	55
9	399	460	340	60	193	160	130	47	80	55
10	470	460	340	62	243	160	130	72	80	55
11	470	460	340	112	243	160	130	96	80	55
12	470	460	340	162	243	160	130	120	80	55
13	452	460	340	112	193	160	130	90	80	55
14	404	460	340	62	173	160	130	60	80	55
15	324	457	340	60	123	160	130	47	80	55
16	244	377	328	60	73	160	130	47	80	55
17	230	297	331	60	90	160	130	47	80	55
18	310	306	340	60	140	160	130	47	80	55
19	390	380	284	60	190	160	130	47	80	55
20	470	460	340	60	240	160	130	77	80	55
21	390	460	340	60	190	160	130	59	80	55
22	310	380	266	60	140	160	130	47	80	55
23	230	300	186	60	90	160	130	47	74	55
24	150	220	209	60	73	160	130	47	80	55

The minimum total fuel cost $f^* = 990172.63$

Elapsed time $\tau = 0.048868$ *sec*

Data: Example 3. LEDP, Problem A2.2 ($x^0 = 0$)

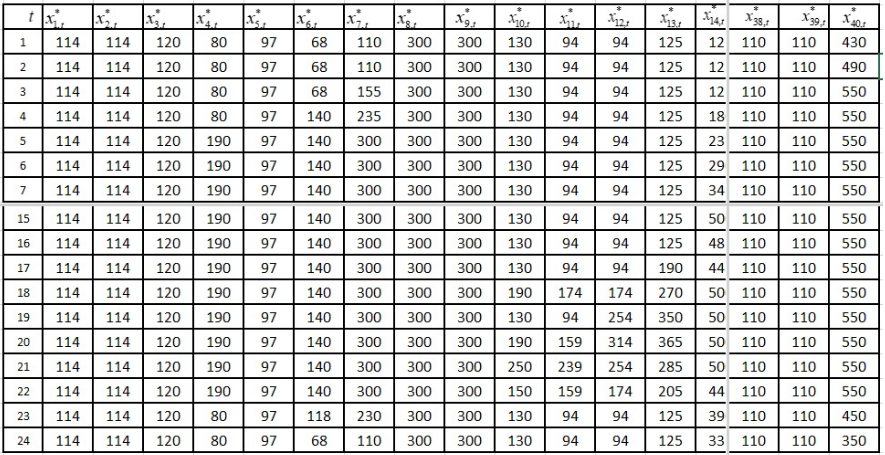
Input: $n = 40, \ T = 24$

unit	a_i \$/MW^{2}h	b_i \$/MWh	c_i \$/h	P_i^{low} MW	P_i^{up} MW	U_i MW	D_i MW
1	0,0069	6,73	94,705	36	114	80	120
2	0,0069	6,73	94,705	36	114	80	120
3	0,02028	7,07	309,54	60	120	130	130
4	0,00942	8,18	369,54	80	190	130	130
5	0,0114	5,35	369,03	47	97	80	120
6	0,01142	8,05	148,89	68	140	80	120
7	0,01142	8,03	222,33	110	300	80	120
8	0,00357	6,99	287,71	135	300	65	100
...	...	...	...	...	...	...	...
39	0,0161	3,33	307,45	25	110	60	100
40	0,00313	7,97	647,83	242	550	60	100

t	1	2	3	4	5	6	7	8	...	12
E_t	9000	9200	9600	10000	10400	10800	11000	11200	...	11600
t	13	14	15	16	17	18	19	20	...	24
E_t	11500	11400	11200	11200	11400	11980	12000	12200	...	9000

Solution: Example 3 LEDP

LEDP solution x^*:



