A Collection of 186 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: May 18, 2012

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 186 Fortran subroutines for a set of non-convex test problems, where most of them are taken from existing collections, especially from the GAMS collection MINLPLib. For each test example, relevant problem data like number of variables or constraints and code for function evaluation are summarized in a single file. 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 included together with the individual source codes of the collection. The implementation is thread-safe basic Fortran (38,000 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 the mixed-integer problems directly.

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 [2] 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 [16] or Maniezzo, Stützle, and Voß [15]. They are implemented in the GAMS modeling language 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 (38,000 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 codes sometimes stop at a feasible solution which is not optimal, although the internal stopping and optimization test 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 $y \in \mathbb{R}^{n_i}$.

We do not provide partial derivatives. In a comparative study of Exler, Lehmann and Schittkowski [9], derivatives subject to integer and boolean variables are approximated by a difference formula evaluated at grid points, and partial derivatives subject to continuous variables are approximated by a forward differences. In this case, we do exploit the fact that the test problems are relaxable.

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 driver program is listed that shows how a nonlinear mixed-integer code, in this case the code MISQP of Exler, Lehmann, and Schittkowski [7, 8], can be implemented. Section 4 contains a list of all individual 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	[4]	4	3	0	0	2	0.10000E+01
12	BRAAK2	[4]	4	3	0	0	4	-0.27183E+01
13	BRAAK3	[4]	4	3	0	0	4	-0.19656E+07
14	DEX2	[5]	0	2	0	0	2	-0.56938E+02
15	FUEL	[2]	12	0	3	6	15	0.85661E+04
16	WP02	[18]	1	1	0	0	2	-0.24444E+01
17	NVS01	[2]	1	2	0	1	3	0.12470E+02
18	NVS02	[2]	3	5	0	3	3	0.59642E+01
19	NVS03	[2]	0	2	0	0	2	0.16000E+02
20	NVS04	[2]	0	2	0	0	0	0.72000E+00
21	NVS05	[2]	6	2	0	4	9	0.54709E+01
22	NVS06	[2]	0	2	0	0	0	0.17703E+01
23	NVS07	[2]	0	3	0	0	2	0.40000E+01
24	NVS08	[2]	1	2	0	0	3	0.23450E+02
25	NVS09	[2]	0	10	0	0	0	-0.43134E+02
26	NVS10	[2]	0	2	0	0	2	-0.31080E+03
27	NVS11	[2]	0	3	0	0	2	-0.43100E+03
28	NVS12	[2]	0	4	0	0	4	-0.48120E+03
29	NVS13	[2]	0	5	0	0	5	-0.58520E+03
30	NVS14	[2]	3	5	0	3	3	-0.40358E+05
31	NVS15	[2]	0	3	0	0	1	0.10000E+01
32	NVS16	[2]	0	2	0	0	0	0.70312E+00
33	NVS17	[2]	0	7	0	0	7	-0.11004E+04
34	NVS18	[2]	0	6	0	0	6	-0.77840E+03
35	NVS19	[2]	0	8	0	0	8	-0.10984E+04
36	NVS20	[2]	11	5	0	0	8	0.23092E+03
37	NVS21	[2]	1	2	0	0	2	-0.56848E+01
38	NVS22	[2]	4	4	0	4	9	0.60582E+01
39	NVS23	[2]	0	9	0	0	9	-0.11252E+04
40	NVS24	[2]	0	10	0	0	10	-0.10332E+04
41	GEAR	[2]	0	4	0	0	0	0.10000E+01
42	GEAR2	[2]	4	24	0	4	4	0.10000E+01
43	GEAR2A	[2]	4	0	24	4	4	0.10000E+01
44	GEAR3	[2]	4	4	0	4	4	0.10000E+01
45	GEAR4	[2]	2	4	0	1	1	0.16434E+01
46	M3	[2]	20	0	6	0	43	0.37800E+02
47	M6	[2]	56	0	30	0	157	0.82257E+02
48	M7	[2]	72	0	42	0	211	0.10676E+03
49	FLOUDAS1	[10]	2	0	3	2	5	0.76672E+01
50	FLOUDAS2	[10]	2	0	1	0	3	0.10765E+01
51	FLOUDAS3	[10]	3	0	4	0	9	0.45796E+01
52	FLOUDAS4	[10]	3	0	8	3	7	-0.94347E+00
53	FLOUDAS40	[10]	3	0	8	3	7	-0.94347E+00

