What's new in HiGHS

Julian Hall

School of Mathematics
University of Edinburgh

Julia and Optimization Days 2023

CNAM, Paris

5 October 2023

HiGHS: Open-source software for large-scale sparse linear optimization

HiGHS: **H**all, **i**vet **G**alabova, **H**uangfu, **S**chork

$$\text{minimize} \quad f = \frac{1}{2}\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} \quad \text{such that} \quad A\boldsymbol{x} = \boldsymbol{b};\ \boldsymbol{x} \geq \boldsymbol{0}, \quad x_i \in \mathbb{Z},\ \forall i \in \mathcal{I}$$

Features

- Simplex and interior point solvers for LP
- Branch-and-cut solver for MIP
- Active set solver for QP
- Written in C++
- Interfaces to other languages and systems

Availability

- Open-source (MIT license)
- No third-party code
- https://HiGHS.dev/

The world's best open-source linear optimization software

Mittelmann (2023)

HiGHS: History

HiGHS: Project

- Integration of Galabova's presolve and Huangfu's dual simplex solver (2016)
- Creation of name (2018)
- Integration of Schork's interior point solver (2019)
- Development of active set solver for QP by Feldmeier (2018–date)
- Development of new MIP solver and LP/MIP presolve by Gottwald (2020–2022)

HiGHS: JuMP

- JuMP first interested in HiGHS (2019)
- JuMP-HiGHS interface developed by Dowson and Galabova (2020)
- JuMP maintains HiGHS binaries (2021–date)
- HiGHS has become the default solver in JuMP documentation (2020–date)

HiGHS: Team

Employees

- Julian Hall: Almost full-time!
- Ivet Galabova: Integration and Development Manager
- Filippo Zanetti: IPM researcher

Contributors

- Oscar Dowson: JuMP, Julia, documentation and ...
- Michael Feldmeier: QP
- Rohit Goswami: Meson build and ...
- Luke Marshall: Column generation, decomposition and modelling language
- Stefan Vigerske: Model file reading and ...
- Others

HiGHS: Callbacks

- New in HiGHS 1.6.0 for C++ and C
- Single user-defined generic method

```
void userCallback(const int callback_type,
                  const char* message,
                  const HighsCallbackDataOut* data_out,
                  HighsCallbackDataIn* data_in,
                  void* user_callback_data);
```

- Logging callbacks
  - All HiGHS messages
  - Data logging from MIP solver
- Improving solution in MIP solver
- Interrupt from solvers for LP and MIP, but not (yet) QP
- No MIP cut callbacks (yet!)

HiGHS: A Python interface

- Not everyone uses Julia, so a Python presence for HiGHS is essential
- SciPy is a very valuable basic host for black-box solvers
- Python-based PuLP and Pyomo modelling languages want HiGHS!

highspy (WIP)

- Built using pybind11
- Allows HiGHS C++ classes, structures and enums to be used in Python
- Can be slow to extract results

pip install highspy

- Easy installation for Python users
- No need for C++ libraries
- Available for HiGHS 1.5.4

highspy modelling (WIP)

- Independent modelling with HiGHS
- Handy in Colab
- Not competing with JuMP!

HiGHS: Slow to extract results with highspy

```python
import highspy

h = highspy.Highs()
h.readModel("ml.mps")
h.run()

solution = h.getSolution()

# By using __getitem_
value = [solution.col_value[i]
         for i in range(h.getNumCol())]

# By copying first to list
col_value = list(solution.col_value)
value = [col_value[i]
         for i in range(h.getNumCol())]
```

- MIP with 32504 variables
- Solve time: 1.6s
- Extraction time:
  - Using getitem: 11s
  - Copying to list: 0.0022s

HiGHS: Documentation

Historically

- Priority has been performance
- Advanced users have worked from header files and CMake build system!

Documentation

https://ergo-code.github.io/HiGHS/

- Uses Documenter.jl
- Parses C API docstrings
- Python-based
- Colab notebooks (WIP)

- For whom are we writing documentation?
- What do we document?
- Full Python-based documentation is WIP
- Documentation for HiGHS is still not great!

