The value of an advanced basis crash for the dual revised simplex method

Julian Hall and Ivet Galabova

School of Mathematics
University of Edinburgh

jajhall@ed.ac.uk; I.L.Galabova@sms.ed.ac.uk

Optimization Methods and Software
Havana, Cuba

December 20, 2017

THE UNIVERSITY of EDINBURGH

Overview

- Established primal simplex crash procedures
- Applicability to the dual simplex algorithm
- Implications for initialising edge weights
- Open-source linear optimization package `HiGHS`
- Results
- Conclusions

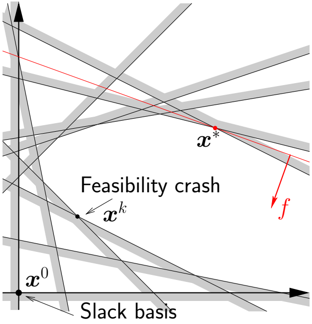
Solving LP problems: Primal overview

- Inequality problem is

$$\text{minimize } f = \boldsymbol{c}^T\boldsymbol{x} \quad \text{subject to } A\boldsymbol{x} \le \boldsymbol{b} \quad \boldsymbol{x} \ge \boldsymbol{0} \quad \text{where } A \in \mathbb{R}^{m\times n}$$

- Add m slack variables $\{x_{n+1}, \ \ldots, \ x_{n+m}\}$ to give

$$\text{minimize } f = \boldsymbol{c}^T\boldsymbol{x} \quad \text{subject to } \begin{bmatrix} A & I \end{bmatrix} \boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{x} \ge \boldsymbol{0}$$



- Simplex algorithm steps from one vertex to another until an optimal vertex is reached
- First task is to reach a feasible vertex: **Phase 1**
- Origin corresponds to an "all-slack" basis $\mathcal{B} = \{n+1, \ \ldots, \ n+m\}$
  - Computationally convenient starting vertex
  - Cost of Phase 1 may be significant
- Classic **crash** aims to find (near-)feasible **advanced basis** $\mathcal{B}$

Solving LP problems: Optimality conditions at a vertex

- General bounded equality problem is

$$\text{minimize } f = \boldsymbol{c}^T\boldsymbol{x} \quad \text{subject to } \begin{bmatrix} A & I \end{bmatrix}\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u}$$

- A **basic solution** correponds to a partition $\mathcal{B} \cup \mathcal{N}$ of $\{1, \dots, n+m\}$ into **basic** components $\boldsymbol{x}_B$, and **nonbasic** components $\boldsymbol{x}_N$ at lower or upper bounds
- Equations partitioned as $B\boldsymbol{x}_B + N\boldsymbol{x}_N = \boldsymbol{b}$ with nonsingular **basis matrix** B
- Substituting $\boldsymbol{x}_B = B^{-1}(\boldsymbol{b} - N\boldsymbol{x}_N) = \widehat{\boldsymbol{b}} - B^{-1}N\boldsymbol{x}_N$ into the objective

$$f = \boldsymbol{c}_B^T\boldsymbol{x}_B + \boldsymbol{c}_N^T\boldsymbol{x}_N \qquad \text{gives} \qquad f = \widehat{f} + \widehat{\boldsymbol{c}}_N^T\boldsymbol{x}_N$$

where $\widehat{f} = \boldsymbol{c}_B^T\widehat{\boldsymbol{b}}$ and $\widehat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{c}_B^T B^{-1}N$ is the vector of **reduced costs**

- Vertex is optimal if there is

Primal feasibility $\boldsymbol{l} \le \widehat{\boldsymbol{b}} \le \boldsymbol{u}$ Dual feasibility $\begin{cases} \widehat{c}_j \ge 0 & x_j = l_j \\ \widehat{c}_j \le 0 & x_j = u_j \end{cases} \quad j \in \mathcal{N}$

Solving LP problems: Primal or dual simplex?

Primal simplex algorithm

- Traditional variant
  - Assume primal feasibility
  - Seek dual feasibility
- Solution generally not primal feasible when (primal) LP is tightened

Dual simplex algorithm

- Preferred variant
  - Assume dual feasibility
  - Seek primal feasibility
- Easier to get dual feasibility
- More progress in many iterations
- Solution dual feasible when primal LP is tightened

Solving LP problems: Finding primal feasibility

"Assume primal feasibility"

- Finding a feasible point is generally no easier than finding an optimal vertex
- Start from an all-slack basis
  - Try using the simplex algorithm to minimize the sum of infeasibilities

$$f_I(\boldsymbol{x}) = \sum_{i\in\mathcal{B}} \max(l_i - x_i, 0, x_i - u_i)$$

    Could perform many iterations and *still* be far from optimality

  - Possibly more efficient to use a penalty function

$$f = \boldsymbol{c}^T\boldsymbol{x} + \mu f_I(\boldsymbol{x})$$

- Or **crash start** from an **advanced basis** which is "more likely to be feasible"