(continued)

no	*name*	*ref*	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
54	FLOUDAS5	[10]	0	2	0	0	4	0.31000E+02
55	FLOUDAS6	[10]	1	1	0	0	3	-0.17000E+02
56	SPRING	[2]	5	1	11	5	8	0.84625E+00
57	DU_OPT5	[2]	7	13	0	0	9	0.80737E+01
58	DU_OPT	[2]	7	13	0	0	9	0.35563E+01
59	ST_E13	[2]	1	0	1	0	2	0.20000E+01
60	ST_E14	[2]	7	0	4	4	13	0.45796E+01
61	ST_E15	[2]	2	0	3	2	5	0.76672E+01
62	ST_E27	[2]	2	0	2	0	6	0.20000E+01
63	ST_E29	[2]	3	0	8	2	7	-0.94347E+00
64	ST_E31	[2]	88	0	24	81	135	-0.20000E+01
65	ST_E32	[2]	16	19	0	17	18	-0.14304E+01
66	ST_E35	[2]	25	0	7	15	39	0.64868E+05
67	ST_E36	[2]	1	1	0	1	2	-0.24600E+03
68	ST_E38	[2]	2	2	0	0	3	0.71977E+04
69	ST_E40	[2]	1	3	0	4	8	0.30414E+02
70	ST_MIQP1	[2]	0	0	5	0	1	0.28100E+03
71	ST_MIQP2	[2]	0	4	0	0	3	0.20000E+01
72	ST_MIQP3	[2]	0	2	0	0	1	-0.60000E+01
73	ST_MIQP4	[2]	3	0	3	0	4	-0.45740E+04
74	ST_MIQP5	[2]	5	2	0	0	13	-0.33389E+03
75	ST_TEST1	[2]	0	5	0	0	1	0.10000E+01
76	ST_TEST2	[2]	0	6	0	0	2	-0.92500E+01
77	ST_TEST3	[2]	0	13	0	0	10	-0.70000E+01
78	ST_TEST4	[2]	0	6	0	0	5	-0.70000E+01
79	ST_TEST5	[2]	0	10	0	0	11	-0.11000E+03
80	ST_TEST6	[2]	0	0	10	0	5	0.47100E+03
81	ST_TEST8	[2]	0	24	0	0	20	-0.29605E+05
82	ST_TESTGR1	[2]	0	10	0	0	5	-0.12812E+02
83	ST_TESTGR3	[2]	0	20	0	0	20	-0.20590E+02
84	ST_TESTPH4	[2]	0	3	0	0	10	-0.80500E+02
85	TLN2	[2]	0	6	2	0	12	0.53000E+01
86	TLN4	[2]	0	20	4	0	24	0.83000E+01
87	TLN5	[2]	0	30	5	0	30	0.10300E+02
88	TLN6	[2]	0	42	6	0	36	0.15300E+02
89	TLN7	[2]	0	56	7	0	42	0.19500E+02
90	TLN12	[2]	0	156	12	0	72	0.90500E+02
91	TLOSS	[2]	0	42	6	0	53	0.16300E+02
92	TLTR	[2]	0	36	12	0	54	0.48067E+02
93	MEANVARX	[2]	21	0	14	8	44	0.14369E+02
94	MINLPHIX	[2]	64	0	20	30	92	0.31669E+03
95	MIP_EX	[11]	2	0	3	0	7	0.35000E+01
96	MGRID_CYCLES1	[19]	0	5	0	0	1	0.80000E+01
97	MGRID_CYCLES2	[19]	0	10	0	0	1	0.30000E+03
98	CROP5	[17]	0	5	0	0	3	0.95310E-01
99	CROP20	[17]	0	20	0	0	3	0.11166E+00
100	CROP50	[17]	0	50	0	0	3	0.32424E+00
101	CROP100	[17]	0	100	0	0	3	0.85147E+00
102	SPLITF1	[9]	3	0	9	3	9	-0.16045E+04
103	SPLITF2	[9]	6	0	18	6	15	-0.18000E+04
104	SPLITF3	[9]	6	0	18	6	15	-0.25083E+04
105	SPLITF4	[9]	6	0	18	6	15	-0.26266E+04
106	SPLITF5	[9]	6	0	18	6	15	-0.28045E+04
107	SPLITF6	[9]	6	0	18	6	15	-0.30995E+04
108	SPLITF7	[9]	9	0	27	9	21	-0.26310E+04
109	SPLITF8	[9]	9	0	27	9	21	-0.30406E+04