HiGHS: MIP

Leona Gottwald wrote us an amazing MIP solver, but left us in autumn 2022

Solver	Gurobi	COPT	FSCIPX	FSCIP	HiGHS	SCIPC	SCIP	Cbc	Matlab
Speed	1	2.01	6.26	7.51	8.77	8.92	10.9	16.3	33.3
Solved	227	204	172	163	158	152	137	107	72

- Much scope for further improvement in HiGHS
  - HiGHS is single threaded
    - Multicore version would be great!
    - Catch up (non-deterministic) FSCIPX and FSCIP!
  - Other MIP techniques still to be added
- Debugging and development are coming slowly
- Possible PhD student starting in September 2024

HiGHS: IPM solver

Aim: Develop the world's best open-source IPM solver for

$$\text{minimize } f = \frac{1}{2}\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} \quad \text{such that } A\boldsymbol{x} = \boldsymbol{b};\ \ \boldsymbol{x} \geq \boldsymbol{0}$$

- Have to solve $\begin{bmatrix} -Q - \Theta^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Delta x} \\ \boldsymbol{\Delta y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{d} \end{bmatrix}$
- Have to write most of it from scratch due to MIT license
  - Quasi-definite decomposition code for non-diagonal Q (at least)
  - Positive definite decomposition code for Newton matrix $G = A(Q + \Theta^{-1})^{-1}A^T$ if
    - Q is diagonal
    - $Q = 0$ (LP) so $G = A\Theta A^T$
  - **Not** graph partitioning code - since latest Metis can be used
- The team
  - Jacek Gondzio: IPM consultant
  - Filippo Zanetti: IPM researcher (2023–)
  - Yanyu Zhou: PhD student (2024–)

HiGHS: IPM solver

First steps

- Written a prototype solver `ProtoIPM` (Zanetti)
- Experimented with existing direct solvers (H and Zhou)
  - `Cholmod` for Newton system
  - `SSIDS`, `MA86` and `QDLDL` for Newton and augmented system

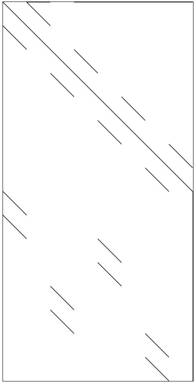
What are their strengths/limitations?

Motivating example

- Industrial client reported HiGHS IPM failing

  7,504,855 rows, 3,752,404 columns; 18,742,867 nonzeros

- LP identified as having a single column with $m/2$ nonzeros
- Can it be solved by IPM using direct decomposition but avoiding the dense column?
- Study "toy" instance of industrial model



2689 rows
1345 columns

672 of count	3
336 of count	4
336 of count	5
1 of count	1344

6384 nonzeros

IPM solver: Dense columns

- **Problem:**
  - If A has a single full column, then $G = A\Theta A^T$ is full
  - Cost of forming and factorizing G is prohibitive
  - Same issue with dense columns—containing $O(m)$ nonzeros—in A
- **Solution:**
  - If A_d are the dense columns in A, consider $G = A_s\Theta_s A_s^T + A_d\Theta_d A_d^T$
  - Represent G using $G_s \equiv A_s\Theta_s A_s^T = LL^T$ and Sherman–Morrison–Woodbury formula
  - Needs G_s to be nonsingular
- **OK in theory:**
  - A contains I and this is obviously sparse
  - Entries of Θ_s are positive
  - G_s is positive definite

What can go wrong?

IPM solver: Dense columns

Consider the LP

$$\text{minimize} \quad f = \begin{bmatrix}-2\\-1\end{bmatrix}^T \begin{bmatrix}x_1\\x_2\end{bmatrix} \quad \text{such that} \begin{bmatrix}1 & -1 & 1 & \\ 1 & 1 & & 1\end{bmatrix} \begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} = \begin{bmatrix}2\\4\end{bmatrix}; \quad \begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} \geq \mathbf{0}$$

Iter	x_1	x_2	Θ_1	Θ_2	Θ_3	Θ_4
1	2.0	0.7	2	1	0.8	0.8
2	0.8	2.2	0.3	1	0.3	4
3	1.0	2.2	10^1	4	10^{-1}	10^1
4	2.9	0.9	$\mathbf{10^4}$	2	10^{-2}	10^{-1}
5	3.0	1.0	10^3	10^2	10^{-2}	10^{-2}
6	3.0	1.0	$\mathbf{10^4}$	10^3	10^{-4}	10^{-3}
7	3.0	1.0	$\mathbf{10^5}$	$\mathbf{10^4}$	10^{-5}	10^{-4}
8	3.0	1.0	$\mathbf{10^6}$	$\mathbf{10^5}$	10^{-6}	10^{-5}

