A Collection of 200 Test Problems for Nonlinear Mixed-Integer Programming in Fortran

- User's Guide -

Address:	Klaus Schittkowski Department of Computer Science University of Bayreuth D - 95440 Bayreuth
E-mail:	klaus.schittkowski@uni-bayreuth.de
Web:	http://www.klaus-schittkowski.de
Date:	September, 2015

Abstract

The availability of mixed-integer nonlinear programming test problems is extremely important to test optimization codes or to develop new algorithms. We describe the usage of 200 Fortran subroutines of a set of non-convex test problems, where most of them are taken from existing collections, especially from the GAMS library MINLPLib. Program organization and numerical test results are presented, moreover some auxiliary routines to facilitate integration under own test environments. A frame to evaluate all test problems in a loop, is described together with the usage of the source codes of the collection. The implementation is thread-safe basic Fortran, more than 40,000 lines, and the codes are easily transferred to C by f2c. Some numerical results obtained by our own mixed-integer optimization code MISQP are included. We show that we need about as many function evaluations for solving the continuously relaxed test problems as for solving all mixed-integer problems directly. The source codes can be downloaded from the home page of the author.

1 Introduction

We consider the nonlinear mixed-integer optimization problem to minimize an objective function f under nonlinear equality and inequality constraints, i.e.,

$$x \in \mathbb{R}^{n_c}, y \in \mathbb{Z}^{n_i} : \quad \begin{array}{l} \min f(x,y) \\ g_j(x,y) = 0 \ , \quad j = 1, \ldots, m_e \ , \\ g_j(x,y) \geq 0 \ , \quad j = m_e + 1, \ldots, m \ , \\ x_l \leq x \leq x_u \ , \\ y_l \leq y \leq y_u \end{array} \tag{1}$$

where x and y denote the continuous and the integer variables, respectively. It is assumed that the problem functions $f(x,y)$ and $g_j(x,y)$, $j = 1, \ldots, m$, are continuously differentiable subject to $x \in \mathbb{R}^{n_c}$. Integer variables include binary variables.

Most of the test problems are taken from the GAMS Model Library MINLPLib, cf. Bussieck, Drud, and Meeraus [3] and can be downloaded from

`http://www.gamsworld.org/minlp/MINLPLib.htm`

The collection is widely used to test and compare algorithms, see e.g. Still and Westerlund [20] or Maniezzo, Stützle, and Voß [17]. They are implemented in GAMS and are easily transformed into other modeling languages like AMPL, BARON, GAMS, LINGO, or MINOPT, for example.

It is important to understand that the implementation and the transfer of test examples from the literature, especially from GAMS to Fortran, is always subject to human errors (more than 40,000 lines of coding), despite of automating the transfer as much as possible. Known optimal solution values are retained. Although we tried to check the implementation over and over, there might be bugs and we would be grateful to receive reports in case of inconsistencies.

Many of the underlying optimization problems are non-convex and we are not sure whether our own solutions are always global ones. Thus, our own code sometimes stops at a feasible solution which is not optimal, although the internal heuristic stopping tests are all satisfied. Note that in mixed-integer optimization, there does not exist a proper definition of a *local minimum*.

Our own motivation for collecting test problems is to develop new nonlinear mixed-integer optimization software for solving practical engineering optimization problems with expensive function evaluations. A typical application is the solution of mechanical structural optimal design problems based on a time-consuming FE analysis. Our main goal is to derive codes requiring as few function evaluations as possible. To adjust the test problems to our needs, we implemented them in Fortran.

All test problems are relaxable, i.e., function values can be computed not only for integer values $y \in \mathbb{Z}^{n_i}$, but also for any real values in between. In other words, the integrality condition $y \in \mathbb{Z}^{n_i}$ in (1) can be replaced by $y \in \mathbb{R}^{n_i}$.

However, derivatives subject to integer and boolean variables are internally approximated by a difference formula evaluated at grid points to get descent information. Partial derivatives subject to continuous variables are approximated by a forward difference formula. In other words, we do not exploit the fact that the test problems are relaxable. In a comparative study of Exler, Lehmann and Schittkowski [10], numerical results for different ways of providing derivative information are presented.

The Fortran source codes of all test problems are available through the link

`http://klaus-schittkowski.de/home.htm`

A brief summary of all examples together with relevant data for n_c, n_i, m, m_e and in particular the best known solution values is presented in Section 2. The usage of the subroutines is documented in Section 3 together with an example. A possible test environment is listed that shows how our nonlinear mixed-integer code MISQP, see Exler, Lehmann, and Schittkowski [8, 9], can be applied. Section 4 contains numerical results obtained by MISQP including objective function values, constraint violations, number of function calls, number of iterations, and especially errors in objective function subject to the best optimal solution values we know.

2 The Test Problems

A list of characteristic problem data is presented in Table 1, where the following data are listed:

no	-	test problem number,
name	-	name of the test problem as used in our collection and in the literature,
ref	-	reference, if available,
n_c	-	number of continuous variables,
n_d	-	number of integer variables without binary ones,
n_b	-	number of binary variables,
m_e	-	number of equality constraints,
m	-	number of all constraints,
$f(x^\star, y^\star)$	-	best known objective function value.

Note that $n_i = n_d + n_b$ and that at least all test problems with nonlinear equality constraints are not convex.

Table 2: Mixed-Integer Test Problems

no	*name*	*ref*	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
1	MITP1		2	3	0	0	1	-0.10010E+05
2	MITP2		2	0	3	0	7	0.35000E+01
3	QIP1		0	4	0	0	4	-0.20000E+02
4	ASAADI11	[1]	1	3	0	0	3	-0.40957E+02
5	ASAADI12	[1]	0	4	0	0	3	-0.38000E+02
6	ASAADI21	[1]	3	4	0	0	4	0.69490E+03
7	ASAADI22	[1]	0	7	0	0	4	0.70000E+03
8	ASAADI31	[1]	4	6	0	0	8	0.37220E+02
9	ASAADI32	[1]	0	10	0	0	8	0.43000E+02
10	DIRTY		12	13	0	0	10	-0.30472E+09
11	BRAAK1	[2]	4	3	0	0	2	0.10000E+01
12	BRAAK2	[2]	4	3	0	0	4	-0.27183E+01
13	BRAAK3	[2]	4	3	0	0	4	-0.19656E+07
14	DEX2	[4]	0	2	0	0	2	-0.56938E+02
15	FUEL	[3]	12	0	3	6	15	0.85661E+04
16	WP02	[22]	1	1	0	0	2	-0.24444E+01
17	NVS01	[3]	1	2	0	1	3	0.12470E+02
18	NVS02	[3]	3	5	0	3	3	0.59642E+01
19	NVS03	[3]	0	2	0	0	2	0.16000E+02
20	NVS04	[3]	0	2	0	0	0	0.72000E+00
21	NVS05	[3]	6	2	0	4	9	0.54709E+01
22	NVS06	[3]	0	2	0	0	0	0.17703E+01
23	NVS07	[3]	0	3	0	0	2	0.40000E+01
24	NVS08	[3]	1	2	0	0	3	0.23450E+02
25	NVS09	[3]	0	10	0	0	0	-0.43134E+02
26	NVS10	[3]	0	2	0	0	2	-0.31080E+03
27	NVS11	[3]	0	3	0	0	3	-0.43100E+03
28	NVS12	[3]	0	4	0	0	4	-0.48120E+03
29	NVS13	[3]	0	5	0	0	5	-0.58520E+03
30	NVS14	[3]	3	5	0	3	3	-0.40358E+05
31	NVS15	[3]	0	3	0	0	1	0.10000E+01
32	NVS16	[3]	0	2	0	0	0	0.70312E+00
33	NVS17	[3]	0	7	0	0	7	-0.11004E+04
34	NVS18	[3]	0	6	0	0	6	-0.77840E+03
35	NVS19	[3]	0	8	0	0	8	-0.10984E+04
36	NVS20	[3]	11	5	0	0	8	0.23092E+03
37	NVS21	[3]	1	2	0	0	2	-0.56848E+01
38	NVS22	[3]	4	4	0	4	9	0.60582E+01
39	NVS23	[3]	0	9	0	0	9	-0.11252E+04
40	NVS24	[3]	0	10	0	0	10	-0.10332E+04
41	GEAR	[3]	0	4	0	0	0	0.10000E+01
42	GEAR2	[3]	4	24	0	4	4	0.10000E+01
43	GEAR2A	[3]	4	0	24	4	4	0.10000E+01
44	GEAR3	[3]	4	4	0	4	4	0.10000E+01
45	GEAR4	[3]	2	4	0	1	1	0.16434E+01
46	M3	[3]	20	0	6	0	43	0.37800E+02
47	M6	[3]	56	0	30	0	157	0.82257E+02
48	M7	[3]	72	0	42	0	211	0.10676E+03
49	FLOUDAS1	[11]	2	0	3	2	5	0.76672E+01
50	FLOUDAS2	[11]	2	0	1	0	3	0.10765E+01
51	FLOUDAS3	[11]	3	0	4	0	9	0.45796E+01
52	FLOUDAS4	[11]	3	0	8	3	7	-0.94636E+00
53	FLOUDAS40	[11]	3	0	8	3	7	-0.93159E+00

