

Solution of multiscale linear programs using Quad precision

Ding Ma¹ and Michael A. Saunders^{1,*}¹Department of Management Science and Engineering, Stanford University, Stanford, CA, USAFor submission to *Bioinformatics*

Technical Report SOL 2014-1, April 2014

ABSTRACT

Motivation: Systems biologists are developing increasingly large models of metabolism and integrated models of metabolism and macromolecular expression. These Metabolic Expression (ME) models lead to sequences of multiscale linear programming problems for which small solution values of order 10^{-6} to 10^{-10} are meaningful. Standard LP solvers do not give sufficiently accurate solutions, and exact simplex solvers are extremely slow. We investigate whether double-precision and quadruple-precision simplex solvers can together achieve reliability at acceptable cost.

Results: A double-precision LP solver often provides a reasonably good starting point for a Quad simplex solver. On a range of multiscale examples we find that 34-digit Quad floating-point achieves exceptionally small primal and dual infeasibilities (of order 10^{-30}) when no more than 10^{-15} is requested. On a significant ME model we also observe robustness in almost all (even small) solution values following relative perturbations of order 10^{-6} to non-integer data values.

Availability and Implementation: Double and Quad Fortran 77 implementations of the linear and nonlinear optimization solver MINOS are available upon request.

Contact: Michael Saunders: saunders@stanford.edu

1 INTRODUCTION

We consider the solution of large, multiscale linear programs (LPs) of the form

$$\min_x c^T x \text{ s.t. } \ell \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u, \quad (1.1)$$

where A is a sparse matrix whose entries, like the variables in x , may be of widely varying magnitude. Such problems arise in systems biology in the modeling of biochemical reaction networks like Metabolic Expression (ME) models. Reliable solution methods are of such importance to systems biologists that *exact simplex solvers* have been employed (Lerman *et al.*, 2012), even though the solution time is measured in weeks for genome-scale models. Exact solvers are based on rational arithmetic, and there has been considerable work on their implementation and application to important problems (Koch, 2004; Applegate *et al.*, 2007; Cook *et al.*, 2007; QSopt_ex, 2008). The use of quadruple precision floating-point has also been mentioned in passing (Koch, 2004; Applegate *et al.*, 2007). Our aim is to explore further the use of Quad simplex solvers, especially for large multiscale problems that would be too costly with exact solvers.

Kahan (2011) notes that “*carrying somewhat more precision in the arithmetic than twice the precision carried in the data and available for the result will vastly reduce embarrassment due to roundoff-induced anomalies.*” He further notes that Quad precision is unlikely to be adopted widely in the foreseeable future because of the cost in CPU time and memory (especially cache) relative to Double, but in terms of finding ways to avoid unexpected total loss of accuracy, “*default evaluation in Quad is the humane option.*”

We apply the “humane” approach to difficult LP problems by invoking a normal Double simplex solver first, saving the final solution, and warm-starting a Quad solver from that point. For a sequence of related problems, warm-starting directly in Quad may be appropriate for each one.

2 SYSTEM AND METHODS

The Constraint Based Reconstruction and Analysis (COBRA) approach has been successfully applied to biological processes such as metabolism and macromolecular synthesis, which are inherently multiscale. In the COBRA approach, a biochemical network is represented by a stoichiometric matrix S with m rows corresponding to metabolites and n columns representing reactions. Mathematically, S is part of the ordinary differential equation that governs the time-evolution of concentrations in the network:

$$\frac{d}{dt}x(t) = Sv(t), \quad (2.1)$$

where $x(t) \in \mathbf{R}^m$ is a vector of time-dependent concentrations and $v(t) \in \mathbf{R}^n$ is a vector of reaction fluxes. With the objective of maximizing the growth rate at the steady state, the following LP is constructed:

$$\max_v c^T v \quad (2.2a)$$

$$\text{s.t. } Sv = 0, \quad (2.2b)$$

$$\ell \leq v \leq u, \quad (2.2c)$$

where growth is defined as the biosynthetic requirements of experimentally determined biomass composition, and biomass generation is a set of reaction fluxes linked in the appropriate ratios (Palsson, 2006).