t	$x^*_{1,t}$	$x^*_{2,t}$	$x^*_{3,t}$	$x^*_{4,t}$	$x^*_{5,t}$	$x^*_{6,t}$	$x^*_{7,t}$	$x^*_{8,t}$	$x^*_{9,t}$	$x^*_{10,t}$	$x^*_{11,t}$	$x^*_{12,t}$	$x^*_{13,t}$	$x^*_{14,t}$	$x^*_{38,t}$	$x^*_{39,t}$	$x^*_{40,t}$
1	114	114	120	80	97	68	110	300	300	130	94	94	125	12	110	110	430
2	114	114	120	80	97	68	110	300	300	130	94	94	125	12	110	110	490
3	114	114	120	80	97	68	155	300	300	130	94	94	125	12	110	110	550
4	114	114	120	80	97	140	235	300	300	130	94	94	125	18	110	110	550
5	114	114	120	190	97	140	300	300	300	130	94	94	125	23	110	110	550
6	114	114	120	190	97	140	300	300	300	130	94	94	125	29	110	110	550
7	114	114	120	190	97	140	300	300	300	130	94	94	125	34	110	110	550
15	114	114	120	190	97	140	300	300	300	130	94	94	125	50	110	110	550
16	114	114	120	190	97	140	300	300	300	130	94	94	125	48	110	110	550
17	114	114	120	190	97	140	300	300	300	130	94	94	190	44	110	110	550
18	114	114	120	190	97	140	300	300	300	190	174	174	270	50	110	110	550
19	114	114	120	190	97	140	300	300	300	130	94	254	350	50	110	110	550
20	114	114	120	190	97	140	300	300	300	190	159	314	365	50	110	110	550
21	114	114	120	190	97	140	300	300	300	250	239	254	285	50	110	110	550
22	114	114	120	190	97	140	300	300	300	150	159	174	205	44	110	110	550
23	114	114	120	80	97	118	230	300	300	130	94	94	125	39	110	110	450
24	114	114	120	80	97	68	110	300	300	130	94	94	125	33	110	110	350

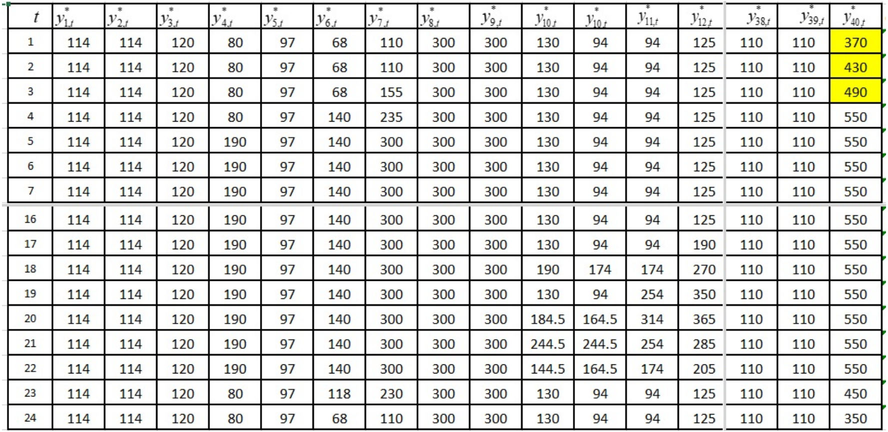
The minimum total fuel cost $f^* = 2533410.3$

Elapsed time $\tau = 0.107098$ *sec*

The x^*-squared norm $q^*(2) = 103051228.0$

Solution: Example 3 Problem A2.2

Problem A2.2 solution y^*:



t	$y^*_{1,t}$	$y^*_{2,t}$	$y^*_{3,t}$	$y^*_{4,t}$	$y^*_{5,t}$	$y^*_{6,t}$	$y^*_{7,t}$	$y^*_{8,t}$	$y^*_{9,t}$	$y^*_{10,t}$	$y^*_{10,t}$	$y^*_{11,t}$	$y^*_{12,t}$	$y^*_{38,t}$	$y^*_{39,t}$	$y^*_{40,t}$
1	114	114	120	80	97	68	110	300	300	130	94	94	125	110	110	370
2	114	114	120	80	97	68	110	300	300	130	94	94	125	110	110	430
3	114	114	120	80	97	68	155	300	300	130	94	94	125	110	110	490
4	114	114	120	80	97	140	235	300	300	130	94	94	125	110	110	550
5	114	114	120	190	97	140	300	300	300	130	94	94	125	110	110	550
6	114	114	120	190	97	140	300	300	300	130	94	94	125	110	110	550
7	114	114	120	190	97	140	300	300	300	130	94	94	125	110	110	550
16	114	114	120	190	97	140	300	300	300	130	94	94	125	110	110	550
17	114	114	120	190	97	140	300	300	300	130	94	94	190	110	110	550
18	114	114	120	190	97	140	300	300	300	190	174	174	270	110	110	550
19	114	114	120	190	97	140	300	300	300	130	94	254	350	110	110	550
20	114	114	120	190	97	140	300	300	300	184.5	164.5	314	365	110	110	550
21	114	114	120	190	97	140	300	300	300	244.5	244.5	254	285	110	110	550
22	114	114	120	190	97	140	300	300	300	144.5	164.5	174	205	110	110	550
23	114	114	120	80	97	118	230	300	300	130	94	94	125	110	110	450
24	114	114	120	80	97	68	110	300	300	130	94	94	125	110	110	350