(continued)

no	*name*	*ref*	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
54	FLOUDAS5	[11]	0	2	0	0	4	0.31000E+02
55	FLOUDAS6	[11]	1	1	0	0	3	-0.17000E+02
56	SPRING	[3]	5	1	11	5	8	0.84625E+00
57	DU_OPT5	[3]	7	13	0	0	9	0.20254E+02
58	DU_OPT	[3]	7	13	0	0	9	0.42010E+01
59	ST_E13	[3]	1	0	1	0	2	0.22361E+01
60	ST_E14	[3]	7	0	4	4	13	0.45796E+01
61	ST_E15	[3]	2	0	3	2	5	0.76672E+01
62	ST_E27	[3]	2	0	2	0	6	0.20000E+01
63	ST_E29	[3]	3	0	8	2	7	-0.94347E+00
64	ST_E31	[3]	88	0	24	81	135	-0.20000E+01
65	ST_E32	[3]	16	19	0	17	18	-0.14304E+01
66	ST_E35	[3]	25	0	7	15	39	0.10620E+06
67	ST_E36	[3]	1	1	0	1	2	-0.24600E+03
68	ST_E38	[3]	2	2	0	0	3	0.71977E+04
69	ST_E40	[3]	1	3	0	4	8	0.30414E+02
70	ST_MIQP1	[3]	0	0	5	0	1	0.28100E+03
71	ST_MIQP2	[3]	0	4	0	0	3	0.20000E+01
72	ST_MIQP3	[3]	0	2	0	0	1	-0.60000E+01
73	ST_MIQP4	[3]	3	0	3	0	4	-0.45740E+04
74	ST_MIQP5	[3]	5	2	0	0	13	-0.33389E+03
75	ST_TEST1	[3]	0	5	0	0	1	0.10000E+01
76	ST_TEST2	[3]	0	6	0	0	2	-0.92500E+01
77	ST_TEST3	[3]	0	13	0	0	10	-0.70000E+01
78	ST_TEST4	[3]	0	6	0	0	5	-0.70000E+01
79	ST_TEST5	[3]	0	10	0	0	11	-0.11000E+03
80	ST_TEST6	[3]	0	10	0	0	5	0.47100E+03
81	ST_TEST8	[3]	0	24	0	0	20	-0.29605E+05
82	ST_TESTGR1	[3]	0	10	0	0	5	-0.12812E+02
83	ST_TESTGR3	[3]	0	20	0	0	20	-0.20590E+02
84	ST_TESTPH4	[3]	0	3	0	0	10	-0.80500E+02
85	TLN2	[3]	0	6	2	0	12	0.53000E+01
86	TLN4	[3]	0	20	4	0	24	0.85000E+01
87	TLN5	[3]	0	30	5	0	30	0.10600E+02
88	TLN6	[3]	0	42	6	0	36	0.16300E+02
89	NEJI		2	1	0	0	6	-0.11111E+02
90	TST_NAG		4	0	4	2	7	0.29250E+15
91	TLOSS	[3]	0	42	6	0	53	0.16300E+02
92	TLTR	[3]	0	36	12	0	54	0.48067E+02
93	MEANVARX	[3]	21	0	14	8	44	0.14190E+02
94	MINLPHIX	[3]	64	0	20	30	92	0.31669E+03
95	MIP_EX	[13]	2	0	3	0	7	0.35000E+01
96	MGRID_CYCLES1	[23]	0	5	0	0	1	0.80000E+01
97	MGRID_CYCLES2	[23]	0	10	0	0	1	0.30000E+03
98	CROP5	[21]	0	5	0	0	3	0.95310E-01
99	CROP20	[21]	0	20	0	0	3	0.11161E+00
100	CROP50	[21]	0	50	0	0	3	0.32424E+00
101	CROP100	[21]	0	100	0	0	3	0.85147E+00
102	SPLITF1	[10]	3	0	9	3	9	-0.16045E+04
103	SPLITF2	[10]	6	0	18	6	15	-0.18000E+04
104	SPLITF3	[10]	6	0	18	6	15	-0.25083E+04
105	SPLITF4	[10]	6	0	18	6	15	-0.26266E+04
106	SPLITF5	[10]	6	0	18	6	15	-0.28045E+04
107	SPLITF6	[10]	6	0	18	6	15	-0.30995E+04
108	SPLITF7	[10]	9	0	27	9	21	-0.26162E+04
109	SPLITF8	[10]	9	0	27	9	21	-0.30406E+04
110	SPLITF9	[10]	9	0	27	9	21	-0.34045E+04

(continued)

no	*name*	*ref*	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
111	ELF	[3]	30	0	24	6	38	0.19167E+00
112	SPECTRA2	[3]	39	0	30	9	72	0.13978E+02
113	WINDFAC	[3]	11	3	0	13	13	0.25449E+00
114	CSCHED1	[3]	13	0	63	12	22	-0.37604E+05
115	ALAN	[3]	4	0	4	2	7	0.28990E+01
116	PUMP	[3]	15	6	3	13	34	0.13426E+06
117	RAVEM	[3]	58	0	54	25	186	0.26959E+06
118	ORTEZ	[3]	69	0	18	24	74	-0.10205E+05
119	EX1221	[3]	2	0	3	2	5	0.76672E+01
120	EX1222	[3]	2	0	1	0	3	0.10765E+01
121	EX1223	[3]	7	0	4	4	13	0.45796E+01
122	EX1223A	[3]	3	0	4	0	9	0.45796E+01
123	EX1223B	[3]	3	0	4	0	9	0.45796E+01
124	EX1224	[3]	3	0	8	2	7	-0.94347E+00
125	EX1225	[3]	2	0	6	2	10	0.31000E+02
126	EX1226	[3]	2	0	3	1	5	-0.17000E+02
127	EX1233	[3]	40	0	12	20	64	0.15501E+06
128	EX1243	[3]	52	0	16	24	96	0.83403E+05
129	EX1244	[3]	72	0	23	30	129	0.82043E+05
130	EX1252	[3]	24	0	15	22	43	0.12889E+06
131	EX1263	[3]	20	0	72	20	55	0.19600E+02
132	EX1263A	[3]	0	20	4	0	35	0.19600E+02
133	EX1264	[3]	20	0	68	20	55	0.86000E+01
134	EX1264A	[3]	0	20	4	0	35	0.86000E+01
135	EX1265	[3]	30	0	100	30	74	0.10300E+02
136	EX1265A	[3]	0	30	5	0	44	0.10300E+02
137	DIOPHE		0	4	0	1	1	-0.20000E+01
138	EX1266A	[3]	0	42	6	0	53	0.16300E+02
139	GBD	[3]	1	0	3	0	4	0.22000E+01
140	EX3	[3]	24	0	8	17	31	0.68010E+02
141	EX4	[3]	11	0	25	0	30	-0.80641E+01
142	FAC1	[3]	16	0	6	10	18	0.16091E+09
143	FAC2	[3]	54	0	12	21	33	0.33184E+09
144	FAC3	[3]	54	0	12	21	33	0.31982E+08
145	GKOCIS	[3]	8	0	3	5	8	-0.19231E+01
146	KG	[15]	7	0	2	5	9	0.10394E+03
147	SYNTHES1	[3]	3	0	3	0	6	0.60098E+01
148	SYNTHES2	[3]	6	0	5	1	14	0.73035E+02
149	SYNTHES3	[3]	9	0	8	2	23	0.68010E+02
150	PARALLEL	[3]	180	0	25	81	115	0.84056E+03
151	SYNHEAT	[3]	44	0	12	20	64	0.15500E+06
152	SEP1	[3]	27	0	2	22	31	-0.53212E+03
153	DAKOTA	[7]	2	2	0	0	2	0.13634E+01
154	BATCH	[3]	23	0	24	12	73	0.28551E+06
155	BATCHDES	[3]	10	0	9	6	19	0.16743E+06
156	ENIPLAC	[3]	117	0	24	87	189	-0.13186E+06
157	PROB02	[3]	0	6	0	0	8	0.11224E+06
158	PROB03	[3]	0	2	0	0	1	0.10000E+02
159	PROB10	[3]	1	1	0	0	2	0.34455E+01
160	NOUS1	[3]	48	0	2	41	43	0.15671E+01
161	NOUS2	[3]	48	0	2	41	43	0.62597E+00
162	TLS2	[3]	4	2	31	6	24	0.53000E+01
163	TLS4	[3]	16	4	85	20	64	0.85000E+01
164	TLS5	[3]	25	5	131	30	90	0.10600E+02
165	OAER	[3]	6	0	3	3	7	-0.19231E+01
166	PROCSEL	[3]	7	0	3	4	7	-0.19231E+01
167	LICHOU_1	[16]	1	1	0	1	2	-0.24600E+03