110	SPLITF9	[9]	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	[2]	30	0	24	6	38	0.19167E+00
112	SPECTRA2	[2]	39	0	30	9	72	0.13978E+02
113	WINDFAC	[2]	11	3	0	13	13	0.25449E+00
114	CSCHED1	[2]	13	0	63	12	22	-0.37604E+05
115	ALAN	[2]	4	0	4	2	7	0.28990E+01
116	PUMP	[2]	15	6	3	13	34	0.13426E+06
117	RAVEM	[2]	58	0	54	25	186	0.26959E+06
118	ORTEZ	[2]	69	0	18	24	74	-0.95320E+04
119	EX1221	[2]	2	0	3	2	5	0.76672E+01
120	EX1222	[2]	2	0	1	0	3	0.10765E+01
121	EX1223	[2]	7	0	4	4	13	0.45796E+01
122	EX1223A	[2]	3	0	4	0	9	0.45796E+01
123	EX1223B	[2]	3	0	4	0	9	0.45796E+01
124	EX1224	[2]	3	0	8	2	7	-0.94347E+00
125	EX1225	[2]	2	0	6	2	10	0.31000E+02
126	EX1226	[2]	2	0	3	1	5	-0.17000E+02
127	EX1233	[2]	40	0	12	20	64	0.15501E+06
128	EX1243	[2]	52	0	16	24	96	0.83403E+05
129	EX1244	[2]	72	0	23	30	129	0.82043E+05
130	EX1252	[2]	24	0	15	22	43	0.12889E+06
131	EX1252A	[2]	15	6	3	13	34	0.12889E+06
132	EX1263	[2]	20	0	72	20	55	0.19600E+02
133	EX1263A	[2]	0	20	4	0	35	0.19600E+02
134	EX1264	[2]	20	0	68	20	55	0.86000E+01
135	EX1264A	[2]	0	20	4	0	35	0.86000E+01
136	EX1265	[2]	30	0	100	30	74	0.10300E+02
137	EX1265A	[2]	0	30	5	0	44	0.10300E+02
138	EX1266	[2]	42	0	138	42	95	0.16300E+02
139	EX1266A	[2]	0	42	6	0	53	0.16300E+02
140	GBD	[2]	1	0	3	0	4	0.22000E+01
141	EX3	[2]	24	0	8	17	31	0.68010E+02
142	EX4	[2]	11	0	25	0	30	-0.80641E+01
143	FAC1	[2]	16	0	6	10	18	0.16091E+09
144	FAC2	[2]	54	0	12	21	33	0.33184E+09
145	FAC3	[2]	54	0	12	21	33	0.31982E+08
146	GKOCIS	[2]	8	0	3	5	8	-0.19231E+01
147	KG	[13]	7	0	2	5	9	0.99200E+02
148	SYNTHES1	[2]	3	0	3	0	6	0.60098E+01
149	SYNTHES2	[2]	6	0	5	1	14	0.73035E+02
150	SYNTHES3	[2]	9	0	8	2	23	0.68010E+02
151	PARALLEL	[2]	180	0	25	81	115	0.92430E+03
152	SYNHEAT	[2]	44	0	12	20	64	0.15500E+06
153	SEP1	[2]	27	0	2	22	31	-0.51008E+03
154	DAKOTA	[3]	2	2	0	0	2	0.13634E+01
155	BATCH	[2]	23	0	24	12	73	0.28551E+06
156	BATCHDES	[2]	10	0	9	6	19	0.16743E+06
157	ENIPLAC	[2]	117	0	24	87	189	-0.13186E+06
158	PROB02	[2]	0	6	0	0	8	0.11224E+06
159	PROB03	[2]	0	2	0	0	1	0.10000E+02
160	PROB10	[2]	1	1	0	0	2	0.34455E+01
161	NOUS1	[2]	48	0	2	41	43	0.15671E+01
162	NOUS2	[2]	48	0	2	41	43	0.62597E+00
163	TLS2	[2]	4	2	31	6	24	0.53000E+01
164	TLS4	[2]	16	4	85	20	64	0.12400E+02
165	TLS5	[2]	25	5	131	30	90	0.14200E+02
166	TLS6	[2]	36	6	173	42	120	0.15300E+02
167	OAER	[2]	6	0	3	3	7	-0.19231E+01