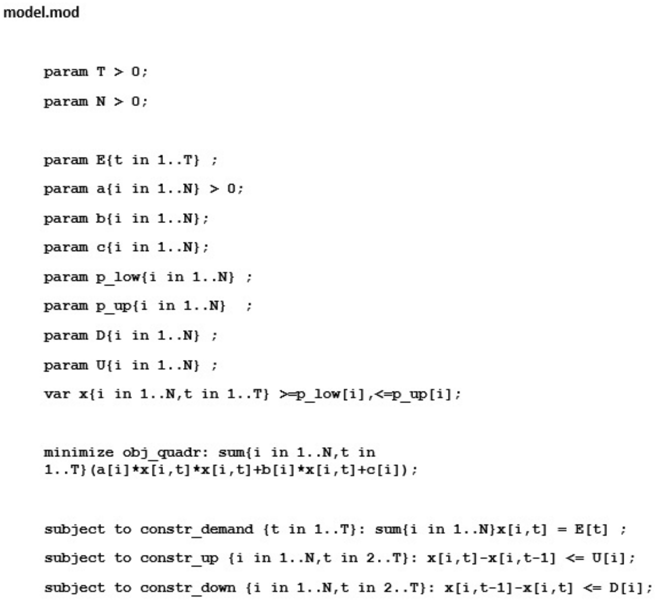
The y^*-squared norm $q^* = 103001307.668$

The minimum total fuel cost $f^* = 2533410.3$

Elapsed time $\tau = 0.258795$ *sec*

$x^* \neq y^*,\ q^* < q^*(2)$

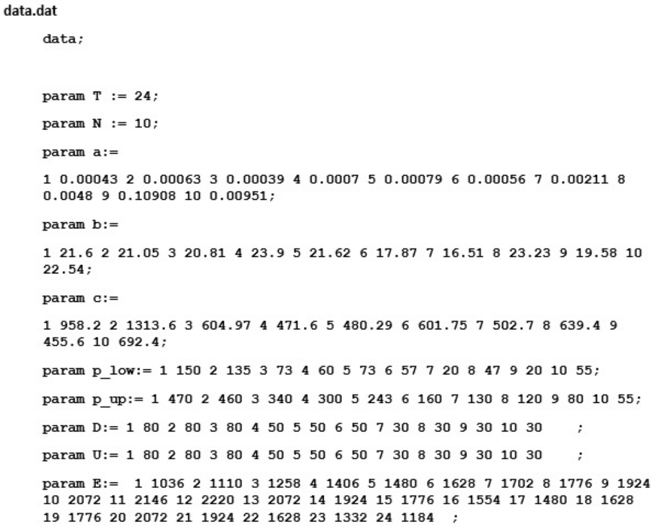
Example 1: AMPL implementation - Block 1. Model



model.mod

```
param T > 0;

param N > 0;


param E{t in 1..T} ;

param a{i in 1..N} > 0;

param b{i in 1..N};

param c{i in 1..N};

param p_low{i in 1..N} ;

param p_up{i in 1..N}  ;

param D{i in 1..N} ;

param U{i in 1..N} ;

var x{i in 1..N,t in 1..T} >=p_low[i],<=p_up[i];


minimize obj_quadr: sum{i in 1..N,t in
1..T}(a[i]*x[i,t]*x[i,t]+b[i]*x[i,t]+c[i]);


subject to constr_demand {t in 1..T}: sum{i in 1..N}x[i,t] = E[t] ;

subject to constr_up {i in 1..N,t in 2..T}: x[i,t]-x[i,t-1] <= U[i];

subject to constr_down {i in 1..N,t in 2..T}: x[i,t-1]-x[i,t] <= D[i];
```