(continued)

no	$name$	ref	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
168	LICHOU_2	[16]	2	2	0	0	4	0.71273E+04
169	LICHOU_3	[16]	0	3	0	0	4	0.30414E+01
170	WU_1	[24]	0	0	32	0	0	0.32000E+00
171	WU_2	[24]	0	0	32	0	0	0.97600E+01
172	WU_3	[24]	0	0	64	0	0	0.13788E+00
173	WU_4	[24]	0	0	64	0	0	0.10240E+02
174	OPTPRLOC	[?]	5	0	25	0	30	-0.80641E+01
175	GASNET	[3]	80	0	10	48	69	0.69994E+07
176	TP83	[14]	1	4	0	0	6	-0.30606E+05
177	TP84	[14]	3	2	0	0	6	-0.57152E+07
178	TP85	[14]	2	3	0	0	38	-0.18958E+01
179	TP87	[14]	4	2	0	4	4	0.89582E+04
180	TP93	[14]	5	1	0	0	2	0.13874E+03
181	FEEDTRAY	[3]	90	0	7	83	91	-0.13406E+02
182	FEEDTRAY2	[3]	51	0	36	6	283	0.10000E+01
183	HILBERT20		0	20	0	20	20	0.21000E+03
184	HILBERT50		0	50	0	50	50	0.12750E+04
185	HILBERT100		0	100	0	100	100	0.50500E+04
186	SLOPPY		0	6	0	0	3	0.00000E+00
187	RASTRIGIN	[14]	1	1	0	0	0	-0.19689E+01
188	EMSO		3	0	3	0	4	-0.19231E+01
189	TP1	[14]	0	2	0	0	0	0.00000E+00
190	TP1A	[14]	0	2	0	0	0	0.00000E+00
191	TP1B	[14]	0	2	0	0	0	0.00000E+00
192	TP9	[14]	0	2	0	1	1	-0.50000E+00
193	TP10	[14]	0	2	0	0	1	-0.10000E+01
194	DEB10	[3]	160	0	22	65	129	0.19880E+03
195	IRAP1	[5, 12]	0	68	0	0	18	0.21814E+03
196	IRAP2	[5, 12]	0	38	0	0	20	0.31652E+03
197	IRAP3	[5, 12]	0	40	0	0	21	0.33661E+03
198	IRAP4	[5, 12]	0	45	0	0	16	0.19616E+03
199	IRAP5	[5, 12]	0	60	0	0	16	0.18674E+03
200	IRAP6	[5, 12]	0	34	0	0	18	0.23936E+03

3 The Fortran Subroutines

This section describes the organization of the Fortran codes and shows how to execute a test problem. Since it is assumed that at least a subset of the problems is used within a series of test runs for different optimization programs, the problems are coded in a very flexible manner. The test examples are implemented in thread-safe Fortran 90 without global data (COMMON) or any special Fortran tricks (EQUIVALENCE, ENTRY) and are easily transferred to C by f2c.

All nonlinear mixed-integer test problems of our collection are available together with a test frame in form of Fortran source codes, see

`http://klaus-schittkowski.de/home.htm`

A test problem is identified by its name as used, e.g., in the GAMS MINLPLib. All test problems are collected in a file with name `ALL_EXAMPLES.FOR` from where problem data and objective and constraint function values are retrieved. To call a subset or all of them within a loop and to identify them by a problem number, an interface subroutine is included with file name `GET_MINLP_PROB.FOR`.

Usage:

```
      CALL   GET_MINLP_PROB (   MODE,   IPROB,      M,      ME,   MMAX,
   /                           NCONT,    NBIN,   NINT,   NMAX,      X,
   /                              XL,      XU,      F,      G,   PNAM,
   /                            PREF,     FEX                         )
```

Parameter Definition:

MODE :	Status for returning data, 0 : Returns M, ME, NCONT, NBIN, NINT, starting values in X, lower and upper bounds in XL and XU, the best known optimal objective function value in FEX, and documentation strings in PNAM and PREF. 1 : Given M, ME, NCONT, NBIN, NINT and X, objective and constraint function values are computed subject to the variable values found in X, and returned in F and G(1), ..., G(M).
IPROB :	Specification of a test problem number for which data and function values are to be evaluated.
M :	Number of all constraints, without bounds.
ME :	Number of equality constraints.
MMAX :	Dimension of G. MMAX has to be at least one and at least M.
NCONT :	Number of continuous variables.
NBIN :	Number of binary variables.
NINT :	Number of integer variables.
NMAX :	Dimension of X, XL, and XU. NMAX has to be at least two and at least NCONT+NBIN+NINT.
X(NMAX) :	When called with MODE=0, X returns starting values. In the driving program, the dimension of X must be equal to NMAX. X contains first NCONT continuous, then NBIN boolean variables followed by NINT integer variables.
XL(NMAX),	When called with MODE=0, the one-dimensional arrays XL and XU contain the lower and
XU(NMAX):	upper bounds of the variables, first for the continuous, then for the binary and subsequently for the integer variables.
F :	When called with MODE=1, the double precision parameter F returns the objective function value computed for X.
G(MMAX) :	When called with MODE=1, the double precision array G contains the constraint function values G(1),...,G(M) computed for X.

PNAM : On return with MODE=0, PNAM contains the test problem name identical to the subroutine name. The string length is 30.

PREF : On return with MODE=0, PREF contains a Latex reference to bibliographic data, if available, as used for this documentation. The string length is 30.

FEX : On return with MODE=0, FEX contains best known optimal objective function value.

It is important that the values of M, ME, N, NBIN, NINT, XL, and XU must not be changed after the first call of GET_MINLP_PROB with MODE=0 for the same IPROB value. The file `GET_MINLP_PROB.FOR` contains auxiliary routines called called by some test problems, i.e., some adapted GAMS routines, one for generating random numbers, and a code for safeguarded division,

```
function sqr(x)
double precision sqr, x
sqr = x*x
return
end
function power(x,m)
double precision power, x
integer m
if (m.eq.2) power = x*x
if (m.eq.3) power = x*x*x
return
end
function xlog(x)
double precision xlog,x,eps
data eps/1.0d-3/
xlog = dlog(dabs(x)+eps)
return
end
function xdiv(x)
double precision xdiv,x,eps
data eps/1.0d-8/
if (x.gt.0.0d0) then
   xdiv = dmax1(dabs(x),eps)
else
   xdiv = -dmax1(dabs(x),eps)
endif
return
end
```

Any individual test problem with placeholder `<TP>` for its name has the same calling sequence without the parameter IPROB, i.e.,

```
          CALL   <TP> (   MODE ,        M,        ME,   MMAX,   NCONT,
      /                    NBIN,   NINT,   NMAX,        X,        XL,
      /                      XU,        F,         G,   PNAM,     PREF,
      /                     FEX                                       )
```