Flux Balance Analysis (FBA) FBA is a mathematical approach widely used for studying biochemical reaction networks (Palsson, 2006; Orth *et al.*, 2010). The biochemical networks reconstructed in FBA with a linear objective function are essentially LPs as in (2.2), where the fluxes in vector v may have widely varying values in the range 0–100 say, with small values such as $v_j = 10^{-10}$ being meaningful. With the increasingly large, sparse, and multiscale nature of biochemical networks, a Quad solver has become more necessary, practical, and even efficient.

ME models (FBA with coupling constraints) FBA has been used by Thiele *et al.* (2012) for the first integrated stoichiometric multiscale model of metabolism and macromolecular synthesis for *Escherichia*

*To whom correspondence should be addressed.

coli K12 MG1655. The model modifies (2.2) by adding constraints that couple enzyme synthesis and catalysis reactions to (2.2b). Coupling constraints of the form

$$c_{\min} \leq \frac{v_i}{v_j} \leq c_{\max} \quad (2.3)$$

become linear constraints

$$c_{\min} v_j \leq v_i, \quad v_i \leq c_{\max} v_j \quad (2.4)$$

for various pairs of fluxes v_i, v_j . They are linear approximations of nonlinear constraints and make S in (2.2b) even less well-scaled because of large variations in reaction rates. Quad precision is evidently more appealing in this case.

ME models with nonlinear constraints As coupling constraints are often functions of the organism's growth rate μ , O'Brien *et al.* (2013) consider growth-rate optimization nonlinearly with the single μ as the objective in (2.2a) instead of via a linear biomass objective function. Nonlinear constraints of the form

$$\frac{v_i}{v_j} \leq \mu \quad (2.5)$$

represented as

$$v_i \leq \mu v_j \quad (2.6)$$

are added to (2.2b), where v_i, v_j, μ are all variables. Constraints (2.6) are linear if μ is fixed at a specific value μ_k within an interval $[\mu_{\min}, \mu_{\max}]$. O'Brien *et al.* (2013) employ a binary search to find the largest $\mu_k \equiv \mu^*$ that keeps the associated linear program feasible. Thus, the procedure requires reliable solution of a sequence of related LPs.

Flux Variability Analysis (FVA) After FBA (2.2) returns an optimal objective value $c^T v^* = Z_0$ (2.2a), FVA examines how far a particular flux v_j can vary within the feasible region without changing the optimal objective significantly:

$$\begin{aligned} \max_v \text{ or } \min_v \quad & v_j \\ \text{s.t.} \quad & Sv = 0, \\ & c^T v \geq \gamma Z_0, \\ & l \leq v \leq u, \end{aligned} \quad (2.7)$$

where $0 < \gamma < 1$. Potentially $2n$ LPs (2.7) are solved if all reactions are of interest, with warm starts being used when j increases to $j + 1$ (Gudmundsson and Thiele, 2010).

Other challenging LPs A set of difficult LP problems has been collected by Mészáros (2004), who names them *problematic* and notes that “*modeling mistakes made these problems “crazy,” but they are excellent examples to test numerical robustness of a solver.*” Our procedure for handling these *problematic* problems seems appropriate for the systems biology models as well.

3 ALGORITHM AND IMPLEMENTATION

The experiments reported below were conducted with Double and Quad Fortran 77 implementations of MINOS (Murtagh and Saunders, 1978, 1982). The primal simplex solver in MINOS includes geometric-mean scaling, the EXPAND anti-degeneracy procedure (Gill *et al.*, 1989), partial pricing, but no steepest-edge pricing (which would generally reduce total iterations and time). Basis factorizations and updates are handled by LUSOL (2013).

For Double MINOS, all floating-point variables are declared `double precision` (approximately 15 digits). For Quad MINOS, they are declared `real (16)` (approximately 34 digits). The LP data A, b, c, S, ℓ, u are stored in Quad even though they are not known to that precision. This allows floating-point operations such as Ax and $A^T y$ to be carried out directly on the elements of A and the Quad vectors x, y . If A were stored in Double, such products would require individual entries A_{ij} to be converted from Double to Quad at runtime.