(continued)

no	*name*	*ref*	n_c	n_d	n_b	m_e	m	$f(x^\star, y^\star)$
168	PROCSEL	[2]	7	0	3	4	7	-0.19231E+01
169	LICHOU_1	[14]	1	1	0	1	2	-0.24600E+03
170	LICHOU_2	[14]	2	2	0	0	4	0.71273E+04
171	LICHOU_3	[14]	0	3	0	0	4	0.30414E+01
172	WU_1	[20]	0	0	32	0	0	0.32000E+00
173	WU_2	[20]	0	0	32	0	0	0.97600E+01
174	WU_3	[20]	0	0	64	0	0	0.13788E+00
175	WU_4	[20]	0	0	64	0	0	0.10240E+02
176	OPTPRLOC	[6]	5	0	25	0	30	-0.80641E+01
177	GASTRANS	[2]	86	0	21	0	149	0.89086E-02
178	GASNET	[2]	81	0	10	48	69	0.69994E+07
179	TP83	[12]	1	4	0	0	6	-0.30606E+05
180	TP84	[12]	3	2	0	0	6	-0.57152E+07
181	TP85	[12]	2	3	0	0	38	-0.18958E+01
182	TP87	[12]	4	2	0	4	4	0.89582E+04
183	TP93	[12]	5	1	0	0	2	0.13874E+03
184	FEEDTRAY	[2]	90	0	7	83	91	-0.13406E+02
185	FEEDTRAY2	[2]	51	0	36	6	283	0.10000E+01
186	DEB10	[2]	160	0	22	65	129	0.20943E+03

For GASTRANS, we did not succeed to find a feasible point, likely due to human error when transferring the equations from GAMS to Fortran. The constraints are relaxed subject to one additional continuous variable, which is also added to the objective function together with a penalty factor.