To give an example, we consider test problem with number 56 called SPRING in the GAMS test problem library MINLPLib [3],

$$
x \in I\!R^5, y \in \mathbb{Z}^{12}: \quad
\begin{array}{l}
\min \ (1.570796327 + 0.7853981635 y_1)\, x_1 x_2^2 \\
-\dfrac{x_1}{x_2} + x_4 = 0 \ , \\
-\dfrac{4x_4 - 1}{4x_4 - 4} + \dfrac{0.615}{x_4} + x_5 = 0 \ , \\
-6.95652173913044 \dfrac{y_1 x_4^3}{x_2} + x_3 = 0 \ , \\
\quad x_2 - 0.207 y_2 - 0.225 y_3 - 0.244 y_4 - 0.263 y_5 - 0.283 y_6 - 0.307 y_7 \\
\qquad -0.331 y_8 - 0.362 y_9 - 0.394 y_{10} - 0.4375 y_{11} - 0.5 y_{12} = 0 \ , \\
\quad y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 + y_9 + y_{10} + y_{11} + y_{12} - 1 = 0 \ , \\
-2546.47908913782 \dfrac{x_5 x_4}{x_2^2} + 189000 \geq 0 \ , \\
-(2.1 + 1.05 y_1) x_2 - 1000 x_3 + 14 \geq 0 \ , \\
-x_1 - x_2 + 3 \geq 0 \ , \\
\quad 0.414 \leq \ x_1 \leq 10 \ , \ \ 0.207 \leq \ x_2 \leq 10 \ , \ \ 0.0018 \leq \ x_3 \leq 0.02 \ , \\
\quad 1.1 \leq \ x_4 \leq 10 \ , \ \ 0.1 \leq \ x_5 \leq 9.5 \ , \ \ 5 \leq \ x_5 \leq 10 \ , \\
\quad 0 \leq \ y_1 \leq 10 \ , \ \ y_i \in \{0, 1\}, i = 2, \ldots 12
\end{array}
$$

We have five continuous, eleven binary, and one integer variable, moreover five nonlinear equality and three inequality constraints. The code for test example SPRING is listed below. Note that we try to follow the GAMS implementation MINLPLib as much as possible.

```
      subroutine spring( mode,     m,     me,  mmax, ncont,
     /                   nbin,  nint,   nmax,     x,    xl,
     /                     xu,     f,      g,  pnam,  pref,
     /                    fex )

*  MINLP written by GAMS Convert at 04/27/01 14:53:07
*
*  Equation counts
*     Total       E       G       L       N       X
*         9       6       0       3       0       0
*
```

```
*  Variable counts
*                x       b       i     s1s     s2s      sc      si
*    Total    cont  binary integer    sos1    sos2   scont    sint
*       18       6      11       1       0       0       0       0
*  FX    0       0       0       0       0       0       0       0
*
*  Nonzero counts
*    Total   const      NL     DLL
*       44      30      14       0
*
*  Solve m using MINLP minimizing objvar;

      implicit none
      integer m, me,ncont, nint, nbin, n, nmax, mmax, mode, i
      double precision x(nmax), xl(nmax), xu(nmax), f, fex,
     /        g(mmax), x1, x2, x3, i4, x5, x6, b7, b8, b9, b10, b11,
     /        b12, b13, b14, b15, b16, b17
      character*30 pnam, pref

      if (mode.eq.0) then
         pnam  = 'SPRING'
         pref  = '\cite{MINLPLib}'
         fex   = 0.8462457d0
         ncont = 5
         nint  = 1
         nbin  = 11
         n     = ncont + nbin + nint
         m     = 8
         me    = 5
         xl(1)  = 0.414d0
         x(1)   = 0.5d0
         xu(1)  = 10.0d0
         xl(2)  = 0.207d0
         x(2)   = 100.0d0
         xu(2)  = 100.0d0
         xl(3)  = 0.00178571428571429d0
         x(3)   = 0.002d0
         xu(3)  = 0.02d0
         xl(4)  = 1.1d0
         x(4)   = 1.5d0
         xu(4)  = 10.0d0
         xl(5)  = 1.0d0
         x(5)   = 1.0d0
         xu(5)  = 10.0d0
         do i=ncont+1,ncont+nbin
            xl(i) = 0.0d0
            x(i)  = 0.0d0
            xu(i) = 1.0d0
         enddo
         xl(n) = 1.0d0
         x(n)  = 1.0d0
         xu(n) = 10.0d0
         goto 999
      endif

      x1 = x(1)
      x2 = x(2)
      x3 = x(3)
      i4 = x(17)
      x5 = x(4)
      x6 = x(5)
      b7 = x(6)
      b8 = x(7)
      b9 = x(8)
      b10 = x(9)
```

```
      b11 = x(10)
      b12 = x(11)
      b13 = x(12)
      b14 = x(13)
      b15 = x(14)
      b16 = x(15)
      b17 = x(16)

      f = (1.570796327d0 + 0.7853981635d0*i4)*x1*x2**2

      g(1) = - x1/x2 + x5

      g(2) = - ((4.0d0*x5 - 1.0d0)/(4.0d0*x5 - 4.0d0) + 0.615d0/x5) + x6

      g(3) = -6.95652173913044d-7*i4*x5**3/x2 + x3

      g(4) = x2 - 0.207d0*b7 - 0.225D0*b8 - 0.244d0*b9 - 0.263d0*b10
     /       - 0.283d0*b11 - 0.307d0*b12 - 0.331d0*b13 - 0.362d0*b14
     /       - 0.394d0*b15 - 0.4375d0*b16 - 0.5d0*b17

      g(5) = b7 + b8 + b9 + b10 + b11 + b12 + b13 + b14 + b15
     /       + b16 + b17 - 1.0d0

      g(6) = -2546.47908913782d0*x6*x5/x2**2 + 189000.0d0

      g(7) = -(2.1d0 + 1.05d0*i4)*x2 - 1000.0d0*x3 + 14.0d0

      g(8) = -x1 - x2 + 3.0d0

  999 continue
      return
      end
```

The subsequent code shows how test example SPRING is executed either directly or from the framework given by subroutine GET_MINLP_PROB. It's serial number is 56. The main program for executing MISQP by reverse communication can be implemented as follows,

```
      implicit     none
      integer      nmax, mmax, mmax0, maxnde, maxcut, lerw, leiw, lelw
      parameter   (nmax   = 1000,
     /             mmax   = 3000,
     /             maxcut = 500,
     /             mmax0  = 2*mmax + maxcut + 20,
     /             maxnde = 1000)
      parameter   (lerw  = 7*nmax*nmax/2 + mmax0*nmax + 102*nmax
     /                    + 34*mmax0 + 3*maxnde + 3*mmax*mmax/2
     /                    + 4*mmax*nmax + 400,
     /             leiw  = 14*nmax + 5*mmax0 + 6*maxnde + 105,
     /             lelw  = 4*nmax + mmax0 + 100)
      double precision x(nmax), g(mmax), df(nmax), dg(mmax,nmax),
     /             xl(nmax), xu(nmax), geps(mmax), rw(lerw)
      logical      lw(lelw), ideriv(nmax), lopt(60)
      character*30 pnam, pref
      double precision f, feps, fex, acc, eps, xbck, ropt(60)
      integer      m, me, n, ncont, nint, nbin, ifail, maxit, iprint,
     /             iout, iprob, i, j, iw(leiw), iopt(60)

!   Set test problem number

      iprob = 56

!   Prepare problem data
```

```
      call get_minlp_prob(     0,  iprob,     m,     me,   mmax,
     /                     ncont,   nbin,  nint,   nmax,      x,
     /                        xl,    xu,      f,      g,   pnam,
     /                      pref,   fex )

!  or call SPRING directly

!       call spring(     0,      m,    me,   mmax,  ncont,
!     /              nbin,   nint,  nmax,      x,     xl,
!     /                xu,      f,     g,   pnam,   pref,
!     /               fex )

      n = ncont + nbin + nint
      do i=ncont+1,n
         ideriv(i) = .false.
      end do

!   Set constants and tolerances for calling MISQP

      do i = 1,60
         ropt(i) = -1.d0
         iopt(i) = -1
         lopt(i) = .true.
      enddo
      iout    = 6       ! output channel
      iprint  = 2       ! print flag
      ifail   = 0       ! initialize flag
      maxit   = 1000    ! maximum number of iterations
      eps     = 1.0d-6  ! tolerance for forward differences
      acc     = 1.0d-6  ! final termination tolerance
      write(iout,*)
      write(iout,*) ' *** solving now ',pnam(1:10), ', fex =',fex

!   Begin of optimization block

! ----------------------------------------------------------------------
!   Call MISQP with reverse communication, integer variables treated as
!   non-relaxable

      ifail = 0
    1 continue

!   Evaluation of function values

      if ((ifail.eq.0).or.(ifail.eq.-1)) then
         call get_minlp_prob(     1,  iprob,     m,     me,   mmax,
     /                        ncont,   nbin,  nint,   nmax,      x,
     /                           xl,    xu,      f,      g,   pnam,
     /                         pref,   fex )

!   or call SPRING directly:

!       call spring(     1,      m,    me,   mmax,  ncont,
!     /              nbin,   nint,  nmax,      x,     xl,
!     /                xu,      f,     g,   pnam,   pref,
!     /               fex )
      endif

!   Approximation of partial derivatives subject to continuous
!   variables by forward differences

      if ((ifail.eq.0).or.(ifail.eq.-2)) then
         do i = 1, ncont
            xbck = x(i)
```

```
           x(i) = x(i) + eps
           call get_minlp_prob(     1,  iprob,     m,     me,   mmax,
     /                          ncont,   nbin,  nint,   nmax,      x,
     /                             xl,     xu,  feps,   geps,   pnam,
     /                           pref,    fex )

!   or call SPRING directly
!          call spring(     1,      m,    me,   mmax,  ncont,
!     /                  nbin,   nint,  nmax,      x,     xl,
!     /                    xu,   feps,  geps,   pnam,   pref,
!     /                   fex )

           df(i) = (feps - f)/eps
           do j = 1, m
              dg(j,i) = (geps(j) - g(j))/eps
           enddo
           x(i) = xbck
        enddo
      endif

!   Call driving routine

      call MISQP(      m,     me,   mmax,      n,   nbin,
     /              nint,      x,      f,      g,     df,
     /                dg,     xl,     xu,    acc,  maxit,
     /            maxcut, maxnde, iprint,   iout,  ifail,
     /            ideriv,   ropt,   iopt,   lopt,     rw,
     /              lerw,     iw,   leiw,     lw,   lelw )
      if (ifail.lt.0) goto 1

!   End of optimization block
! ----------------------------------------------------------------------

      stop
      end
```