Example 1: AMPL implementation - Block 2. Data

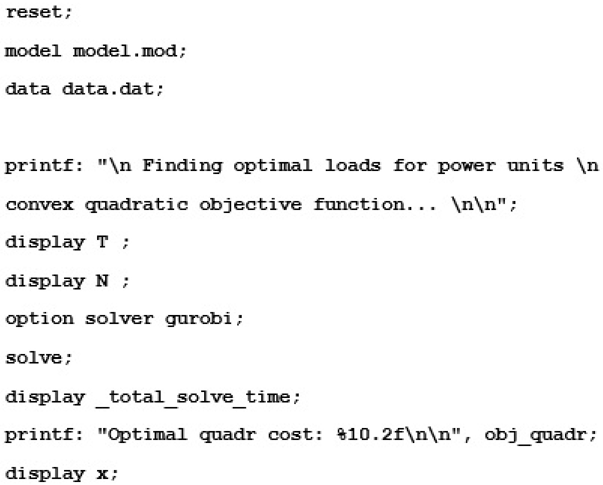


data.dat

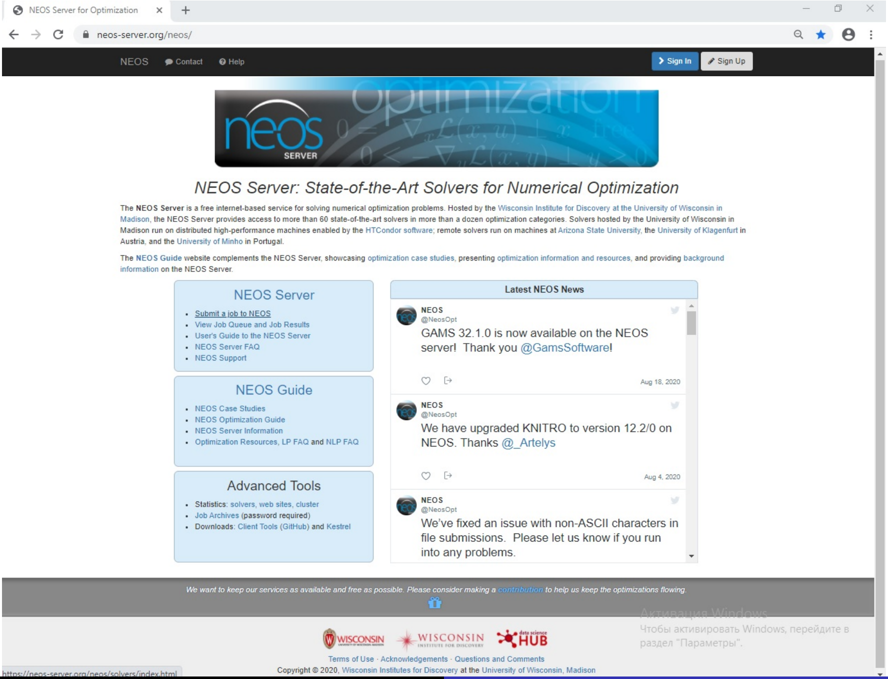
```
data;


param T := 24;

param N := 10;

param a:=

1 0.00043 2 0.00063 3 0.00039 4 0.0007 5 0.00079 6 0.00056 7 0.00211 8
0.0048 9 0.10908 10 0.00951;

param b:=

1 21.6 2 21.05 3 20.81 4 23.9 5 21.62 6 17.87 7 16.51 8 23.23 9 19.58 10
22.54;

param c:=

1 958.2 2 1313.6 3 604.97 4 471.6 5 480.29 6 601.75 7 502.7 8 639.4 9
455.6 10 692.4;

param p_low:= 1 150 2 135 3 73 4 60 5 73 6 57 7 20 8 47 9 20 10 55;

param p_up:= 1 470 2 460 3 340 4 300 5 243 6 160 7 130 8 120 9 80 10 55;

param D:= 1 80 2 80 3 80 4 50 5 50 6 50 7 30 8 30 9 30 10 30     ;

param U:= 1 80 2 80 3 80 4 50 5 50 6 50 7 30 8 30 9 30 10 30     ;

param E:=  1 1036 2 1110 3 1258 4 1406 5 1480 6 1628 7 1702 8 1776 9 1924
10 2072 11 2146 12 2220 13 2072 14 1924 15 1776 16 1554 17 1480 18 1628
19 1776 20 2072 21 1924 22 1628 23 1332 24 1184  ;
```