To achieve reliability on the Mészáros problems, we developed the following 3-step procedure for solving challenging examples of problem (1.1), which includes all of problems (2.2)–(2.7):

1. (Cold start in Double with scaling) Apply Double MINOS with somewhat strict settings. Save a final basis file.
2. (Warm start in Quad with scaling) Start Quad MINOS from the saved file with tighter Feasibility and Optimality tolerances. Save a final basis file.
3. (Warm start in Quad without scaling) Start Quad MINOS from the second saved file with no scaling but tighter Factor tolerance.

Cold starts use a Crash procedure to find a triangular initial basis. Steps 1 and 2 are “obvious” and should usually be sufficient. Step 3 provides some insurance in case Step 2 was interrupted, and ensures that the Feasibility and Optimality tolerances are imposed upon the original problem (not the scaled problem).

The default options for the primal simplex method in Double MINOS include the following:

Scale option	2
Feasibility tol	1e-6
Optimality tol	1e-6
LU Factor tol	100.0
LU Update tol	10.0
Expand frequency	10000

The options chosen for Steps 1–3 above are summarized here:

	Step1	Step2	Step3
Scale option	2	2	0
Feasibility tol	1e-7	1e-15	1e-15
Optimality tol	1e-7	1e-15	1e-15
LU Factor tol	10.0	10.0	5.0
LU Update tol	10.0	10.0	5.0

with Expand frequency 100000 in all cases.

The Feasibility tolerance δ_1 is applied in absolute form. Thus, a (possibly scaled) solution v is considered feasible for problem (2.2) if $\ell - \delta_1 \leq v \leq u + \delta_1$. The Optimality tolerance δ_2 is applied in a relative way. If the current basic solution is of the form $Sv \equiv Bv_B + Nv_N = b$ and if $B^T y = c_B$ for the nonsingular basis matrix B , the current v is considered optimal if $z \equiv c - A^T y$ has the correct sign to within the tolerance $(1 + \|y\|_\infty)\delta_2$.

For conventional Double solvers it is reasonable to set δ_1 and δ_2 in the range 10^{-6} to 10^{-8} . For our Quad solver we set $\delta_1 = \delta_2 = 10^{-15}$ to capture fluxes v_j of order 10^{-10} . Of significant interest is our empirical finding that Quad MINOS almost invariably achieves *far more accurate solutions than requested*, in the sense that the maximum primal and dual infeasibilities are almost always far smaller than 10^{-15} , as shown in Table 2.

4 NUMERICAL RESULTS

We report results from Double and Quad versions of the primal simplex solver in MINOS. All runs were on a 2.93 GHz Apple iMac with quad-core Intel i7, using the gfortran compiler with `-O` flag and either 64-bit hardware floating-point throughout, or 128-bit software floating-point throughout (gfortran's `quadmath` library).

We applied our 3-step procedure to three sets of LP problems. Table 1 lists the problem dimensions and solution norms. Table 2 summarizes the results of Steps 1–3 for each problem.

All problems were input from files in the classical MPS format of commercial mathematical programming systems (MPS, 1960) with 12-character fields for all data values. This was a fortuitous limitation for the ME models, as we discuss below.

Table 1. Three Netlib problems, the Mészáros *problematic* LPs, and three ME biochemical network models. Dimensions of $m \times n$ constraint matrices A ($= S$ for the ME models), and size of the largest primal and dual variables.