3 The Fortran Subroutines

This section describes the organization of the Fortran subroutines 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`

To allow computation of relative errors in function values, a given optimal function value 0 (FEX) is replaced by one, i.e., the value 1 is added to the objective function. The modified examples are GEAR, GEAR2, GEAR2a, and GEAR3.

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, a 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 :	Input of an available problem number by which a specific test problem with this serial number is to be evaluated.
M :	Number of all constraints, without bounds.
ME :	Number of all equality constraints.
MMAX :	Dimension of G. MMAX has to be at least one and at least M for the largest problem to be executed.
NCONT :	Number of all continuous variables.
NBIN :	Number of all binary variables.
NINT :	Number of all integer variables.
NMAX :	Dimension of X, XL, and XU. NMAX has to be at least two and at least NCONT+NBIN+NINT for the largest problem to be executed.
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), XU(NMAX):	On return, the one-dimensional arrays XL and XU contain the lower and 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 at X.
G(MMAX) :	When called with MODE=1, the double precision array G contains the constraint function values G(1),...,G(M) computed at 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, 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 some auxiliary routines required to execute test problems, mostly some adapted GAMS routines:

```
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 of the individual test problems 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 [2],

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

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 follow the implementation of the GAMS 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 = 10000)
      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)

c   Set test problem number and prepare initial data

      iprob = 56

      iprob=11


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

```
c  or call SPRING directly

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

c   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
      n = ncont + nbin + nint
      do i=ncont+1,n
         ideriv(i) = .false.
      enddo
      write(iout,*)
      write(iout,*) ' *** solving now ',pnam(1:10), ', fex =',fex

c   Begin of optimization block

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

      ifail = 0
    1 continue

c   Evaluation of function values

      if ((ifail.eq.0).or.(ifail.eq.-1)) then

c   call through interface

c         call get_minlp_prob(     1,  iprob,     m,     me,   mmax,
c     /                        ncont,   nbin,  nint,   nmax,      x,
c     /                           xl,     xu,     f,      g,   pnam,
c     /                         pref,    fex )

c   or call SPRING directly

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

c   approximation of partial derivatives subject to continuous
c   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

c   call through interface
```

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

c   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

c   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

c   End of optimization block
c ----------------------------------------------------------------------

      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 CODE MISQP
   ------------------------------------------------------------

   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
     Total number of constraints:                   8   M
     Number of equality constraints:                5   ME
     Termination accuracy:                  0.100D-05   ACC
     Maximum number of iterations:                300   MAXIT
     Maximum number of QP cuts:                   500   MAXCUT
     Maximum number of nodes:                    1000   MAXNDE
     Output level:                                  2   IPRINT

   Non-default options:

     Termination accuracy of MIQP solver:   0.100D-08   ACCQP, ROPT( 1)
     Initial integer trust region radius:   0.900D+01   TRUSTI,ROPT( 7)
     Internal scaling suppressed:                   0   SCALE, IOPT( 1)

   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
-----------------------------------------------------------------------
  0  0.10000D+01  0.10D+01  0.10D+04   0  0.00D+00  0.00D+00  0.00D+00
  1  0.37588D+01  0.69D+00  0.10D+04   1  0.75D+01  0.10D+01  0.00D+00
  2  0.16475D+01  0.59D+00  0.20D+04   2  0.33D+01  0.00D+00  0.00D+00
  3  0.50584D+00  0.48D+00  0.20D+04   2  0.17D+01  0.00D+00  0.00D+00
  4  0.14099D+00  0.16D+00  0.20D+04   2  0.83D+00  0.00D+00  0.00D+00
  5  0.32652D-01  0.29D+00  0.20D+04   1  0.33D+01  0.00D+00  0.00D+00
  6  0.84068D-02  0.51D+00  0.20D+04   2  0.32D+01  0.00D+00  0.10D+01
  7  0.14650D-02  0.41D+00  0.20D+04   2  0.13D+01  0.00D+00  0.00D+00
  8  0.33704D-04  0.33D+00  0.20D+04   1  0.17D+01  0.00D+00  0.10D+01
  9  0.19260D-04  0.27D-01  0.20D+04   1  0.66D+00  0.00D+00  0.10D+01
 10  0.18699D-04  0.18D-02  0.20D+04   1  0.95D-02  0.00D+00  0.00D+00

 ...

 39  0.43173D-04  0.24D+00  0.80D+04   1  0.24D+01  0.20D+01  0.50D+01
 40  0.38477D-04  0.23D+00  0.80D+04   1  0.22D+01  0.20D+01  0.80D+01
 41  0.47451D-04  0.78D-01  0.80D+04   1  0.19D+01  0.20D+01  0.50D+01
 42  0.80140D-04  0.97D-01  0.80D+04   1  0.19D+01  0.20D+01  0.10D+01
 43  0.72802D-04  0.20D-04  0.80D+04   1  0.11D+00  0.00D+00  0.10D+01
 44  0.72938D-04  0.67D-06  0.80D+04   2  0.11D-01  0.00D+00  0.00D+00

 --- 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)  =
        0.00000000D+00 -0.55410086D-08 -0.10834160D-09 -0.80250001D-11
        0.00000000D+00  0.53376264D-02  0.29523550D-01  0.15322656D-01
   Distances from lower bounds:     XL-X  =
       -0.80904103D+00 -0.75999999D-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 =     1168
   Termination reason:              IFAIL =        0
```