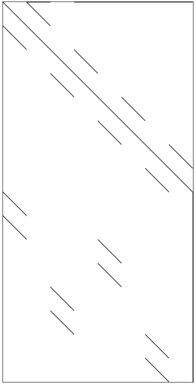
If first column is treated as dense

- $A_s = \begin{bmatrix}-1 & 1 & \\ 1 & & 1\end{bmatrix}$ is full rank
- $\Theta_s \rightarrow \begin{bmatrix}\theta_2 & 0 & 0\end{bmatrix}^T$
- $G_s \rightarrow \theta_2 \begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$

IPM solver: Dense columns

- **In general:**
  - Entries of Θ_s can range over many orders of magnitude
  - Some terms in $G_s = \sum_{i \in \mathcal{S}} \Theta_i \boldsymbol{a}_i \boldsymbol{a}_i^T$ are computationally (near-)zero
  - G_s may be (near-)singular
- Well-conditioned G_s needs m sufficiently large values in Θ_s
  - Large values of Θ_i correspond to $x_i \rightarrow x_i^* > 0$: **basic** variables
  - There are only m basic variables
  - Dense columns are usually basic!
- Choice of dense columns may be too restricted

IPM solver: Dense columns



2689 rows
1345 columns
672 of count 3
336 of count 4
336 of count 5
1 of count 1344
6384 nonzeros

How about the industrial problem?

Solver	Time	Comment
`Gurobi`	0.02	Presolve very effective
`HiGHS`	0.72	Using Schork's PCG
`ProtoIPM`	6.92	Vanilla Newton
	5.99	Newton (1 dense column)
	0.65	Augmented system

- Why is Newton little better with 1 dense column?
  - Dense column only used for first 3 of 22 iterations
  - Solver fails if the use of dense column is forced
- Why isn't `SSIDS` doing better with the augmented system?

IPM solver: Augmented system

Decompose

$$\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$$

Observation

Choosing $-\Theta_1^{-1}, \;\ldots,\; -\Theta_n^{-1}$ as pivots yields partial decomposition

$$\begin{bmatrix} I & \\ -A\Theta & I \end{bmatrix} \begin{bmatrix} -\Theta^{-1} & A^T \\ & A\Theta A^T \end{bmatrix}$$

Equivalent to the elimination that yields the Newton system

- Initial pivots come from $-\Theta^{-1}$ and cause fill-in in the $(2,2)$ block
- Hope to choose pivots from $(2,2)$ block before pivoting on dense column(s)
- Large Θ_i for basic variables yields small Θ_i^{-1}
  - General indefinite decomposition views these as bad pivots
  - Some pivots good for sparsity are postponed
  - Fill-in is greater than suggested by structural analysis
- Perturb small pivots so that they can be used

  Corresponds to solving LP with small quadratic term: **regularization**

IPM solver: Augmented system

Prototype augmented system solver for HiGHS

- Symmetric systems are special case of asymmetric systems!
- HiGHS has efficient code to decompose $B = LU$
- How does its performance compare with SSIDS?

Augmented system solvers applied to i-002-168

Solver	LP time	Factorize time	Fill-in	Residual error
SSIDS	0.62	0.53	4.71	1e-12
HiGHS	**0.35**	**0.27**	**1.36**	1e-12

- SSIDS is parallel
- SSIDS is constrained by diagonal pivots

IPM solver: Augmented system

- How general is this behaviour?
- Test single augmented system solve for Netlib and Mittelmann LPs

Solver	Factorize time	Fill-in	Residual error	Solution error
SSIDS	0.079	10.74	**2.8e-14**	7.4e-14
HiGHS	**0.030**	**2.82**	1.4e-12	2.6e-14

- Transformed gradware into software
- World-beating open-source performance
- Using software to create Impact
- Developing a better IPM solver

  May even generate research output!
- Like running a small software company
- Hugely fulfilling!

https://HiGHS.dev/

Q. Huangfu and J. A. J. Hall.
Novel update techniques for the revised simplex method.
Computational Optimization and Applications, 60(4):587–608, 2015.

Q. Huangfu and J. A. J. Hall.
Parallelizing the dual revised simplex method.
Mathematical Programming Computation, 10(1):119–142, 2018.

L. Schork and J. Gondzio.
Implementation of an interior point method with basis preconditioning.
Mathematical Programming Computation, pages 1–33, 2020.