The subsequent output is generated by MISQP. Note that objective function and constraint function values are scaled if not set otherwise.

```
     --------------------------------------------------------------------
                Start of the Mixed-Integer SQP Algorithm MISQP
                            Version 7.2 (Apr 2014)
     --------------------------------------------------------------------

   Parameters:

     Number of all variables:                      17   N
     Number of continuous variables:                5   NCONT
     Number of binary variables:                   11   NBIN
     Number of integer variables:                   1   NINT
     Number of all constraints:                     8   M
     Number of equality constraints:                5   ME
     Termination accuracy:                  0.100D-05   ACC
     Maximum number of iterations:                100   MAXIT
     Number of steps without progress:             10   MNFS
     Maximum number of nodes:                     100   MAXNDE
     Output level:                                  2   IPRINT
     Initial integer trust region radius:   0.900D+01   TRUSTI,ROPT( 7)

   Output in the following order:

     IT    - iteration number
     F     - objective function value
     MCV   - maximum constraint violation
```

```
    SIGMA - penalty parameter
    IL    - number inner loops
    DMAXC - maximum norm of continuous step D_C
    D1B   - 1-norm of binary step DELTA_B
    DMAXI - maximum norm of integer step D_I

  IT      F         MCV      SIGMA    IL   DMAXC      D1B      DMAXI
-----------------------------------------------------------------------
   1  0.10000D+01  0.10D+01  0.10D+04   1  0.75D+01  0.10D+01  0.00D+00
   2  0.37588D+01  0.69D+00  0.10D+04   2  0.33D+01  0.00D+00  0.00D+00
   3  0.16475D+01  0.59D+00  0.20D+04   2  0.17D+01  0.00D+00  0.00D+00
   4  0.50584D+00  0.48D+00  0.20D+04   2  0.83D+00  0.00D+00  0.00D+00
   5  0.14099D+00  0.16D+00  0.20D+04   1  0.33D+01  0.00D+00  0.00D+00
   6  0.32652D-01  0.29D+00  0.20D+04   2  0.32D+01  0.00D+00  0.10D+01

      .............

  40  0.43173D-04  0.24D+00  0.80D+04   1  0.22D+01  0.20D+01  0.80D+01
  41  0.38477D-04  0.23D+00  0.80D+04   1  0.19D+01  0.20D+01  0.50D+01
  42  0.47451D-04  0.78D-01  0.80D+04   1  0.19D+01  0.20D+01  0.10D+01
  43  0.80140D-04  0.97D-01  0.80D+04   1  0.11D+00  0.00D+00  0.10D+01
  44  0.72802D-04  0.20D-04  0.80D+04   2  0.11D-01  0.00D+00  0.00D+00
  45  0.72938D-04  0.67D-06  0.80D+04   2  0.93D+00  0.40D+01  0.20D+01


  --- FINAL CONVERGENCE ANALYSIS ---

    Objective function value:        F(X)  =  0.71831564D-04
    Approximation of solution:       X     =
         0.12230410D+01  0.28300000D+00  0.17857143D-02  0.43216998D+01
         0.13680931D+01  0.00000000D+00  0.00000000D+00  0.00000000D+00
         0.00000000D+00  0.10000000D+01  0.00000000D+00  0.00000000D+00
         0.00000000D+00  0.00000000D+00  0.00000000D+00  0.00000000D+00
         0.90000000D+01
    Constraint function values:      G(X)  =
         0.59409928D-15 -0.55789944D-08 -0.10890424D-09 -0.80249973D-12
         0.00000000D+00  0.53376308D-02  0.29523550D-01  0.15322656D-01
    Distances from lower bounds:     XL-X  =
        -0.80904103D+00 -0.76000000D-01  0.00000000D+00 -0.32216998D+01
        -0.36809312D+00  0.00000000D+00  0.00000000D+00  0.00000000D+00
         0.00000000D+00 -0.10000000D+01  0.00000000D+00  0.00000000D+00
         0.00000000D+00  0.00000000D+00  0.00000000D+00  0.00000000D+00
        -0.80000000D+01
    Distances from upper bounds:     XU-X  =
         0.87769590D+01  0.99717000D+02  0.18214286D-01  0.56783002D+01
         0.86319069D+01  0.10000000D+01  0.10000000D+01  0.10000000D+01
         0.10000000D+01  0.00000000D+00  0.10000000D+01  0.10000000D+01
         0.10000000D+01  0.10000000D+01  0.10000000D+01  0.10000000D+01
         0.10000000D+01
    Number of function calls:        NFUNC =      634
     - within TR method:             NF_TR =       58
     - integer derivatives:          NF_2D =      576
    Number of gradient calls:        NGRAD =       45
    Number of calls of QP solver:    NQL   =       74
     - 2nd order corrections:        NQL2  =       12
    Number of B&B nodes:             NODES =     1085
    Termination reason:              IFAIL =        0


  --- UNSCALED VALUES ---

    Objective function value:        F(X)  =  0.84624568D+00
    Constraint function values:      G(X)  =
         0.88817842D-15 -0.10655879D-07 -0.10890424D-09 -0.80249973D-10
         0.00000000D+00  0.10088102D+04  0.89456357D+01  0.14939590D+01
```

4 Numerical Results

A summary of numerical results obtained by the code MISQP of Exler, Lehmann, and Schittkowski [8, 9] is given below. With default tolerances and options, all problems are successfully solved, i.e., MISQP terminates with IFAIL=0 at a feasible solution, in most cases the same as the known one reported in the literature. Note that many test examples are non-convex and that the global solution is not known in all cases. Moreover, we are unable to specify the term *local solution* due to the lack of formal mathematical optimality conditions.

The Fortran codes are compiled by the Intel Composer XE 2013 Fortran Compiler under Windows 7 Professional and executed on an Intel Core(TM)i7-2720QM 64 bit CPU with 2.2 GHz and 8 GB RAM.

The following data are listed:

no	-	serial number
name	-	test problem name (PNAM)
n_{func}	-	number of equivalent function calls, i.e., all function calls including those needed for approximating partial derivatives,
$f(x^\star, y^\star)$	-	final objective function value,
$e(x^\star, y^\star)$	-	relative error of objective function value subject to the known one from literature,
$r(x^\star, y^\star)$	-	constraint violation at final solution,
time	-	average execution times in seconds.