Example 1: AMPL implementtion - Block 3. Command file

loads.run



```
reset;
model model.mod;
data data.dat;

printf: "\n Finding optimal loads for power units \n
convex quadratic objective function... \n\n";
display T ;
display N ;
option solver gurobi;
solve;
display _total_solve_time;
printf: "Optimal quadr cost: %10.2f\n\n", obj_quadr;
display x;
```



NEOS Server: State-of-the-Art Solvers for Numerical Optimization

The **NEOS Server** is a free internet-based service for solving numerical optimization problems. Hosted by the Wisconsin Institute for Discovery at the University of Wisconsin in Madison, the NEOS Server provides access to more than 60 state-of-the-art solvers in more than a dozen optimization categories. Solvers hosted by the University of Wisconsin in Madison run on distributed high-performance machines enabled by the HTCondor software; remote solvers run on machines at Arizona State University, the University of Klagenfurt in Austria, and the University of Minho in Portugal.

The **NEOS Guide** website complements the NEOS Server, showcasing optimization case studies, presenting optimization information and resources, and providing background information on the NEOS Server.

NEOS Server

- Submit a job to NEOS
- View Job Queue and Job Results
- User's Guide to the NEOS Server
- NEOS Server FAQ
- NEOS Support

NEOS Guide

- NEOS Case Studies
- NEOS Optimization Guide
- NEOS Server Information
- Optimization Resources, LP FAQ and NLP FAQ

Advanced Tools

- Statistics: solvers, web sites, cluster
- Job Archives (password required)
- Downloads: Client Tools (GitHub) and Kestrel

Latest NEOS News

NEOS
@NeosOpt
GAMS 32.1.0 is now available on the NEOS server! Thank you @GamsSoftware!
Aug 18, 2020

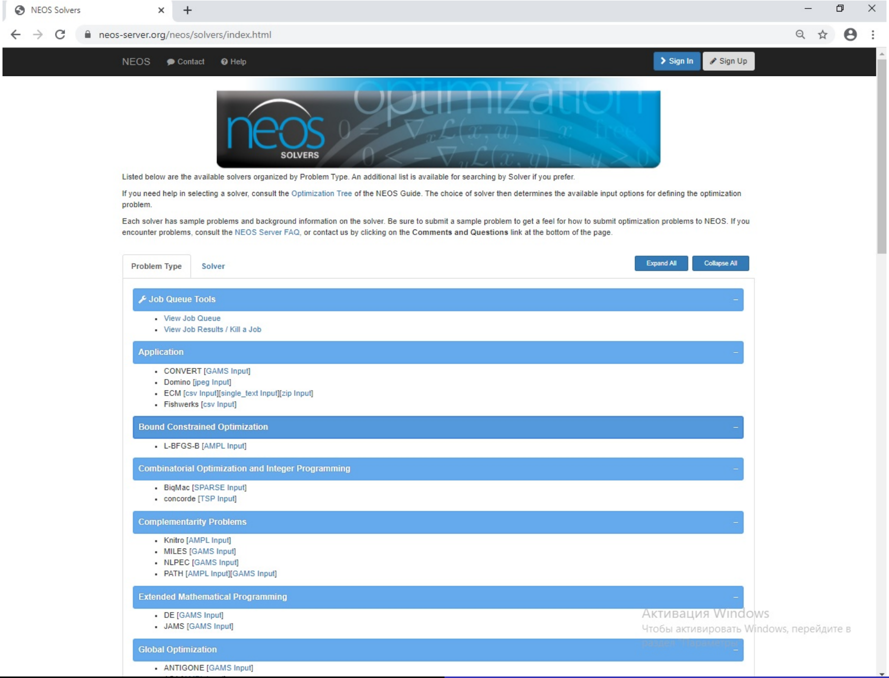
NEOS
@NeosOpt
We have upgraded KNITRO to version 12.2/0 on NEOS. Thanks @_Artelys
Aug 4, 2020

NEOS
@NeosOpt
We've fixed an issue with non-ASCII characters in file submissions. Please let us know if you run into any problems.

We want to keep our services as available and free as possible. Please consider making a contribution to help us keep the optimizations flowing.

Terms of Use · Acknowledgements · Questions and Comments

https://neos-server.org/neos/solvers/index.html



Listed below are the available solvers organized by Problem Type. An additional list is available for searching by Solver if you prefer.

If you need help in selecting a solver, consult the Optimization Tree of the NEOS Guide. The choice of solver then determines the available input options for defining the optimization problem.