Solving LP problems: Primal simplex crash

$$\text{minimize } f = \boldsymbol{c}^T\boldsymbol{x} \quad \text{subject to } \begin{bmatrix} A & I \end{bmatrix}\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u}$$

Type	Name	Range	Feasibility priority
0	Fixed variable	$l_j = x_j = u_j$	Lowest
1	Boxed variable	$l_j \le x_j \le u_j$	Lower
2	One-sided variable	$l_j \le x_j$ or $x_j \le u_j$	Higher
3	Free variable	$-\infty \le x_j \le \infty$	Highest

- An all-slack basis may have
  - Many fixed and boxed variables
  - Few one-sided and free variables
- Aim of classical primal crash
  - Replace slack variables of low priority by original variables of high priority
  - Maintain condition and sparsity of B

Solving LP problems: Primal simplex crash

Bixby (1992)

- Aims only to replace fixed slacks in $\mathcal{B}$
- Replacements chosen from original variables:
  - Prioritises free over bounded, bounded over boxed
  - Breaks ties via bound and cost metric
- Ensures near-triangular, well-conditioned B

Maros and Mitra (1998)

- Aims to replace all slacks in $\mathcal{B}$
- Replacements chosen from original variables:
  - Must have higher feasibility priority than slack it replaces
- Ensures triangular, well-conditioned B

Solving LP problems: Finding dual feasibility

$$\text{minimize } f = \boldsymbol{c}^T\boldsymbol{x} \quad \text{subject to } \begin{bmatrix} A & I \end{bmatrix}\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$$

- Vertex is optimal if there is

  Primal feasibility $\boldsymbol{l} \leq \widehat{\boldsymbol{b}} \leq \boldsymbol{u}$ Dual feasibility $\begin{cases} \widehat{c}_j \geq 0 & x_j = l_j \\ \widehat{c}_j \leq 0 & x_j = u_j \end{cases} \quad j \in \mathcal{N}$
- Finding a dual feasible point can be easier than finding a primal feasible point
  - Suppose all of $\boldsymbol{l}$ and $\boldsymbol{u}$ are finite for original variables
    - All-slack basis is dual feasible
    - Assign $x_j = l_j$ $(x_j = u_j)$ if $\widehat{c}_j \geq 0$ $(\widehat{c}_j < 0)$, $j \in \mathcal{N}$
  - Generally:
    - Good to have fixed and boxed variables in $\mathcal{N}$
    - Bad to have one-sided and free variables in $\mathcal{N}$
  - Dual feasibility priorities are the same as primal!
- Has this been done before?

Solving LP problems: Initial edge weights at advanced basis

Primal simplex edge weights

- Primal simplex algorithm chooses $q = \underset{j\in\mathcal{N}}{\text{argmin}}\ \dfrac{\widehat{c}_j}{w_j}$ to enter $\mathcal{B}$
- Values $w_j = 1$ are traditional
- Better are (measures of) $\|\widehat{\boldsymbol{a}}_j\|_2$, where $\widehat{\boldsymbol{a}}_j = B^{-1}\boldsymbol{a}_j$
  - "Devex" is common default: initial weights are $w_j = 1$; Fine if $B \neq I$

Dual simplex edge weights

- Dual simplex algorithm chooses $p = \underset{i\in\mathcal{B}}{\text{argmin}}\ \dfrac{\widehat{b}_i}{w_i}$ to leave $\mathcal{B}$
- Values $w_i = 1$ are traditional
- Better are (measures of) $\|\widehat{\boldsymbol{\pi}}_i\|_2$, where $\widehat{\boldsymbol{\pi}}_i = B^{-T}\boldsymbol{e}_i$
  - "Dual steepest edge" is common default: (initial) weights are $\|\widehat{\boldsymbol{\pi}}_i\|_2$;
  - Computational cost when $B \neq I$?

Solving LP problems: Computing initial dual steepest edge weights

- Solving $B^T\widehat{\boldsymbol{\pi}}_i = \boldsymbol{e}_i$ for $i = 1\ldots, m$ to get $w_i = \|\widehat{\boldsymbol{\pi}}_i\|_2$ looks expensive!
- Possible trick (Davis)
  - Observe
$$w_i^2 \;=\; \|\widehat{\boldsymbol{\pi}}_i\|_2^2 \;=\; (B^{-T}\boldsymbol{e}_i)^T B^{-T}\boldsymbol{e}_i \;=\; \boldsymbol{e}_i^T B^{-1} B^{-T}\boldsymbol{e}_i$$
  - Form $LL^T = B^T B$, then
$$w_i^2 = \boldsymbol{e}_i^T(L^{-T}L^{-1}\boldsymbol{e}_i)$$
  - Solving for just one component of $L^{-T}L^{-1}\boldsymbol{e}_i$ is *very* fast with `Cholmod`
- Alternative
  - Observe that B is (near-)triangular
  - Exploit hyper-sparsity when solving each $B^T\widehat{\boldsymbol{\pi}}_i = \boldsymbol{e}_i$