model	m	n	$\text{nnz}(A)$	$\max A_{ij} $	$\ x^*\ _\infty$	$\ y^*\ _\infty$
pilot4	411	1000	5145	2.8e+04	9.6e+04	2.7e+02
pilot	1442	3652	43220	1.5e+02	4.1e+03	2.0e+02
pilot87	2031	4883	73804	1.0e+03	2.4e+04	1.1e+01
de063155	853	1488	5405	8.3e+11	3.1e+13	6.2e+04
de063157	937	1488	5551	2.3e+18	2.3e+17	6.2e+04
de080285	937	1488	5471	9.7e+02	1.1e+02	2.6e+01
gen1	770	2560	64621	1.0e+00	3.0e+00	1.0e+00
gen2	1122	3264	84095	1.0e+00	3.3e+00	1.0e+00
gen4	1538	4297	110174	1.0e+00	3.0e+00	1.0e+00
l30	2702	15380	64790	1.8e+00	1.0e+09	4.2e+00
iprob	3002	3001	12000	9.9e+03	3.1e+02	1.1e+00
TMA_ME	18210	17535	336302	2.1e+04	5.9e+00	1.1e+00
GlcAerWT	68300	76664	926357	8.0e+05	6.3e+07	2.4e+07
GlcAlift	69529	77893	928815	2.6e+05	6.3e+07	2.4e+07

The pilot problems

These are economic models developed by Prof G. B. Dantzig’s group in the Systems Optimization Laboratory at Stanford University during the 1980s. They are available from Netlib (1988). For the middle example (pilot), MINOS required about 24 hours on a DEC MicroVAX II during 1987, and did not perform reliably until the EXPAND anti-degeneracy procedure was developed.

Line 1 for pilot in Table 2 shows that Double MINOS with cold start and scaling required 16060 primal simplex iterations and 5.7 CPU seconds. The final unscaled primal solution x satisfied the bounds ℓ and u in (1.1) to within $O(10^{-6})$, and the dual solution y satisfied the optimality conditions to within $O(10^{-3})$.

Line 2 for pilot shows that Quad MINOS starting from that point with scaling needed only 29 iterations and 0.7 seconds to obtain a very accurate solution (where $\log_{10}(\text{Pinf}) = -99$ means that the maximum primal infeasibility was 0.0).

Line 3 for pilot shows that in the “insurance” step, Quad MINOS warm-starting again but with no scaling gave a full quad-precision solution at almost no cost: maximum infeasibilities 0.0 and $O(10^{-32})$. The final Double and Quad objective values differ in the 4th significant digit, as suggested by removal of Step 1’s $O(10^{-3})$ primal infeasibility.

Results for the bigger problem pilot87 are analogous.

The Mészáros problems

The *problematic* LPs were provided as MPS files by Ed Klotz (Klotz, 2014). The first two problems have unusually large entries in the constraint matrix A . The Step 1 Double MINOS solution has at best 1 digit of precision in the objective value for de063155, and is quite erroneous for de063157. Nevertheless, the Step 2 and 3 Quad solutions are seen to be highly accurate when the solution norms are taken into account.

The gen* problems come from image reconstruction, with no large entries in A , x , y , but highly degenerate primal solutions x . (In both Steps 1 and 2 for gen1, 60% of the iterations made no improvement to the objective, and the final solution has 30% of the basic variables on their lower bound.) For gen1, warm-starting Quad MINOS from the Step 1 basis gave an almost feasible initial solution (266 basic variables outside their bounds by more than 10^{-15} with a sum of infeasibilities of only $O(10^{-8})$), yet nearly 250,000 iterations were needed in Step 2 to reach optimality. These examples show that steepest-edge pricing would still be valuable in Quad.

Problem l30 behaved similarly (80% degenerate iterations in Steps 1–2). The tiny objective value is essentially zero, so we can’t expect the Step 2 and 3 objectives to agree in their leading digits.