4 Numerical Results

A summary of numerical results obtained by the code MISQP of Exler, Lehmann, and Schittkowski [7, 8] is given below. With default tolerances and options, nearly all problems can be solved successfully (95.7 %), i.e., MISQP terminates with IFAIL=0 at a feasible solution, in most cases the same as the known one reported in the literature (77.4 %). 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*. The Fortran codes are compiled by the Intel Visual Fortran Compiler 10.1 under Windows Vista and executed on an Intel Core(TM)2 duo E8500 64 bit processor with 3.16 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 forward 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 equations and that integer variables could be relaxed. The average number of successful

The average number of iterations or gradient evaluations, respectively, is 27 and the average number of equivalent function evaluations including those needed for approximating partial derivatives subject to continuous variables by a one-sided and subject to integer variables by a two-sided difference formula, is 1,271. By changing default tolerances, also the eight problems with non-feasible returns could be successively solved.

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 the 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	329	-0.100097E+05	0.70E-09	0.00E+00	0.0000
2	MITP2	48	0.350000E+01	-0.12E-08	0.16E-08	0.0000
3	QIP1	29	-0.200000E+02	0.00E+00	0.00E+00	0.0000
4	ASAADI11	76	-0.409574E+02	-0.20E-04	0.43E-09	0.0000
5	ASAADI12	112	-0.380000E+02	0.00E+00	0.00E+00	0.0156
6	ASAADI21	403	0.694903E+03	0.11E-05	0.00E+00	0.0000
7	ASAADI22	383	0.700000E+03	0.00E+00	0.00E+00	0.0156
8	ASAADI31	369	0.372191E+02	-0.11E-04	0.48E-09	0.0156
9	ASAADI32	203	0.430000E+02	0.00E+00	0.00E+00	0.0156
10	DIRTY	7324	-0.304679E+09	0.15E-03	0.00E+00	0.5000
11	BRAAK1	2414	0.100025E+01	0.25E-03	0.00E+00	0.0312
12	BRAAK2	425	-0.271810E+01	0.67E-04	0.00E+00	0.0000
13	BRAAK3	508	-0.196559E+07	0.65E-05	0.00E+00	0.0156
14	DEX2	33	-0.569375E+02	0.00E+00	0.00E+00	0.0000
15	FUEL	1068	0.856612E+04	-0.58E-08	0.83E-07	0.0312
16	WP02	35	-0.244444E+01	-0.15E-05	0.00E+00	0.0000
17	NVS01	120	0.124697E+02	-0.95E-07	0.28E-11	0.0000
18	NVS02	455	0.596418E+01	-0.80E-07	0.18E-10	0.0156
19	NVS03	52	0.160000E+02	0.00E+00	0.00E+00	0.0000
20	NVS04	100	0.855200E+02	0.12E+03	0.00E+00	0.0000
21	NVS05	864	0.547093E+01	-0.78E-06	0.97E-06	0.0156
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	210	0.234497E+02	-0.14E-06	0.24E-06	0.0000
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	183	-0.431000E+03	0.00E+00	0.00E+00	0.0000
28	NVS12	425	-0.481200E+03	-0.12E-15	0.00E+00	0.0156
29	NVS13	327	-0.585200E+03	-0.19E-15	0.00E+00	0.0000
30	NVS14	280	-0.403582E+05	-0.12E-06	0.17E-09	0.0156
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	0.0156
34	NVS18	527	-0.778400E+03	0.15E-15	0.00E+00	0.0156
35	NVS19	629	-0.109840E+04	0.00E+00	0.00E+00	0.0156
36	NVS20	1110	0.231518E+03	0.26E-02	0.19E-08	0.0469
37	NVS21	110	-0.509525E+01	0.10E+00	0.42E-07	0.0000
38	NVS22	350	0.605822E+01	0.00E+00	0.48E-08	0.0000
39	NVS23	1413	-0.112520E+04	-0.20E-15	0.00E+00	0.0781
40	NVS24	415	-0.102320E+04	0.97E-02	0.00E+00	0.0156
41	GEAR	248	0.100000E+01	0.45E-07	0.00E+00	0.0000
42	GEAR2	697	0.100000E+01	0.41E-06	0.53E-08	0.9688
43	GEAR2A	3027	0.100000E+01	0.50E-06	0.53E-08	17.3438
44	GEAR3	619	0.100000E+01	0.89E-09	0.11E-08	0.0156
45	GEAR4	1704	0.000000E+00	-0.10E+01	0.19E-03	1.2656
46	M3	1215	0.378000E+02	-0.63E-08	0.59E-08	0.1094
47	M6	11180	0.119325E+03	0.45E+00	0.20E-07	1003.7969
48	M7	9868	0.106757E+03	-0.23E-06	0.16E-07	251.3438
49	FLOUDAS1	24	0.766718E+01	-0.35E-08	0.18E-08	0.0000
50	FLOUDAS2	28	0.107654E+01	0.29E-05	0.11E-08	0.0000
51	FLOUDAS3	184	0.457958E+01	-0.98E-07	0.23E-06	0.0000
52	FLOUDAS4	422	-0.946359E+00	-0.31E-02	0.44E-08	0.0312
53	FLOUDAS40	96	-0.922082E+00	0.23E-01	0.11E-08	0.0000
54	FLOUDAS5	17	0.310000E+02	0.00E+00	0.00E+00	0.0000
55	FLOUDAS6	14	-0.170000E+02	-0.29E-09	0.55E-08	0.0156
56	SPRING	859	0.846246E+00	-0.30E-07	0.56E-08	0.0469