Each solver has sample problems and background information on the solver. Be sure to submit a sample problem to get a feel for how to submit optimization problems to NEOS. If you encounter problems, consult the NEOS Server FAQ, or contact us by clicking on the **Comments and Questions** link at the bottom of the page.

Job Queue Tools

- View Job Queue
- View Job Results / Kill a Job

Application

- CONVERT [GAMS Input]
- Domino [jpeg Input]
- ECM [csv Input][single_text Input][zip Input]
- Fishwerks [csv Input]

Bound Constrained Optimization

- L-BFGS-B [AMPL Input]

Combinatorial Optimization and Integer Programming

- BiqMac [SPARSE Input]
- concorde [TSP Input]

Complementarity Problems

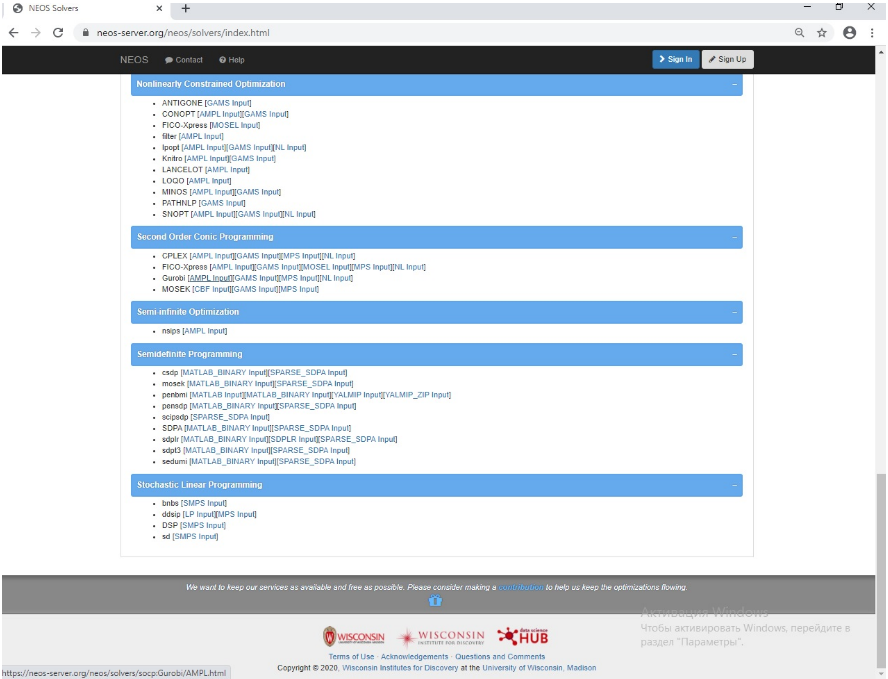
- Knitro [AMPL Input]
- MILES [GAMS Input]
- NLPEC [GAMS Input]
- PATH [AMPL Input][GAMS Input]

Extended Mathematical Programming

- DE [GAMS Input]
- JAMS [GAMS Input]

Global Optimization

- ANTIGONE [GAMS Input]



Nonlinearly Constrained Optimization

- ANTIGONE [GAMS Input]
- CONOPT [AMPL Input][GAMS Input]
- FICO-Xpress [MOSEL Input]
- filter [AMPL Input]
- Ipopt [AMPL Input][GAMS Input][NL Input]
- Knitro [AMPL Input][GAMS Input]
- LANCELOT [AMPL Input]
- LOQO [AMPL Input]
- MINOS [AMPL Input][GAMS Input]
- PATHNLP [GAMS Input]
- SNOPT [AMPL Input][GAMS Input][NL Input]

Second Order Conic Programming

- CPLEX [AMPL Input][GAMS Input][MPS Input][NL Input]
- FICO-Xpress [AMPL Input][GAMS Input][MOSEL Input][MPS Input][NL Input]
- Gurobi [AMPL Input][GAMS Input][MPS Input][NL Input]
- MOSEK [CBF Input][GAMS Input][MPS Input]

Semi-infinite Optimization

- nsips [AMPL Input]