Table 2. Iterations and runtimes in seconds for Step 1 (Double MINOS) and Steps 2–3 (Quad MINOS); final objective value; \log_{10} of maximum primal and dual infeasibilities. Problem iprob is infeasible.

model	Itns	Times	Final objective	Pinf	Dinf
pilot4	1571	0.1	-2.5811392602e+03	-05	-13
	6	0.0	-2.5811392589e+03	-39	-31
	0	0.0	-2.5811392589e+03	-99	-30
pilot	16060	5.7	-5.5739887685e+02	-06	-03
	29	0.7	-5.5748972928e+02	-99	-27
	0	0.2	-5.5748972928e+02	-99	-32
pilot87	19340	15.1	3.0171038489e+02	-09	-06
	32	2.2	3.0171034733e+02	-99	-33
	0	1.2	3.0171034733e+02	-99	-33
de063155	921	0.0	1.8968704286e+10	-13	+03
	78	0.1	9.8830944565e+09	-99	-17
	0	0.0	9.8830944565e+09	-99	-24
de063157	488	0.0	1.4561118445e+11	+20	+18
	476	0.5	2.1528501109e+07	-27	-12
	0	0.0	2.1528501109e+07	-99	-12
de080285	418	0.0	1.4495817688e+01	-09	-02
	132	0.1	1.3924732864e+01	-35	-32
	0	0.0	1.3924732864e+01	-99	-32
gen1	369502	205.3	-1.6903658594e-08	-06	-12
	246428	9331.3	1.2935699163e-06	-12	-31
	2394	81.6	1.2953925804e-06	-45	-30
gen2	44073	60.0	3.2927907828e+00	-04	-11
	1599	359.9	3.2927907840e+00	-99	-29
	0	10.4	3.2927907840e+00	-99	-32
gen4	45369	212.4	1.5793970394e-07	-06	-10
	53849	14812.5	2.8932268196e-06	-12	-30
	37	10.4	2.8933064888e-06	-54	-30
l30	1229326	876.7	9.5266141574e-01	-10	-09
	275287	7507.1	-7.5190273434e-26	-25	-32
	0	0.2	-4.2586876849e-24	-24	-33
iprob	1087	0.2	2.6891551285e+03	+02	-11
	0	0.0	2.6891551285e+03	+02	-31
	0	0.0	2.6891551285e+03	+02	-28
TMA_ME	12225	37.1	8.0051076669e-07	-06	-05
	685	61.5	8.7036315385e-07	-24	-30
	0	6.7	8.7036315385e-07	-99	-31
GlcAerWT	62856	9707.3	-2.4489880182e+04	+04	-05
	5580	3995.6	-7.0382449681e+05	-07	-26
	4	60.1	-7.0382449681e+05	-19	-21
GlcAlift	134693	14552.8	-5.1613878666e+05	-03	-01
	3258	1067.1	-7.0434008750e+05	-09	-26
	2	48.1	-7.0434008750e+05	-20	-22

Problem iprob is an artificial one that was intended to be feasible with a very ill-conditioned optimal basis, but the MPS file provided to us contained low-precision data (many entries like 0.604 or 0.0422). Our Double and Quad runs agree that the problem is infeasible. This is an example of Quad removing some doubt that was inevitable with just Double.

The systems biology ME problems

Like the gen* problems, the ME models showed 40–60% degenerate iterations in Step 1, but fortunately not so many total iterations in Step 2. This is important for FVA and for ME with nonlinear constraints, where there are many warm starts.

Problem TMA_ME developed by Lerman (2012) has some large matrix entries $|S_{ij}|$ and many small solution values v_j that are meaningful to

systems biologists. The ME part of the model also contains small matrix entries. In Step 1, almost all iterations went on finding a feasible solution, and the objective then had the correct order of magnitude.

This was the first ME model that we used for Quad experiments (in April 2012). The data S , c , l , u in (2.2) came as a Matlab structure with $c_j = 0$, $l_j = 0$, $u_j = 1000$ for most j , except $c_{17533} = 1$ (meaning maximize flux v_{17533}), four variables had smaller positive upper bounds, the last variable had moderate positive bounds, and 64 variables were fixed at zero. We output the data to a plain text file in 19.12e format. Most entries of S are integers (represented exactly), but about 5000 S_{ij} values are of the form $8.037943687315e-01$ or $3.488862338191e-06$ with the last significant digits nonzero. The text data was read into Double and Quad versions of a prototype Fortran 90 implementation of SQOPT (Gill *et al.*, 2005).