(continued)

no	$name$	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	$time$
57	DU_OPT5	2692	0.211639E+02	0.16E+01	0.00E+00	0.1094
58	DU_OPT	4536	0.713761E+01	0.10E+01	0.00E+00	0.1875
59	ST_E13	12	0.223607E+01	0.12E+00	0.00E+00	0.0000
60	ST_E14	376	0.457958E+01	-0.78E-07	0.23E-06	0.0312
61	ST_E15	29	0.766718E+01	0.60E-09	0.18E-08	0.0000
62	ST_E27	10	0.200000E+01	-0.70E-09	0.23E-09	0.0000
63	ST_E29	253	-0.934453E+00	0.96E-02	0.20E-06	0.0156
64	ST_E31	9870	-0.200000E+01	-0.26E-10	0.39E-06	88.2031
65	ST_E32	555	-0.143041E+01	0.71E-08	0.89E-09	0.2031
66	ST_E35	1796	0.714669E+05	0.10E+00	0.64E-06	0.2500
67	ST_E36	161	-0.246000E+03	-0.76E-10	0.29E-08	0.0000
68	ST_E38	197	0.719773E+04	0.28E-10	0.25E-11	0.0000
69	ST_E40	32	0.441421E+01	-0.85E+00	0.19E+00	0.0000
70	ST_MIQP1	30	0.281000E+03	0.00E+00	0.00E+00	0.0000
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	28	-0.457400E+04	-0.53E-09	0.11E-07	0.0000
74	ST_MIQP5	91	-0.333889E+03	0.33E-07	0.47E-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	0.0156
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	0.0000
80	ST_TEST6	77	0.471000E+03	0.00E+00	0.00E+00	0.0000
81	ST_TEST8	419	-0.295750E+05	0.10E-02	0.00E+00	0.0312
82	ST_TESTGR1	134	-0.127976E+02	0.11E-02	0.00E+00	0.0156
83	ST_TESTGR3	371	-0.205900E+02	0.00E+00	0.00E+00	0.1875
84	ST_TESTPH4	38	-0.805000E+02	0.00E+00	0.00E+00	0.0000
85	TLN2	181	0.530000E+01	0.00E+00	0.00E+00	0.0312
86	TLN4	3366	0.930000E+01	0.12E+00	0.00E+00	7.4062
87	TLN5	1914	0.115000E+02	0.12E+00	0.00E+00	7.5781
88	TLN6	2945	0.178000E+02	0.16E+00	0.00E+00	13.0781
89	TLN7	4835	0.208000E+02	0.67E-01	0.10E+00	41.9531
90	TLN12	5867	0.120000E+02	-0.87E+00	0.10E+01	98.1875
91	TLOSS	883	0.163000E+02	0.00E+00	0.00E+00	1.5781
92	TLTR	310	0.480667E+02	-0.74E-15	0.00E+00	0.5625
93	MEANVARX	616	0.141897E+02	-0.12E-01	0.15E-09	0.0781
94	MINLPHIX	1561	0.316693E+03	-0.93E-07	0.50E-06	20.3750
95	MIP_EX	36	0.350000E+01	0.00E+00	0.00E+00	0.0000
96	MGRID_CYCLES1	93	0.800000E+01	0.00E+00	0.00E+00	0.0000
97	MGRID_CYCLES2	510	0.306000E+03	0.20E-01	0.00E+00	0.0156
98	CROP5	66	0.953099E-01	-0.32E-08	0.00E+00	0.0156
99	CROP20	6702	0.124560E+00	0.12E+00	0.00E+00	6.2031
100	CROP50	6112	0.324243E+00	0.14E-07	0.00E+00	8.6094
101	CROP100	10706	0.851471E+00	0.56E-08	0.00E+00	23.4375
102	SPLITF1	276	-0.160449E+04	0.35E-05	0.16E-08	0.0312
103	SPLITF2	1160	-0.180000E+04	-0.42E-10	0.17E-08	1.0781
104	SPLITF3	1461	-0.250829E+04	0.19E-07	0.12E-08	0.8125
105	SPLITF4	1486	-0.262493E+04	0.63E-03	0.18E-08	0.9375
106	SPLITF5	910	-0.280449E+04	0.20E-05	0.17E-08	0.4688
107	SPLITF6	986	-0.309953E+04	-0.10E-04	0.11E-08	0.4688
108	SPLITF7	1524	-0.250829E+04	0.47E-01	0.24E-08	5.7344
109	SPLITF8	1636	-0.300739E+04	0.11E-01	0.23E-08	5.5625
110	SPLITF9	1038	-0.340449E+04	0.16E-05	0.17E-08	3.5469
111	ELF	4200	0.191666E+00	-0.38E-05	0.51E-07	51.2188
112	SPECTRA2	11803	0.139783E+02	-0.61E-06	0.10E-05	101.7031
113	WINDFAC	934	0.254487E+00	-0.84E-08	0.91E-10	0.0469