HiGHS (2011–date)

Overview

- Written in C++ to study parallel simplex
- Dual simplex with steepest edge and BFRT
- Forrest-Tomlin update
  - complex and inherently serial
  - efficient and numerically stable

Concept

- High performance serial solver (`hsol`)
- Exploit limited task and data parallelism in standard dual RSM iterations (`sip`)
- Exploit greater task and data parallelism via minor iterations of dual SSM (`pami`)
- Test-bed for research
- Work-horse for consultancy

Huangfu, H and Galabova (2011–date)

HiGHS: Performance and reliability

Extended testing using 159 test problems

- 98 Netlib
- 16 Kennington
- 4 Industrial
- 41 Mittelmann

Exclude 7 which are "hard"

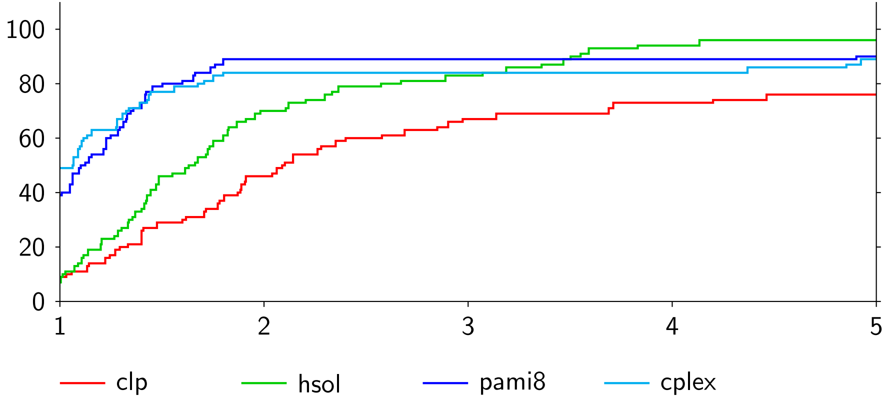
Performance

Benchmark against `clp` (v1.16) and `cplex` (v12.5)

- Dual simplex
- No presolve
- No crash

Ignore results for 82 LPs with minimum solution time below 0.1s

HiGHS: Performance

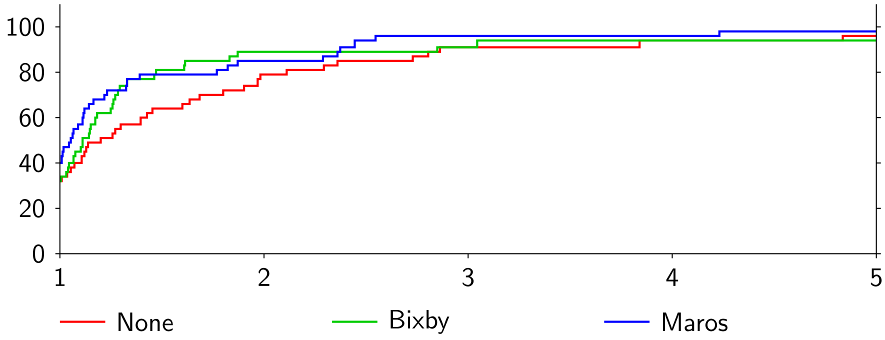


HiGHS: Crash basis analysis

For the initial basis matrix on the 70 harder LP problems

- $B = I$ for 11 LPs (Bixby) and 10 LPs (Maros)
- Average proportion of slacks is 55% (Bixby) and 33% (Maros)
- Bixby basis has fewer slacks in three cases
- Average cost of computing initial steepest edge weights is 0.06% solution time

HiGHS: Crash performance



- Bixby crash improved performance by 15%: best is a factor of 4.1
- Maros crash improved performance by 21%: best is a factor of 19.

HiGHS: Development

2016

- Presolve (Galabova)
- Crash (Hall)

2017

- SCIP interface (Hall)
- Prototype MIP solver (Galabova)

2018

- Interior point solver (Schork)
- Prototype QP solver (Feldmeier)

Long term

Replacement for `clp`?

Academic involvement

- SCIP
- Open source

Commercial involvement

- Cargill (feed formulation)
- Google (techniques in `glop`)
- Consultancy

Conclusions

- Dual advanced basis crash has same criteria as primal
- Initialising dual steepest edge weights has no significant overhead
- Dual advanced basis crash of modest general value

    But of significant value for some problems

Slides: http://www.maths.ed.ac.uk/hall/OMS17