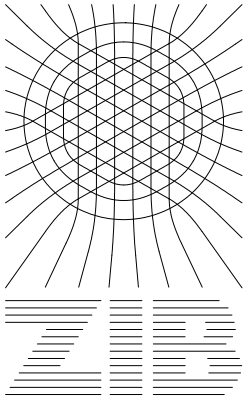

Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

THORSTEN KOCH

The final The final NETLIB-LP results

URL: http://www.zib.de/koch/perplex

ZIB-Report 03-05 (March 2003)

The final NETLIB-LP results

Thorsten Koch*

June 16, 2003

Abstract

The NETLIB has now served for 18 years as a repository of LP problem instances available to test new codes and compare performance. But with standard linear programming solvers there is always some uncertainty about the precise values of the optimal solutions. We have implemented a program called perPlex, which using rational arithmetic computes proofs for the feasibility and optimality of a LP solution. This paper reports finally the *exact* optimal objective values for all NETLIB problems.

1 Introduction

In the 18 years since its introduction in 1985 the NETLIB [6] has served as a repository of linear programming (LP) instances. While in the beginning some of the problems were considered large and difficult, for today's solvers they are small and easy. But down to the present day the exact solution values were not known for all NETLIB problems.

When comparing different solvers, different computer architectures, when changing solver parameters, and even compiler options, the results will vary. Not much, but enough to ask whether the solutions are really optimal or even feasible. As the NETLIB `readme` file states:

> *Note that MINOS control parameters, such as SCALE, PARTIAL PRICE, FEASIBILITY TOLERANCE, OPTIMALITY TOLERANCE, and CRASH OPTION may affect the optimal value that MINOS reports (as may the version of MINOS, the computer, and even the compiler used).*
> ...
> *Bob Bixby reports that the CPLEX solver (running on a Sparc station) finds slightly different optimal values for some of the problems. On a MIPS processor, MINOS version 5.3 (with crash and scaling of December 1989) also finds different optimal values for some of the problems.*

In the following we assume the reader is familiar with linear programming and concepts like the simplex algorithm and the LP basis (cf. [2, 7, 14, 15]).

*koch@zib.de, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustr. 7, D-14195 Berlin, Germany

2 Limited precision

Due do the limited precision floating point arithmetic[1] used by most LP solvers, errors are introduced throughout the computation, for example in the LU and Cholesky factorizations. For a detailed explanation of possible errors see [9, 7].

The solvers[2] deal with this numerical inaccuracies by incorporating tolerances, among others, for feasibility and optimality. Values in the order of 10^{-6} are usually used. Similar, numbers below a given threshold, such as 10^{-14}, are declared to be zero. The inherent arithmetic errors together with the use of tolerances make it possible that the final LP basis is not optimal or even feasible.

In the process of improving the SoPlex [13] LP solver the need arose to find the *exact* solutions for all the Netlib instances to have something to compare against, or at least to test whether a solution obtained is indeed optimal.

3 Unlimited precision

The obvious way to get exact solutions is to implement an algorithm that works with exact arithmetic, for example, using rational numbers of arbitrary size as done in [1, 4]. Unfortunately, such an algorithm is rather slow in the general case [5].

A possible solution is to compute an LP basis with a standard solver and then to check whether it is feasible and optimal. This has been done most notable by C. Kwappik in [12] and carried on in [3]. However, some LP problems from Netlib have not been tackled and for the others only rounded objective function values were published. To the knowledge of the author no code has been made available in public, so there is no possibility to access the exact solution values.

This led to the implementation of perPlex, a simplex based verification tool for feasibility, optimality, and integrality of LP solutions using the GNU multi precision arithmetic library (GMP) [8]. The implementation is along the lines of [15] and features a sparse LU factorization with Markowitz pivoting.

Given an LP file in MPS [10] format and a basis in BAS [11] format, perPlex computes the primal and dual variable values using rational arithmetic and tests if the basis is feasible and optimal. The integrality of integer variables can be also verified. The program is available at `http://www.zib.de/koch/perplex`.

4 Results

CPlex [11] and SoPlex were used to compute LP bases for the Netlib LP instances. As expected the results varied with computer architecture, software settings and program version. At least part of this behavior seems to be a

[1] If a solver uses the same algorithms on all architectures and a full implementation of IEEE floating arithmetic the same numbers should result on all machines. But both conditions are usually deliberately not met for performance reasons.