For the present paper, we used the same Matlab data to generate an MPS file for input into MINOS. Since this is limited to 12.5e format, the above two values were rounded to $8.03794e-01$ and $3.48886e-06$ and in total about 5000 S_{ij} values had $O(10^{-6})$ relative perturbations of this kind. We have been concerned that such data perturbations could alter the FBA solution greatly because the final basis matrices could have condition number as large as 10^6 or even 10^{12} (as estimated by LUSOL). In comparing Quad SQOPT on Matlab data with Quad MINOS on MPS data, we fortunately observe that the final objective values for TMA_ME agree to 5 digits and match the results from SoPlex (1996) and QSOpt.ex (2008), as reported by Lerman (2012):

SoPlex 80bit	8.703671403e-07	Matlab data
QSOpt.ex	8.703646169e-07	Matlab data
Quad SQOPT	8.703646169e-07	Matlab data
Quad MINOS	8.703631539e-07	MPS file

More importantly, for the most part *even small solution values* are perturbed in only the 5th or 6th significant digit. Let \hat{v} and \check{v} be the solutions obtained by the two Quad solvers. Some example values follow:

j	107	201	302
Quad SQOPT \hat{v}_j	2.336815e-06	8.703646e-07	1.454536e-11
Quad MINOS \check{v}_j	2.336823e-06	8.703632e-07	1.454540e-11

Among all j for which $\max(\hat{v}_j, \check{v}_j) > 10^{-15}$ (the feasibility tolerance), the largest relative difference $|\hat{v}_j - \check{v}_j| / \max(\hat{v}_j, \check{v}_j)$ was less than 10^{-5} for all but 31 variables. For 22 of these pairs, either \hat{v}_j or \check{v}_j was primal or dual degenerate (meaning one of them was zero and there are alternative solutions with the same objective value). The remaining 9 variables had these values:

j	\hat{v}_j	\check{v}_j	Relative diff
16383	6.0731e-07	2.0374e-06	7.0e-01
16459	1.7090e-06	2.1778e-06	2.2e-01
16483	2.4675e-06	5.9936e-07	7.6e-01
16730	1.4432e-06	7.8685e-07	4.6e-01
17461	1.7090e-06	2.1778e-06	2.2e-01
17462	2.4675e-06	5.9936e-07	7.6e-01
17478	6.0731e-07	2.0374e-06	7.0e-01
17507	1.4432e-06	7.8685e-07	4.6e-01
17517	8.7036e-07	2.9740e-06	7.1e-01

We see that the \hat{v}_j, \check{v}_j values are quite small (the same magnitude as the data perturbation), and for each of the 9 pairs there is about 1 digit of agreement. In general we could expect thousands of small solution pairs to differ this much, yet for almost *all* 17535 pairs, there are at least 5 digits of agreement.

We trust that these observations about two forms of problem TMA_ME are welcome empirical evidence of the robustness of this particular multiscale model. Quad solvers can help evaluate the robustness of future (increasingly large) models of metabolic networks by enabling comparison of perturbed high-accuracy solutions in the same way.

Problem GlcAerWT is an ME model from the detailed study by Thiele *et al.* (2012). Difficulties with solving TMA_ME and GlcAerWT led to the lifting technique of Sun *et al.* (2013) (and to problem GlcAlift).

After 33000 iterations on GlcAerWT, Double MINOS began to report singularities every 50–100 iterations following updates to the basis LU factors. After another 30000 iterations, MINOS terminated Step 1 with maximum infeasibility $O(10^4)$. Step 2 required significant work to achieve a reasonably accurate solution. Step 3 quickly confirmed the final objective value with high accuracy considering the $O(10^7)$ primal and dual solutions norms.