Semidefinite Programming

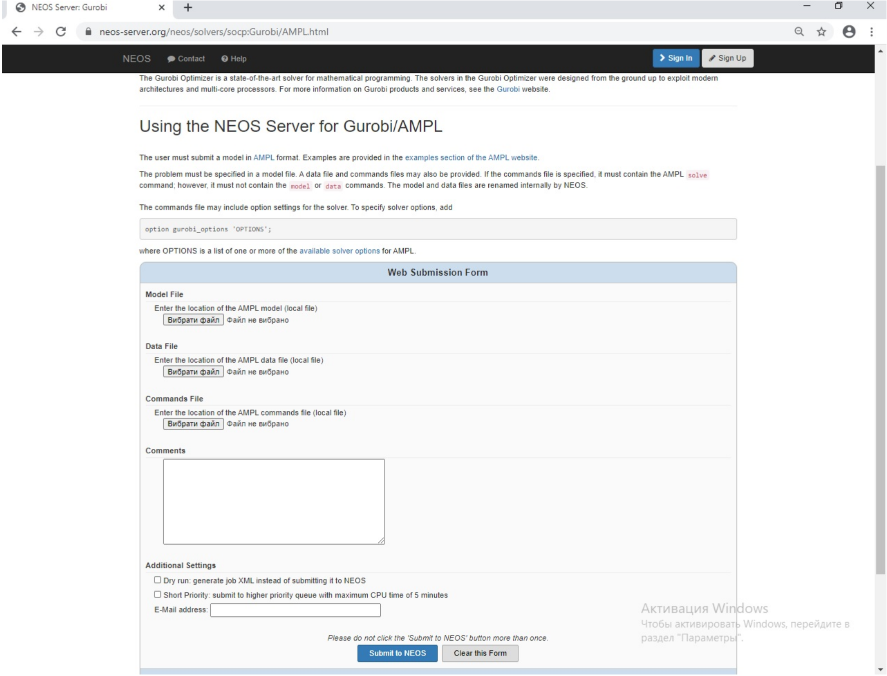
- csdp [MATLAB_BINARY Input][SPARSE_SDPA Input]
- mosek [MATLAB_BINARY Input][SPARSE_SDPA Input]
- penbmi [MATLAB Input][MATLAB_BINARY Input][YALMIP Input][YALMIP_ZIP Input]
- pensdp [MATLAB_BINARY Input][SPARSE_SDPA Input]
- scipsdp [SPARSE_SDPA Input]
- SDPA [MATLAB_BINARY Input][SPARSE_SDPA Input]
- sdplr [MATLAB_BINARY Input][SDPLR Input][SPARSE_SDPA Input]
- sdpt3 [MATLAB_BINARY Input][SPARSE_SDPA Input]
- sedumi [MATLAB_BINARY Input][SPARSE_SDPA Input]

Stochastic Linear Programming

- bnbs [SMPS Input]
- ddsip [LP Input][MPS Input]
- DSP [SMPS Input]
- sd [SMPS Input]

We want to keep our services as available and free as possible. Please consider making a contribution to help us keep the optimizations flowing.

Terms of Use · Acknowledgements · Questions and Comments



The Gurobi Optimizer is a state-of-the-art solver for mathematical programming. The solvers in the Gurobi Optimizer were designed from the ground up to exploit modern architectures and multi-core processors. For more information on Gurobi products and services, see the Gurobi website.

Using the NEOS Server for Gurobi/AMPL

The user must submit a model in AMPL format. Examples are provided in the examples section of the AMPL website.

The problem must be specified in a model file. A data file and commands files may also be provided. If the commands file is specified, it must contain the AMPL `solve` command; however, it must not contain the `model` or `data` commands. The model and data files are renamed internally by NEOS.

The commands file may include option settings for the solver. To specify solver options, add

```
option gurobi_options 'OPTIONS';
```

where OPTIONS is a list of one or more of the available solver options for AMPL.

Web Submission Form

Model File

Enter the location of the AMPL model (local file)

Вибрати файл Файл не вибрано

Data File

Enter the location of the AMPL data file (local file)

Вибрати файл Файл не вибрано

Commands File

Enter the location of the AMPL commands file (local file)