[2] We usual refer to simplex based solvers, but the same is also true for interior point methods. Especially, since interior point methods are by principle not capable of finding an optimal solution, but use some kind of simplex based crossover procedure at the end to convert a nearly optimal, nearly feasible solution to an (hopefully) optimal feasible one.

property of the instances itself, since there is a group of "usual suspects" that are likely to have varying "optimal" bases reported after optimization.

With default parameter settings, CPLEX 8.0 is able to find the true optimal bases for all instances except for `etamacro`, `d2q06c`, and `scsd6`. With the simplex feasibility and optimality tolerances set to 10^{-9}, activated aggressive scaling, and deactivated preprocessing also those remaining instances are solved to optimality.

The current development version of SOPLEX using 10^{-6} as tolerance finds true optimal bases to all instances besides `d2q05c`, `etamacro`, `nesm`, `dfl001`, and `pilot4`. Changing the representation from 64 to 128 bit floating point arithmetic allows also to solve these cases to optimality.

The columns of Tables 1, 2, and 3 show the name of the instance, the number of non-zero entries (NZ) of the constraint matrix, the storage size of the exact objective value in bits, and the time used for the computation in CPU seconds on a 3 GHz P4 computer.

Due to lack of space in the tables, we only provide the optimal objective value up to 32 digits of precision (the last digit is rounded). Note that for the largest value (`maros-r7`), 47040 bits, which correspond to more than 14,000 decimal digits (nominator and denominator together) are needed.

The complete values given as nominator/denominator pairs together with the optimal basis and the corresponding variable values are available for download at `http://www.zib.de/koch/perplex`. Solutions to other known problems, like the MIPLIB root LPs, can also be found there.

5 Conclusions

Our experiments show that at least for the NETLIB problems the trust we have into standard LP solvers is mostly justified.

The time needed for verification seems to be more dependent on the specific values and structure of the problem itself than purely on the number of non-zero entries in the constraint matrix. Preliminary experiments show that LP relaxations of integer programming problems are likely to be easier in general.

It remains to be seen how other linear programs behave in this regard.

Name	NZ	Size	Time	Optimal objective value	$\times 10^n$
cre-a	14987	96	0.6	$0.23595407060971607914108674680625$	8
cre-b	256095	128	6.1	$0.23129639886832364609512881638847$	8
cre-c	13244	160	0.5	$0.25275116140880216542103232084057$	8
cre-d	242646	96	5.8	$0.24454969764549248 19803581786941$	8
ken-07	8404	96	0.5	-0.679520443381687	9
ken-11	49058	96	4.9	-0.6972382262519971	10
ken-13	97246	96	15.2	-0.10257394789482432	11
ken-18	358171	96	179.8	-0.5221702528739681	11
osa-07	143694	96	2.6	$0.53572251729935104299116808137737$	6
osa-14	314760	96	5.9	$0.11064628447362547969656403391343$	7
osa-30	600138	128	11.6	$0.21421398732097579473449352967425$	7
osa-60	1397793	128	27.6	$0.40440725031574228960285586791611$	7
pds-02	16390	96	0.5	0.2885786201	11
pds-06	62524	96	2.8	0.277610376	11
pds-10	106436	96	6.3	0.26727094976	11
pds-20	230200	96	20.6	0.2382165864	11
pds-40	462128	96	71.7	0.18855198824	11

Table 1: Optimal objective values for the Kennington LP instances.