Note that the number of function calls includes those which are used by a difference formula to approximate partial derivatives subject to continuous variables, and those to generate descent information for integer variables based on an adapted two-sided difference formula. In the latter case, partial derivatives are approximated at neighbored grid points only, i.e., we do not exploit the fact that all test problems are given by analytical expressions and that integer variables can be relaxed. The average number of iterations is 25 and the average number of equivalent function evaluations including those needed for approximating partial derivatives is 1,284. The average solution time is 2.0 sec.

Although all text problems are relaxable, the code MISQP only requires function values evaluated at integer points. However, it is possible to treat all test problems as continuous ones without any integer conditions, and to solve them by the continuous solver NLPQLP, see Schittkowski [18, 19]. NLPQLP needs 28 iterations and 1,174 function evaluations including those needed for approximating derivatives. Average execution time is 0.12 seconds.

It is important to note that the main design criterion behind MISQP is to develop a code for complex engineering applications, where calculation time for function evaluations is high and where the model functions are not composed of analytical expressions, which could otherwise be exploited. In particular, we do not identify special types of variables or constraints, say SOS variables, nor do we require that integer variables are relaxable.

Table 3: Individual Test Results for Mixed-Integer Problems

no	*name*	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	*time*
1	MITP1	367	-0.100097E+05	0.71E-09	0.00E+00	0.0150
2	MITP2	79	0.350000E+01	0.26E-10	0.20E-09	0.0000
3	QIP1	29	-0.200000E+02	0.00E+00	0.00E+00	0.0000
4	ASAADI11	88	-0.409574E+02	-0.20E-04	0.41E-09	0.0000
5	ASAADI12	112	-0.380000E+02	0.00E+00	0.00E+00	0.0000
6	ASAADI21	332	0.694903E+03	0.53E-08	0.00E+00	0.0160
7	ASAADI22	383	0.700000E+03	0.00E+00	0.00E+00	0.0160
8	ASAADI31	442	0.372190E+02	-0.14E-04	0.48E-10	0.0150
9	ASAADI32	203	0.430000E+02	0.00E+00	0.00E+00	0.0160
10	DIRTY	6411	-0.304669E+09	0.18E-03	0.11E-09	0.3900
11	BRAAK1	2346	0.100000E+01	0.37E-08	0.35E-11	0.0310
12	BRAAK2	2030	-0.271828E+01	-0.27E-06	0.11E-09	0.0470
13	BRAAK3	576	-0.196559E+07	0.65E-05	0.00E+00	0.0000
14	DEX2	33	-0.569375E+02	0.00E+00	0.00E+00	0.0000
15	FUEL	912	0.856612E+04	-0.23E-07	0.87E-06	0.0310
16	WP02	44	-0.244444E+01	-0.15E-05	0.00E+00	0.0160
17	NVS01	140	0.124697E+02	-0.95E-07	0.28E-12	0.0000
18	NVS02	489	0.596418E+01	-0.80E-07	0.18E-11	0.0150
19	NVS03	52	0.160000E+02	0.00E+00	0.00E+00	0.0000
20	NVS04	110	0.720000E+00	0.83E-14	0.00E+00	0.0000
21	NVS05	770	0.547093E+01	-0.40E-06	0.51E-06	0.0160
22	NVS06	64	0.177031E+01	0.28E-06	0.00E+00	0.0000
23	NVS07	22	0.400000E+01	0.00E+00	0.00E+00	0.0000
24	NVS08	138	0.234497E+02	0.42E-06	0.00E+00	0.0150
25	NVS09	32	-0.431343E+02	0.71E-07	0.00E+00	0.0000
26	NVS10	43	-0.310800E+03	-0.18E-15	0.00E+00	0.0000
27	NVS11	174	-0.431000E+03	0.00E+00	0.00E+00	0.0000
28	NVS12	425	-0.481200E+03	0.00E+00	0.00E+00	0.0160
29	NVS13	327	-0.585200E+03	0.00E+00	0.00E+00	0.0000
30	NVS14	301	-0.403582E+05	-0.12E-06	0.17E-10	0.0160
31	NVS15	54	0.100000E+01	0.00E+00	0.00E+00	0.0000
32	NVS16	28	0.703125E+00	0.00E+00	0.00E+00	0.0000
33	NVS17	448	-0.110040E+04	0.21E-15	0.00E+00	0.0150
34	NVS18	601	-0.778400E+03	0.15E-15	0.00E+00	0.0160
35	NVS19	629	-0.109840E+04	0.41E-15	0.00E+00	0.0150
36	NVS20	1845	0.230924E+03	0.58E-05	0.65E-08	0.0630
37	NVS21	281	-0.568478E+01	0.88E-07	0.00E+00	0.0000
38	NVS22	325	0.605822E+01	0.00E+00	0.47E-09	0.0150
39	NVS23	1413	-0.112520E+04	-0.20E-15	0.00E+00	0.0630
40	NVS24	2043	-0.103080E+04	0.23E-02	0.00E+00	0.1250
41	GEAR	248	0.100000E+01	0.45E-07	0.00E+00	0.0150
42	GEAR2	873	0.100000E+01	0.41E-06	0.11E-08	0.4680
43	GEAR2A	2907	0.100000E+01	0.62E-08	0.53E-09	2.2470
44	GEAR3	631	0.100000E+01	0.45E-07	0.15E-09	0.0310
45	GEAR4	1083	0.818403E+00	-0.50E+00	0.12E-04	0.2960
46	M3	1207	0.378000E+02	-0.13E-08	0.12E-08	0.0940
47	M6	7136	0.822855E+02	0.35E-03	0.24E-08	71.8690
48	M7	6266	0.106757E+03	-0.22E-06	0.16E-08	58.1410
49	FLOUDAS1	35	0.766718E+01	-0.41E-08	0.15E-09	0.0000
50	FLOUDAS2	41	0.107654E+01	0.29E-05	0.11E-09	0.0000
51	FLOUDAS3	274	0.457958E+01	-0.26E-07	0.15E-07	0.0160
52	FLOUDAS4	509	-0.946359E+00	0.16E-08	0.15E-09	0.0310
53	FLOUDAS40	52	-0.931588E+00	0.21E-05	0.96E-10	0.0000
54	FLOUDAS5	17	0.310000E+02	0.00E+00	0.00E+00	0.0000
55	FLOUDAS6	18	-0.170000E+02	-0.28E-10	0.53E-09	0.0000
56	SPRING	965	0.846246E+00	-0.21E-07	0.56E-08	0.0630

(continued)