(continued)

no	$name$	n_{func}	$f(x^\star, y^\star)$	$e(x^\star, y^\star)$	$r(x^\star, y^\star)$	$time$
114	CSCHED1	1931	-0.302885E+05	0.19E+00	0.11E-06	6.0312
115	ALAN	381	0.292500E+01	0.90E-02	0.10E-07	0.0156
116	PUMP	3195	0.128894E+06	-0.40E-01	0.10E-08	0.6094
117	RAVEM	10086	0.269580E+06	-0.36E-04	0.14E-06	180.0000
118	ORTEZ	4849	-0.102058E+05	-0.71E-01	0.66E-07	5.9688
119	EX1221	30	0.766718E+01	-0.22E-09	0.10E-08	0.0000
120	EX1222	38	0.107654E+01	0.96E-07	0.11E-08	0.0000
121	EX1223	229	0.457958E+01	0.68E-07	0.57E-07	0.0000
122	EX1223A	99	0.457958E+01	0.83E-07	0.11E-07	0.0156
123	EX1223B	161	0.457958E+01	0.11E-07	0.17E-06	0.0000
124	EX1224	447	-0.934453E+00	0.96E-02	0.35E-08	0.0156
125	EX1225	72	0.310000E+02	-0.47E-09	0.71E-08	0.0000
126	EX1226	18	-0.170000E+02	-0.83E-09	0.70E-08	0.0156
127	EX1233	7134	0.165895E+06	0.70E-01	0.76E-08	6.7969
128	EX1243	4080	0.985190E+05	0.18E+00	0.13E-06	7.4062
129	EX1244	3553	0.847569E+05	0.33E-01	0.19E-06	16.6406
130	EX1252	5910	0.133826E+06	0.38E-01	0.25E-08	4.2812
131	EX1252A	841	0.000000E+00	-0.10E+01	0.25E+00	0.1094
132	EX1263	1488	0.216000E+02	0.10E+00	0.15E-07	29.4219
133	EX1263A	814	0.196000E+02	0.00E+00	0.00E+00	1.5312
134	EX1264	1068	0.100000E+02	0.16E+00	0.28E-07	17.2500
135	EX1264A	602	0.860000E+01	0.00E+00	0.00E+00	1.9688
136	EX1265	2882	0.120000E+02	0.17E+00	0.11E-07	131.2031
137	EX1265A	1189	0.103000E+02	0.00E+00	0.00E+00	2.3906
138	EX1266	4347	0.244896E+02	0.50E+00	0.17E-02	253.4688
139	EX1266A	345	0.163000E+02	0.00E+00	0.00E+00	0.0938
140	GBD	25	0.220000E+01	0.00E+00	0.00E+00	0.0000
141	EX3	2409	0.771043E+02	0.13E+00	0.42E-06	0.2656
142	EX4	2132	-0.806415E+01	-0.20E-05	0.54E-06	2.2031
143	FAC1	70	0.387855E+09	0.14E+01	0.80E-07	0.0312
144	FAC2	9543	0.331840E+09	0.67E-05	0.12E-08	7.4062
145	FAC3	3027	0.319823E+08	-0.60E-08	0.21E-08	2.2812
146	GKOCIS	408	-0.192310E+01	0.13E-06	0.13E-08	0.0156
147	KG	130	0.103938E+03	0.48E-01	0.24E-08	0.0000
148	SYNTHES1	239	0.598177E+01	-0.47E-02	0.92E-08	0.0000
149	SYNTHES2	808	0.730353E+02	0.29E-07	0.58E-08	0.0312
150	SYNTHES3	903	0.680097E+02	-0.25E-09	0.12E-08	0.0625
151	PARALLEL	1005	0.217293E+05	0.23E+02	0.71E-06	1.7969
152	SYNHEAT	9449	0.195982E+06	0.26E+00	0.36E-05	2.2969
153	SEP1	1064	-0.532125E+03	-0.43E-01	0.35E-13	0.1719
154	DAKOTA	113	0.136340E+01	-0.25E-06	0.23E-07	0.0000
155	BATCH	8580	0.285506E+06	0.17E-05	0.20E-07	5.8281
156	BATCHDES	460	0.167412E+06	-0.96E-04	0.58E-07	0.0156
157	ENIPLAC	15610	-0.130181E+06	0.13E-01	0.26E-06	270.4531
158	PROB02	143	0.112235E+06	0.00E+00	0.00E+00	0.0000
159	PROB03	15	0.100000E+02	0.00E+00	0.00E+00	0.0000
160	PROB10	16	0.344550E+01	-0.24E-09	0.63E-09	0.0000
161	NOUS1	5461	0.158273E+01	0.10E-01	0.81E-09	2.5000
162	NOUS2	7619	-0.312197E+01	-0.60E+01	0.99E-09	2.2656
163	TLS2	475	0.530000E+01	0.00E+00	0.50E-07	0.9531
164	TLS4	5139	0.190000E+02	0.53E+00	0.85E-07	149.6094
165	TLS5	3305	0.230000E+02	0.62E+00	0.85E-07	187.7500
166	TLS6	1994	0.355000E+02	0.13E+01	0.33E-01	168.6875
167	OAER	160	-0.192310E+01	0.25E-06	0.47E-09	0.0000
168	PROCSEL	364	-0.192310E+01	0.13E-06	0.40E-08	0.0156
169	LICHOU_1	240	-0.246000E+03	-0.12E-09	0.74E-08	0.0000
170	LICHOU_2	44	0.719801E+04	0.99E-02	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	LICHOU_3	64	0.304142E+01	0.70E-05	0.00E+00	0.0000
172	WU_1	66	0.320000E+00	0.87E-15	0.00E+00	0.0156
173	WU_2	99	0.976000E+01	0.55E-15	0.00E+00	0.0000
174	WU_3	130	0.137884E+00	0.12E-08	0.00E+00	0.0312
175	WU_4	130	0.102400E+02	0.00E+00	0.00E+00	0.0469
176	OPTPRLOC	1056	-0.806414E+01	-0.48E-05	0.72E-07	0.5156
177	GASTRANS	54498	0.316339E+03	0.36E+05	0.26E-07	12.4219
178	GASNET	46092	0.101819E+08	0.45E+00	0.21E-06	17.5625
179	TP83	130	-0.306060E+05	0.25E-08	0.18E-09	0.0000
180	TP84	78	-0.571515E+07	-0.54E-08	0.00E+00	0.0000
181	TP85	336	-0.189579E+01	-0.11E-05	0.90E-07	0.0156
182	TP87	113	0.895823E+04	0.31E-08	0.21E-06	0.0000
183	TP93	16	0.152440E+03	0.99E-01	0.18E+00	0.0000
184	FEEDTRAY	34376	-0.132520E+02	0.11E-01	0.19E-07	28.3281
185	FEEDTRAY2	5399	0.100000E+01	0.50E-07	0.36E-06	6.0781
186	DEB10	14599	0.198801E+03	-0.51E-01	0.31E-06	113.0625

References

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

[2] 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

[3] 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

[4] 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

[5] 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

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

[8] 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

[9] 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

[10] 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

[11] 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

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

[13] 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

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

[15] 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

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

[17] 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

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

[19] 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

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