Name	NZ	Size	Time	Optimal objective value	$\times 10^n$
25fv47	10400	3584	1.2	0.55018458882867447945812325883916	4
80bau3b	21002	1344	0.4	0.98722419240909025757911711738799	6
adlittle	383	224	0.0	0.22549496316238038228101176621492	6
afiro	83	64	0.0	−0.46475314285714285714285714285714	3
agg	2410	480	0.1	−0.35991767286576506712640824319636	8
agg2	4284	416	0.1	−0.20239252355977109024317661926133	8
agg3	4300	416	0.1	0.10312115935089225579061058796215	8
bandm	2494	1280	0.1	−0.15862801845012064052174123768736	3
beaconfd	3375	96	0.0	0.335924858072	5
blend	491	288	0.0	−0.30812149845828220173774356124984	2
bnl1	5121	640	0.1	0.19776295615228892439564398331821	4
bnl2	13999	704	0.2	0.18112365403585451704484136976910	4
boeing1	3485	448	0.1	−0.33521356750712662184296973146820	3
boeing2	1196	192	0.0	−0.31501872801520287870462195913263	3
bore3d	1429	576	0.0	0.13730803942084927215581987251301	4
brandy	2148	576	0.1	0.15185098964881283835426751550618	4
capri	1767	1024	0.0	0.26900129137681610087717280693754	4
cycle	20720	64	0.2	−0.52263930248941017172447233836217	1
czprob	10669	288	0.1	0.21851966988565774858951155947191	7
d2q06c	32417	13152	18.1	0.12278421081418945895739128812392	6
d6cube	37704	64	0.5	0.31549166666666666666666666666667	3
degen2	3978	64	0.0	−0.1435178	4
degen3	24646	64	0.2	−0.987294	3
dfl001	35632	224	1.1	0.11266396046671392202377652175477	8
e226	2578	800	0.1	−0.18751929066370549102605687681285	2
etamacro	2409	608	0.0	−0.75571523337491333507925836677730	3
fffff800	6227	736	0.1	0.55567956481749637653286437896900	6
finnis	2310	480	0.0	0.17279106559561159432297900375543	6
fit1d	13404	96	0.1	−0.91463780924209269467749025024617	4
fit1p	9868	96	0.1	0.91463780924209269467749025024617	4
fit2d	129018	128	1.4	−0.68464293293832069575943518435837	5
fit2p	50284	128	0.8	0.68464293293832069575943518435837	5
forplan	4563	384	0.1	−0.66421896127220457481235119701692	3
ganges	6912	224	0.1	−0.10958573612927811623444890424357	6
gfrd-pnc	2377	96	0.1	0.69022359995488088295415596232193	7
greenbea	30877	3808	1.4	−0.72555248129845987457557870574845	8
greenbeb	30877	4576	2.5	−0.43022602612065867539213672544432	7
grow15	5620	2272	2.8	−0.10687094129357533671604040930313	9
grow22	8252	800	4.2	−0.16083433648256296718456039982613	9
grow7	2612	1536	0.9	−0.47787811814711502616766956242865	8
israel	2269	288	0.1	−0.89664482186304572966200464196045	6
kb2	286	320	0.0	−0.17499001299062057129526866493726	4
lotfi	1078	128	0.0	−0.2526470606188	2
maros-r7	144848	47040	446.7	0.14971851664796437907337543903552	7
maros	9614	1344	0.2	−0.58063743701125895401208534974734	5
modszk1	3168	2048	0.1	0.32061972906431580494333823530763	3
nesm	13288	1696	0.3	0.14076036487562728337980641375835	8

Table 2: Optimal objective values for NETLIB LP instances, part 1

Name	NZ	Size	Time	Optimal objective value	$\times 10^n$
perold	6018	8064	4.9	$-0.93807552782351606734833270386511$	4
pilot-ja	14698	9344	8.5	$-0.61131364655813432748848620538024$	4
pilot-we	9126	12352	6.0	$-0.27201075328449639629439185412556$	7
pilot	43167	15200	125.8	$-0.55748972928406818073034256636894$	3
pilot4	5141	6720	4.6	$-0.25811392588838886745830997266797$	4
pilot87	73152	34464	2972.7	$0.30171034733311052772160020 1995$	3
pilotnov	13057	256	1.5	$-0.44972761882188711430996211783943$	4
recipe	663	64	0.0	-0.266616	3
sc105	280	96	0.0	$-0.52202061211707248062628010857689$	2
sc205	551	96	0.0	$-0.52202061211707248062628010857689$	2
sc50a	130	64	0.0	$-0.64575077058564509026860413914575$	2
sc50b	118	64	0.0	-0.7	2
scagr25	1554	224	0.0	$-0.14753433060768523167790925075974$	8
scagr7	420	96	0.0	-0.2331389824330984	7
scfxm1	2589	608	0.0	$0.18416759028348943683579089143655$	5
scfxm2	5183	896	0.1	$0.36660261564998812956939504848248$	5
scfxm3	7777	896	0.1	$0.54901254549751454623888724925519$	5
scorpion	1426	256	0.0	$0.18781248227381066296479411763586$	4
scrs8	3182	608	0.0	$0.90429695380079143579923107948844$	3
scsd1	2388	128	0.0	$0.86666666743333647292533502995263$	1
scsd6	4316	192	0.1	$0.50500000077144345289985489081569$	2
scsd8	8584	96	0.1	$0.90499999992546440094445846417654$	3
sctap1	1692	64	0.0	0.141225	4
sctap2	6714	64	0.1	$0.17248071428571428571428571428571$	4
sctap3	8874	64	0.1	0.1424	4
seba	4352	64	0.1	0.157116	5
share1b	1151	608	0.0	$-0.76589318579185681127972 74346007$	5
share2b	694	224	0.0	$-0.41573224074141948654519910873841$	3
shell	3556	64	0.0	0.1208825346	10
ship04l	6332	128	0.1	$0.17933245379703557625562556255626$	7
ship04s	4352	160	0.1	$0.17987147004453921719954082726049$	7
ship08l	12802	160	0.2	$0.19090552113891315179803819998425$	7
ship08s	7114	160	0.1	$0.19200982105346197101726954 74693$	7
ship12l	16170	160	0.2	$0.14701879193292648227022175543886$	7
ship12s	8178	160	0.1	$0.14892361344061291565686421605401$	7
sierra	7302	96	0.1	0.1539436218363193	8
stair	3856	4608	4.2	$-0.25126695119296330352803637106304$	3
standata	3031	64	0.0	0.12576995	4
standmps	3679	64	0.0	0.14060175	4
stocfor1	447	448	0.0	$-0.41131976219436406065682760731514$	5
stocfor2	8343	1184	0.2	$-0.39024408537882029604587908772433$	5
stocfor3	64875	1728	1.9	$-0.39976783943649587403509204700686$	5
truss	27836	160	0.4	$0.45881584718561673163624258476816$	6
tuff	4520	256	0.1	$0.29214776509361284488226309994302$	0
vtp-base	908	192	0.0	$0.12983146246136136573959846 84889$	6
wood1p	70215	768	2.3	$0.14429024115734092400010936668043$	1
woodw	37474	192	0.4	$0.13044763330842292690055520855 66$	1