no	$name$	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	$time$
57	DU_OPT5	3374	0.211636E+02	0.45E-01	0.00E+00	0.1240
58	DU_OPT	6107	0.420099E+01	-0.34E-05	0.00E+00	0.3440
59	ST_E13	16	0.223607E+01	-0.98E-08	0.00E+00	0.0000
60	ST_E14	958	0.457958E+01	-0.77E-09	0.18E-08	0.0460
61	ST_E15	34	0.766718E+01	0.81E-09	0.17E-08	0.0000
62	ST_E27	12	0.200000E+01	-0.70E-09	0.23E-09	0.0000
63	ST_E29	405	-0.934453E+00	0.96E-02	0.19E-08	0.0320
64	ST_E31	16114	-0.200000E+01	0.13E-09	0.63E-06	18.7040
65	ST_E32	569	-0.143041E+01	0.11E-07	0.89E-10	0.1400
66	ST_E35	1404	0.150156E+06	0.41E+00	0.21E-09	0.2810
67	ST_E36	188	-0.246000E+03	-0.24E-10	0.91E-09	0.0000
68	ST_E38	267	0.719773E+04	0.33E-11	0.15E-11	0.0000
69	ST_E40	31	0.304142E+02	0.00E+00	0.41E-10	0.0000
70	ST_MIQP1	30	0.281000E+03	0.00E+00	0.00E+00	0.0160
71	ST_MIQP2	65	0.200000E+01	0.00E+00	0.00E+00	0.0000
72	ST_MIQP3	13	-0.600000E+01	0.00E+00	0.00E+00	0.0000
73	ST_MIQP4	32	-0.457400E+04	-0.19E-09	0.30E-08	0.0000
74	ST_MIQP5	101	-0.333889E+03	0.33E-07	0.24E-09	0.0000
75	ST_TEST1	6	0.100000E+01	0.00E+00	0.00E+00	0.0000
76	ST_TEST2	42	-0.925000E+01	0.00E+00	0.00E+00	0.0000
77	ST_TEST3	59	-0.700000E+01	0.00E+00	0.00E+00	0.0000
78	ST_TEST4	63	-0.700000E+01	0.00E+00	0.00E+00	0.0000
79	ST_TEST5	44	-0.110000E+03	0.00E+00	0.00E+00	0.0150
80	ST_TEST6	158	0.471000E+03	0.00E+00	0.00E+00	0.0000
81	ST_TEST8	547	-0.295750E+05	0.10E-02	0.00E+00	0.0320
82	ST_TESTGR1	134	-0.127976E+02	0.11E-02	0.00E+00	0.0150
83	ST_TESTGR3	371	-0.205900E+02	0.00E+00	0.00E+00	0.1410
84	ST_TESTPH4	38	-0.805000E+02	0.00E+00	0.00E+00	0.0000
85	TLN2	123	0.530000E+01	0.00E+00	0.00E+00	0.0150
86	TLN4	1784	0.850000E+01	0.00E+00	0.00E+00	1.3110
87	TLN5	1817	0.116000E+02	0.94E-01	0.00E+00	1.7620
88	TLN6	1984	0.163000E+02	0.00E+00	0.00E+00	3.1520
89	NEJI	80	-0.111111E+02	-0.10E-07	0.00E+00	0.0000
90	TST_NAG	330	0.292500E+15	0.57E-07	0.92E-13	0.0150
91	TLOSS	1594	0.163000E+02	0.00E+00	0.00E+00	3.4170
92	TLTR	1228	0.482708E+02	0.42E-02	0.00E+00	1.9960
93	MEANVARX	481	0.141897E+02	-0.29E-08	0.15E-10	0.0470
94	MINLPHIX	2168	0.316693E+03	-0.22E-07	0.50E-07	2.1060
95	MIP_EX	42	0.350000E+01	0.00E+00	0.00E+00	0.0160
96	MGRID_CYCLES1	64	0.800000E+01	0.00E+00	0.00E+00	0.0000
97	MGRID_CYCLES2	493	0.300000E+03	0.00E+00	0.00E+00	0.0310
98	CROP5	66	0.953099E-01	-0.32E-08	0.00E+00	0.0000
99	CROP20	4356	0.111652E+00	0.34E-03	0.00E+00	2.3240
100	CROP50	6137	0.324195E+00	-0.15E-03	0.00E+00	2.7300
101	CROP100	4842	0.851471E+00	0.56E-08	0.00E+00	2.9800
102	SPLITF1	214	-0.160449E+04	0.35E-05	0.21E-09	0.0160
103	SPLITF2	2105	-0.180000E+04	-0.12E-11	0.29E-09	1.2480
104	SPLITF3	1851	-0.240739E+04	0.40E-01	0.15E-09	0.7800
105	SPLITF4	2418	-0.262493E+04	0.63E-03	0.18E-09	1.0920
106	SPLITF5	1849	-0.280449E+04	0.20E-05	0.21E-09	0.5150
107	SPLITF6	924	-0.309953E+04	-0.10E-04	0.11E-09	0.3120
108	SPLITF7	5491	-0.250909E+04	0.41E-01	0.12E-09	12.7760
109	SPLITF8	2616	-0.304004E+04	0.19E-03	0.26E-09	2.7770
110	SPLITF9	1646	-0.340449E+04	0.17E-05	0.17E-09	1.5910
111	ELF	690	0.434667E+00	0.13E+01	0.61E-08	3.3700
112	SPECTRA2	6707	0.139783E+02	-0.41E-06	0.16E-07	47.0490
113	WINDFAC	3321	0.254487E+00	-0.68E-08	0.19E-10	0.1410

(continued)

no	*name*	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	*time*
114	CSCHED1	1560	-0.292792E+05	0.22E+00	0.11E-07	0.3580
115	ALAN	360	0.292500E+01	0.90E-02	0.10E-09	0.0160
116	PUMP	1362	0.166087E+06	0.24E+00	0.36E-06	0.1720
117	RAVEM	5704	0.269582E+06	-0.32E-04	0.13E-06	27.9550
118	ORTEZ	4549	-0.102055E+05	-0.49E-08	0.25E-08	4.9610
119	EX1221	35	0.766718E+01	-0.22E-10	0.10E-09	0.0000
120	EX1222	47	0.107654E+01	0.98E-07	0.11E-09	0.0000
121	EX1223	275	0.457958E+01	0.69E-07	0.57E-07	0.0150
122	EX1223A	150	0.457958E+01	0.83E-07	0.11E-07	0.0000
123	EX1223B	256	0.457958E+01	0.11E-07	0.17E-06	0.0160
124	EX1224	391	-0.934453E+00	0.96E-02	0.11E-07	0.0150
125	EX1225	80	0.310000E+02	-0.55E-10	0.70E-09	0.0160
126	EX1226	21	-0.170000E+02	-0.92E-10	0.70E-09	0.0000
127	EX1233	5359	0.155522E+06	0.33E-02	0.25E-07	3.2600
128	EX1243	2873	0.129302E+06	0.55E+00	0.78E-06	2.2940
129	EX1244	4348	0.105231E+06	0.28E+00	0.83E-07	10.2180
130	EX1252	5342	0.128894E+06	0.32E-06	0.67E-08	6.8170
131	EX1263	1786	0.243000E+02	0.24E+00	0.45E-08	17.0200
132	EX1263A	780	0.196000E+02	0.00E+00	0.00E+00	0.5300
133	EX1264	1530	0.170000E+02	0.98E+00	0.62E-09	9.5160
134	EX1264A	813	0.860000E+01	0.00E+00	0.00E+00	1.2320
135	EX1265	1618	0.145000E+02	0.41E+00	0.12E-09	19.6410
136	EX1265A	958	0.103000E+02	0.00E+00	0.00E+00	1.1850
137	DIOPHE	81	-0.100000E+01	0.50E+00	0.00E+00	0.0470
138	EX1266A	345	0.163000E+02	0.00E+00	0.00E+00	0.0630
139	GBD	30	0.220000E+01	0.00E+00	0.00E+00	0.0000
140	EX3	5425	0.680097E+02	-0.39E-05	0.26E-06	1.0770
141	EX4	869	-0.731327E+01	0.93E-01	0.33E-08	0.2650
142	FAC1	1013	0.160913E+09	0.76E-07	0.18E-07	0.1710
143	FAC2	6962	0.331871E+09	0.10E-03	0.59E-07	2.8080
144	FAC3	9584	0.319823E+08	-0.49E-08	0.12E-09	6.3030
145	GKOCIS	373	-0.192310E+01	-0.12E-05	0.69E-06	0.0150
146	KG	143	0.103938E+03	-0.42E-07	0.24E-09	0.0000
147	SYNTHES1	185	0.598177E+01	-0.47E-02	0.30E-08	0.0000
148	SYNTHES2	623	0.730353E+02	0.36E-07	0.58E-09	0.0320
149	SYNTHES3	878	0.680097E+02	-0.10E-05	0.93E-06	0.0460
150	PARALLEL	1035	0.217293E+05	0.25E+02	0.71E-06	1.0460
151	SYNHEAT	6698	0.195965E+06	0.26E+00	0.36E-06	4.1490
152	SEP1	532	-0.532125E+03	0.10E-10	0.52E-09	0.0150
153	DAKOTA	114	0.136340E+01	-0.25E-06	0.23E-07	0.0160
154	BATCH	1816	0.285506E+06	0.16E-05	0.36E-07	2.3400
155	BATCHDES	483	0.167412E+06	-0.96E-04	0.58E-07	0.0160
156	ENIPLAC	23548	-0.130147E+06	0.13E-01	0.70E-08	272.8750
157	PROB02	143	0.112235E+06	0.00E+00	0.00E+00	0.0000
158	PROB03	15	0.100000E+02	0.00E+00	0.00E+00	0.0000
159	PROB10	20	0.344550E+01	-0.12E-10	0.29E-10	0.0000
160	NOUS1	5463	0.158273E+01	0.10E-01	0.78E-14	2.1530
161	NOUS2	6060	0.269717E+01	0.33E+01	0.14E-06	1.3890
162	TLS2	1167	0.530000E+01	0.00E+00	0.10E-09	1.7160
163	TLS4	2409	0.200000E+02	0.14E+01	0.25E-09	34.8970
164	TLS5	4842	0.206000E+02	0.94E+00	0.85E-08	35.4430
165	OAER	176	-0.192310E+01	0.25E-06	0.35E-09	0.0000
166	PROCSEL	362	-0.192310E+01	0.14E-06	0.40E-09	0.0160
167	LICHOU_1	358	-0.246000E+03	-0.41E-10	0.26E-08	0.0000
168	LICHOU_2	51	0.719801E+04	0.99E-02	0.23E-11	0.0000
169	LICHOU_3	51	0.304142E+01	0.70E-05	0.00E+00	0.0150
170	WU_1	66	0.320000E+00	0.87E-15	0.00E+00	0.0000