Problem GlcAlift is a reformulated version of GlcAerWT in which some large matrix entries c_{\max} in (2.4) have been reduced via the lifting technique (Sun *et al.*, 2013). In Step 1, Double MINOS again reported frequent singularities and required twice as many iterations as GlcAerWT, but a near-optimal solution was found (allowing for the primal and dual solution norms of $O(10^7)$), and Steps 2–3 were more efficient and accurate. The objective function for both GlcA models is to maximize variable v_{60069} . The fact that the Step 1 objective values have no correct digits illustrates the challenge these models present and emphasizes the benefits that Quad precision offers. Theoretically the optimal objectives for GlcAerWt and GlcAlift should agree. We assume that the limited data precision in the MPS files is responsible for only 3-digit agreement. Fortunately the Tomlab interface used by Thiele *et al.* (2012) allows full double-precision data (TOMLAB, 2014). We can do the same for MINOS, as we did for SQOPT.

5 DISCUSSION

While today’s advanced LP solvers such as CPLEX, Mosek, and Gurobi (CPLEX, 2014; MOSEK, 2014; Gurobi, 2014) are effective on a wide range of large and challenging linear optimization models, the study by Thiele *et al.* (2012) emphasized the need for improved reliability in solving FBA and ME models in systems biology. Fortunately, reformulation and careful use of the commercial solvers permitted successful analysis in (Thiele *et al.*, 2012) of the GlcAerWT and GlcAlift models discussed here, and we encourage this approach for Step 1 of our proposed 3-step procedure (section 3). The bulk of the work in solving multiscale LP problems can still be performed there by conventional Double solvers (possibly including Barrier solvers with simplex “cross-over” to provide a basis).

Our aim has been to demonstrate that the Step 2 and 3 warm starts with Quad solvers will be acceptably efficient, and that the accuracy achieved exceeds requirements by a very safe margin. The “humane” approach of Kahan (2011)—use of Quad LP solvers—is certainly more efficient than applying exact simplex solvers, even though the latter have proved their value in several applications (Koch, 2004; Applegate *et al.*, 2007; Cook *et al.*, 2007; Lerman *et al.*, 2012).

An intriguing question remains concerning the bold figures in Table 2. The primal and dual solutions obtained with Quad precision are *substantially more accurate than the 10^{-15} requested*. The same has been true for all of the classic set of Netlib problems (Netlib, 1988) that we have run. Kahan (2011) explains that “perturbations get amplified by singularities near the data.” He describes a “pejorative surface” of data points where singularity exists, and expects loss of accuracy as data approaches the surface. The volume surrounding the pejorative surface is the danger zone, but: “*Arithmetic precision is usually extravagant enough if it is somewhat more than twice as [great] as the data’s and the desired result’s. Often that shrunken volume contains no data.*” We can surmise that Kahan has anticipated our observed situation, wherein LP problems defined with double-precision data appear unlikely to be too ill-conditioned for a Quad solver.

We believe that quadruple-precision solutions are now practical for multiscale LP applications such as FBA and FVA, and that they justify increased confidence as systems biologists build ever-larger models to explore new hypotheses about metabolism and macromolecular synthesis.

FUNDING

This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health [award U01GM102098]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

ACKNOWLEDGEMENTS

We thank Ronan Fleming, Philip Gill, Ed Klotz, Joshua Lerman, Edward O'Brien, Yuekai Sun, Ines Thiele, and Elizabeth Wong for much help during the course of this work. Joshua Lerman at UC San Diego provided the model named TMA_ME here (originally model_final_build_unscaled.mat) and advised us of the final objective values obtained by SoPlex and QSOpt-exact. Ines Thiele at the University of Luxembourg provided the GlcAerWT model and insight into the ME coupling constraints. Yuekai Sun at Stanford University created the reformulated version named GlcAlift here. Ed Klotz of IBM in Carson City NV provided MPS files for the Mészáros *problematic* LPs and lengthy discussions of their properties.

Conflicts of interest: None declared.