Table 3: Optimal objective values for NETLIB LP instances, part 2

References

[1] David Avis. A revised implementation of the reverse search vertex enumeration algorithm. In Gil Kalai and Günter M. Ziegler, editors, *Polytopes – Combinatorics and Computation*, volume 29 of *DMV Seminar*, pages 177–198. Birkhäuser, Basel, 2000. Software available at `http://cgm.cs.mcgill.ca/~avis/lrs.html`.

[2] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, 1983.

[3] M. Dhiflaoui, S. Funke, C. Kwappik, K. Mehlhorn, M. Seel, E. Schömer, R. Schulte, and D. Weber. Certifying and repairing solutions to large lps - how good are lp-solvers? In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003. Paper available at `http://www.mpi-sb.mpg.de/~mehlhorn/ftp/LPExactShort.ps`.

[4] K. Fukuda. cddlib reference manual, cddlib version 092a, 2001. Program available at `http://www.cs.mcgill.ca/~fukuda/software/cdd_home/cdd.html`.

[5] B. Gärtner. Exact arithmetic at low cost – a case study in linear programming. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1998.

[6] M. Gay. Electronic mail distribution of linear programming test problems. *Mathematical Programming Society COAL Bulletin no. 13*, pages 10–12, 1985. Data available at `http://www.netlib.org/netlib/lp`.

[7] P.E. Gill, W. Murray, and M.H. Wright. *Numerical Linear Algebra and Optimization*, volume 1. Addison-Wesley, 1991.

[8] GNU multiple precision arithmetic library (GMP), version 4.1.2., 2003. Code and documentation available at `http://www.swox.com/gmp`.

[9] D. Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1):5–48, 1991.

[10] IBM optimization library guide and reference, 1997. For an online reference see `http://www6.software.ibm.com/sos/features/featur11.htm`.

[11] ILOG CPLEX Division, 889 Alder Avenue, Suite 200, Incline Village, NV 89451, USA. *ILOG CPLEX 8.0 Reference Manual*, 2002. Information available at `http://www.cplex.com`.

[12] C. Kwappik. Exact linear programming. Master's thesis, Universität des Saarlandes, Fachbereich 14, 1998.

[13] SoPlex – The sequential object-oriented simplex class library, 2003. See `http://www.zib.de/Optimization/Software/Soplex`.

[14] R.J. Vanderbei. *Linear Programming : Foundations and Extensions*. Kluwer Academic, 2nd edition, 2001.

[15] R. Wunderling. Paralleler und objektorientierter Simplex. Technical Report TR 96-09, Konrad-Zuse-Zentrum Berlin, 1996. Electronically available at `http://www.zib.de/PaperWeb/abstracts/TR-96-09`.