(continued)

no	*name*	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	*time*
171	WU_2	99	0.976000E+01	0.55E-15	0.00E+00	0.0000
172	WU_3	130	0.137884E+00	0.12E-08	0.00E+00	0.0310
173	WU_4	261	0.102400E+02	0.00E+00	0.00E+00	0.0940
174	OPTPRLOC	1058	-0.806414E+01	-0.48E-05	0.71E-07	0.4520
175	GASNET	27771	0.103629E+08	0.48E+00	0.30E-09	11.9030
176	TP83	145	-0.306060E+05	0.25E-08	0.17E-10	0.0160
177	TP84	91	-0.571515E+07	-0.65E-08	0.00E+00	0.0000
178	TP85	221	-0.189579E+01	-0.71E-07	0.57E-08	0.0000
179	TP87	126	0.895823E+04	0.31E-08	0.21E-06	0.0000
180	TP93	1087	0.138741E+03	-0.39E-06	0.37E-06	0.0310
181	FEEDTRAY	25487	-0.134060E+02	0.20E-06	0.12E-07	13.0420
182	FEEDTRAY2	3560	0.100000E+01	0.50E-07	0.16E-07	3.0260
183	HILBERT20	1169	0.210000E+03	0.00E+00	0.31E-15	0.7180
184	HILBERT50	3635	0.127500E+04	0.00E+00	0.50E-15	3.9470
185	HILBERT100	6634	0.505000E+04	0.00E+00	0.53E-15	16.0360
186	SLOPPY	743	0.960870E-06	0.96E-06	0.00E+00	0.0160
187	RASTRIGIN	428	-0.196885E+01	0.25E-04	0.00E+00	0.0000
188	EMSO	120	-0.192310E+01	-0.21E-07	0.11E-09	0.0000
189	TP1	45	0.000000E+00	0.00E+00	0.00E+00	0.0000
190	TP1A	45	0.000000E+00	0.00E+00	0.00E+00	0.0000
191	TP1B	26	0.000000E+00	0.00E+00	0.00E+00	0.0000
192	TP9	25	-0.500000E+00	0.00E+00	0.00E+00	0.0000
193	TP10	101	-0.100000E+01	0.00E+00	0.00E+00	0.0000
194	DEB10	18462	0.198801E+03	-0.42E-08	0.74E-07	70.2620
195	IRAP1	6142	0.218136E+03	-0.19E-04	0.00E+00	0.7490
196	IRAP2	1349	0.316516E+03	0.81E-08	0.00E+00	0.1090
197	IRAP3	1426	0.336613E+03	-0.82E-08	0.00E+00	0.1090
198	IRAP4	152	0.196164E+03	-0.75E-08	0.00E+00	0.0160
199	IRAP5	197	0.186745E+03	0.18E-07	0.00E+00	0.0310
200	IRAP6	1143	0.239359E+03	-0.16E-07	0.00E+00	0.0310

References

[1] Asaadi J. (1973): *A computational comparison of some non-linear programs*, Mathematical Programming, Vol. 4, 144–154

[2] van de Braak G. (2001): *Das Verfahren MISQP zur gemischt ganzzahligen nichtlinearen Programmierung für den Entwurf elektronischer Bauteile*, Diploma Thesis, Department of Numerical and Instrumental Mathematics, University of Münster, Germany

[3] Bussieck M.R., Drud A.S., Meeraus A. (2007): *MINLPLib - A collection of test models for mixed-integer nonlinear programming*, GAMS Development Corp., Washington D.C., USA

[4] Cha J.Z., Mayne R.W. (1989): *Optimization with discrete variables via recursive quadratic programming: Part 2 - algorithms and results*, Transactions of the ASME, Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, 130–136

[5] M.S. Chern (1992): *On the computational complexity of reliability redundancy allocation in a series system*, Operation Research Letters, Vol. 5, 309-315

[6] Duran M., Grossmann I.E. (1986): *An outer-approximation algorithm for a class of mixed-integer nonlinear programs*, Mathematical Programming, Vol. 36, 307–339

[7] Eldred M.S., Giunta A.A.,van Bloemen Waanders B.G., Wojtkiewicz S.F., Hart W.E., Alleva M.D. (2002): *DAKOTA: A multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis, Version 3.1 Users Manuel*, Report SAND20001-3796, Sandia National Laboratories, PO Box 5800, Albuquerque, NM 87185-0847

[8] Exler O., Schittkowski K. (2006): *A trust region SQP algorithm for mixed-integer nonlinear programming*, Optimization Letters, Vol. 1, 269-280

[9] Exler O., Lehmann T., Schittkowski K. (2011): *MISQP: A Fortran implementation of a trust region SQP algorithm for mixed-integer nonlinear programming - User's guide*, Report, Department of Computer Science, University of Bayreuth, Germany

[10] Exler O., Lehmann T., Schittkowski K. (2012): *A comparative study of SQP-type algorithms for nonlinear and non-convex mixed-integer optimization*, toappear: Mathematical Programming Computation

[11] Floudas C.A., Pardalos P.M., Adjiman C.S., Esposito W.R., Gumus Z.H., Harding S.T., Klepeis J.L., Meyer C.A., Schweiger C.A. (1999): *Handbook of Test Problems in Local and Global Optimization*, Kluwer Academic Publishers

[12] Gago-Vargas J., Hartillo-Hermoso M.I., Puerto-Albandoz J., Ucha-Entiques J.M., *Exact cost minimization of a series-parallel reliable system with multiple component choices using an algebraic method*, Report, University of Sevilla, Spain

[13] Grossmann I.E., Kravanja Z. (1997): *Mixed-integer nonlinear programming: A survey of algorithms and applications*, in: Conn A.R., Biegler L.T., Coleman T.F., Santosa F.N. (eds.): *Large-Scale Optimization with Applications, Part II: Optimal Design and Control*, Springer, New York, Berlin

[14] Hock W., Schittkowski K. (1981): *Test Examples for Nonlinear Programming Codes,* Lecture Notes in Economics and Mathematical Systems, Vol. 187, Springer

[15] Kocis G.R., Grossmann I.E. (1988): *Global Optimization of Non-Convex MINLP Problems in Process Synthesis*, Industrial and Engineering Chemical Research, Vol. 27, 1407–1421

[16] Li H.-L., Chou C.-T. (1994): *A global approach for nonlinear mixed discrete programming in design optimization*, Engineering Optimization, Vol. 22, 109–122

[17] Maniezzo V., Stützle T., Voß S. (2009): *A good recipe for solving MINLPs*, in: Metaheuristics, Ser. Annals of Information Systems, Vol. 10, 231-244, Springer

[18] Schittkowski K. (1983): *On the convergence of a sequential quadratic programming method with an augmented Lagrangian search direction,* Optimization, Vol. 14, 197-216

[19] Schittkowski K. (1985/86): *NLPQL: A Fortran subroutine solving constrained nonlinear programming problems,* Annals of Operations Research, Vol. 5, 485-500

[20] Still C., Westerlund T. (2006): *Solving convex MINLP optimization problems using a sequential cutting plane algorithm*, Computational Optimization and Applications, Vol. 34, 63-83

[21] Sun X., Ruan N., Li D. (2006): *An efficient algorithm for nonlienar integer programming problems arising in series-parallel reliability systems*, Optimization Methods and Software, Vol. 21, 617-634

[22] Westerlund T., Pörn R. (2002): *Solving pseudo-convex mixed integer optimization problems by cutting plane techniques*, Optimization and Engineering, Vol. 3, 253-280

[23] Thekale A., Gradl T., Klamroth K., Rüde U. (2009): *Optimizing the number of multigrid cycles in the full multigrid algorithm*, Lehrstuhlbericht 09-5, Friedrich-Alexander-Universität Erlangen-Nürnberg, Institut für Informatik

[24] Wu Z. (1994): *A subgradient algorithm for nonlinear integer programming*, MCS-P454-0794, Technical Report, Argonne National Laboratory