REFERENCES

- Applegate, D. L., Cook, W., Dash, S., and Espinoza, D. G. (2007). Exact solutions to linear programming problems. *Operations Res. Lett.*, **35**, 693–699.
- Cook, W., Dash, S., and Espinoza, D. G. (2007). Exact solutions to linear programming problems. *Operations Research Letters*.
- CPLEX (2014). IBM ILOG CPLEX optimizer. <http://www.ibm.com/software/commerce/optimization/cplex-optimizer/>.
- Gill, P. E., Murray, W., Saunders, M. A., and Wright, M. H. (1989). A practical anti-cycling procedure for linear and nonlinear programming. *Math. Program.*, **45**, 437–474.
- Gill, P. E., Murray, W., and Saunders, M. A. (2005). SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, **47**(1), 99–131. SIGEST article.
- Gudmundsson, S. and Thiele, I. (2010). Computationally efficient flux variability analysis. *BMC Bioinformatics*, **11**(489), 3 pp.
- Gurobi (2014). Gurobi optimization system for linear and integer programming. <http://www.gurobi.com>.
- Kahan, W. (2011). Desperately needed remedies for the undebuggability of large floating-point computations in science and engineering. IFIP/SIAM/NIST Working Conference on Uncertainty Quantification in Scientific Computing, Boulder CO, <http://www.eecs.berkeley.edu/~wkahan/Boulder.pdf>.
- Klotz, E. (2014). Private communication, IBM, Carson City, NV.
- Koch, T. (2004). The final NETLIB-LP results. *Operations Research Letters*, **32**, 138–142.
- Lerman, J. A. (2012). Private communication, Department of Bioengineering, University of California San Diego, CA.
- Lerman, J. A., Hyduke, D. R., Latif, H., Portnoy, V. A., Lewis, N. E., Orth, J. D., Schrimpe-Rutledge, A. C., Smith, R. D., Adkins, J. N., Zengler, K., and Palsson, B. Ø. (2012). In silico method for modelling metabolism and gene product expression at genome scale. *Nature Communications*, **3**(929), 10 pp.
- LUSOL (2013). Sparse LU factorization package. <http://www.stanford.edu/group/SOL/software/lusol>.
- Mészáros, C. (2004). A collection of challenging LP problems. http://www.sztaki.hu/~meszaros/public_ftp/lptestset/problematic.
- MOSEK (2014). MOSEK Optimization Software. <http://www.mosek.com/>.
- MPS (1960). Input format for LP data. <http://lpsolve.sourceforge.net/5.5/mps-format.htm>.
- Murtagh, B. A. and Saunders, M. A. (1978). Large-scale linearly constrained optimization. *Math. Program.*, **14**, 41–72.
- Murtagh, B. A. and Saunders, M. A. (1982). A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. *Math. Program. Study*, **16**, 84–117.
- Netlib (1988). Netlib collection of LP problems in MPS format. <http://www.netlib.org/lp/data>.
- O'Brien, E. J., Lerman, J. A., Chang, R. L., Hyduke, D. R., and Palsson, B. Ø. (2013). Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction. *Molecular Systems Biology*, **9**(693), 13 pp.
- Orth, J. D., Thiele, I., and Palsson, B. Ø. (2010). What is flux balance analysis? *Nature Biotechnology*, **28**(3), 245–248.
- Palsson, B. Ø. (2006). *Systems Biology: Properties of Reconstructed Networks*. Cambridge University Press, NY.
- QSOpt-ex (2008). Qsopt-ex: A simplex solver for computing exact rational solutions to LP problems. <http://www.math.uwaterloo.ca/~bico/qsopt/index.html>.
- SoPlex (1996). Soplex: The sequential object-oriented simplex solver. <http://soplex.zib.de>.
- Sun, Y., Fleming, R. M. T., Thiele, I., and Saunders, M. A. (2013). Robust flux balance analysis of multiscale biochemical reaction networks. *BMC Bioinformatics*, **14**(240).
- Thiele, I., Fleming, R. M. T., Que, R., Bordbar, A., Diep, D., and Palsson, B. Ø. (2012). Multiscale modeling of metabolism and macromolecular synthesis in E. coli and its application to the evolution of codon usage. *PLOS ONE*, **7**(9), 18 pp.
- TOMLAB (2014). Optimization environment for MATLAB. <http://tomopt.com>.