Вибрати файл Файл не вибрано

Comments

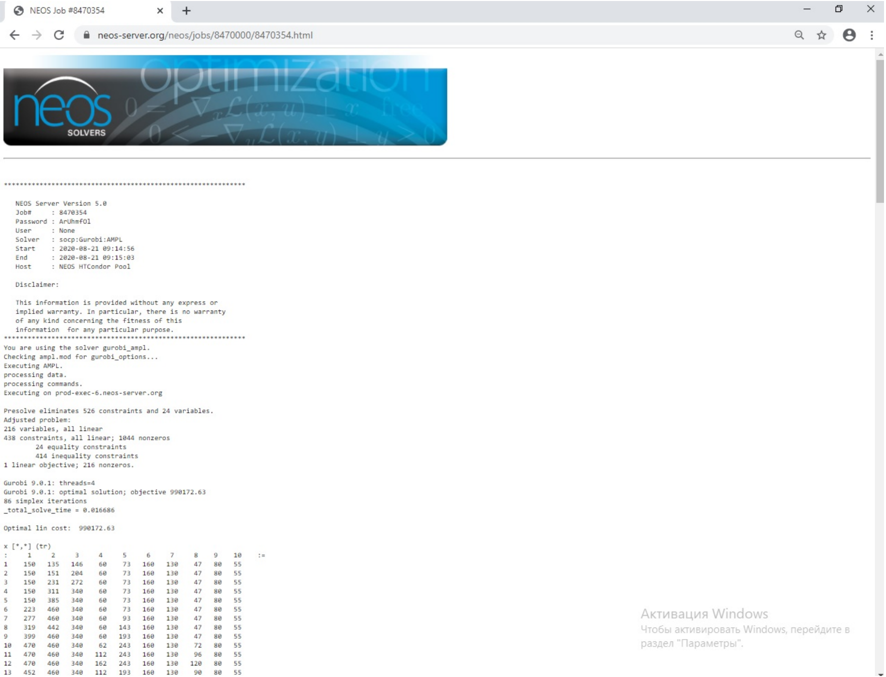
Additional Settings

□ Dry run: generate job XML instead of submitting it to NEOS

□ Short Priority: submit to higher priority queue with maximum CPU time of 5 minutes

E-Mail address:

Please do not click the 'Submit to NEOS' button more than once.



```
*************************************************************

   NEOS Server Version 5.0
   Job#     : 8470354
   Password : ArUhmfOl
   User     : None
   Solver   : socp:Gurobi:AMPL
   Start    : 2020-08-21 09:14:56
   End      : 2020-08-21 09:15:03
   Host     : NEOS HTCondor Pool

   Disclaimer:

   This information is provided without any express or
   implied warranty. In particular, there is no warranty
   of any kind concerning the fitness of this
   information  for any particular purpose.
*************************************************************
You are using the solver gurobi_ampl.
Checking ampl.mod for gurobi_options...
Executing AMPL.
processing data.
processing commands.
Executing on prod-exec-6.neos-server.org

Presolve eliminates 526 constraints and 24 variables.
Adjusted problem:
216 variables, all linear
438 constraints, all linear; 1044 nonzeros
        24 equality constraints
        414 inequality constraints
1 linear objective; 216 nonzeros.

Gurobi 9.0.1: threads=4
Gurobi 9.0.1: optimal solution; objective 990172.63
86 simplex iterations
_total_solve_time = 0.016686

Optimal lin cost:  990172.63

x [*,*] (tr)
:     1     2     3     4     5     6     7     8    9    10    :=
1    150   135   146    60    73   160   130    47   80   55
2    150   151   204    60    73   160   130    47   80   55
3    150   231   272    60    73   160   130    47   80   55
4    150   311   340    60    73   160   130    47   80   55
5    150   385   340    60    73   160   130    47   80   55
6    223   460   340    60    73   160   130    47   80   55
7    277   460   340    60    93   160   130    47   80   55
8    319   442   340    60   143   160   130    47   80   55
9    399   460   340    60   193   160   130    47   80   55
10   470   460   340    62   243   160   130    72   80   55
11   470   460   340   112   243   160   130    96   80   55
12   470   460   340   162   243   160   130   120   80   55
13   452   460   340   112   193   160   130    90   80   55
```

Conclusion

- ED-problems in the form of convex quadratic and linear optimization programs are studied;
- Two classes of the problems having a unique solution are singled out;
- For the case of multiple EDP-solutions, a way to choose a single one is proposed;
- The problems are implemented in AMPL modeling language;
- The results of computational experiments using the Gurobi solver on the NEOS server of three EDP test examples for 10 and 40 generators and a daily hourly electrical load are presented. For two of them, the existence of the unique solution is justified, and for one of them, the presence of multiple solutions is confirmed.

Thank you!
Questions?