Computational issues in linear programming

Julian Hall

School of Mathematics

University of Edinburgh

15th May 2007

Overview

- Introduction to linear programming and solution methods
- Computational issues
  - Simplex method: hyper-sparsity and parallelism
  - IPM: preconditioning for iterative solution of equations
- Future directions

Linear programming problems

$$\begin{array}{ll} \text{minimize} & f = \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{x} \geq \boldsymbol{0} \end{array}$$

- Linear programming (LP) is the fundamental model in optimal decision-making
- Solution techniques
  - Simplex method (1947)
  - Interior point method (1984)

Linear programming problems

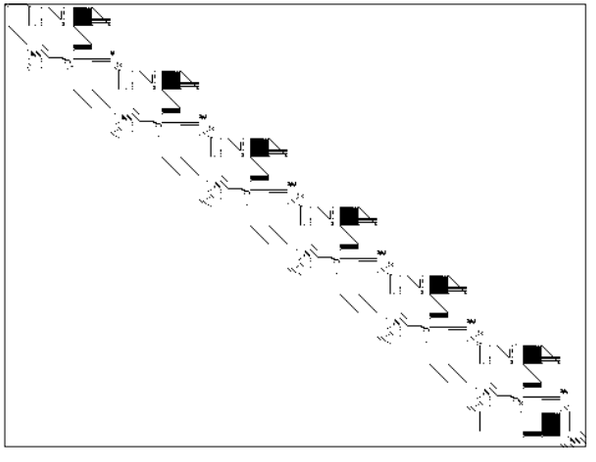
$$\begin{array}{ll} \text{minimize} & f = \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{x} \geq \boldsymbol{0} \end{array}$$

- Linear programming (LP) is the fundamental model in optimal decision-making
- Solution techniques
  - Simplex method (1947)
  - Interior point method (1984)
- Large problems have
  - $10^3 - 10^7$ variables
  - $10^3 - 10^7$ constraints

Linear programming problems

$$\begin{array}{ll} \text{minimize} & f = \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{x} \geq \boldsymbol{0} \end{array}$$

- Linear programming (LP) is the fundamental model in optimal decision-making
- Solution techniques
  - Simplex method (1947)
  - Interior point method (1984)
- Large problems have
  - $10^3 - 10^7$ variables
  - $10^3 - 10^7$ constraints
- Matrix A is sparse

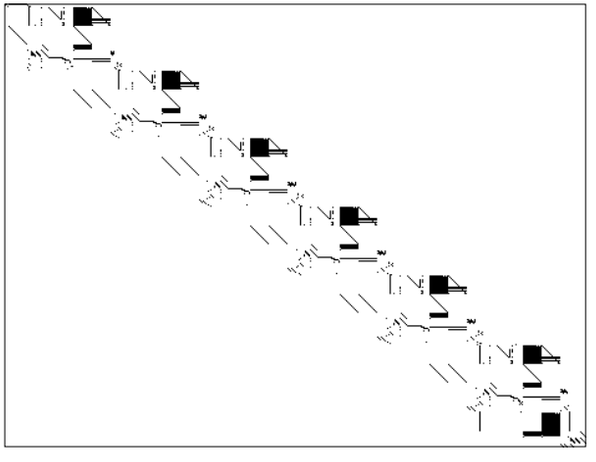


STAIR: 356 rows, 467 columns and 3856 nonzeros

Linear programming problems

$$\begin{array}{ll} \text{minimize} & f = \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \quad \boldsymbol{x} \geq \boldsymbol{0} \end{array}$$

- Linear programming (LP) is the fundamental model in optimal decision-making
- Solution techniques
  - Simplex method (1947)
  - Interior point method (1984)
- Large problems have
  - $10^3 - 10^7$ variables
  - $10^3 - 10^7$ constraints
- Matrix A is sparse



STAIR: 356 rows, 467 columns and 3856 nonzeros

- Structure as much as size determines the computational challenge

Satisfying the optimality conditions for LP

Lagrange multipliers $\boldsymbol{y}$ and $\boldsymbol{s}$ exist such that the following conditions hold

$$
\begin{array}{rcll}
A\boldsymbol{x} = \boldsymbol{b} & & \boldsymbol{x} \geq \boldsymbol{0} & \text{primal feasibility} \\
A^T\boldsymbol{y} + \boldsymbol{s} = \boldsymbol{c} & & \boldsymbol{s} \geq \boldsymbol{0} & \text{dual feasibility} \\
\boldsymbol{s}^T\boldsymbol{x} = 0 & (\Rightarrow \quad x_j s_j = 0 \quad \forall j) & & \text{complementarity condition}
\end{array}
$$

Satisfying the optimality conditions for LP

Lagrange multipliers $\boldsymbol{y}$ and $\boldsymbol{s}$ exist such that the following conditions hold

$$\begin{array}{rcll} A\boldsymbol{x} = \boldsymbol{b} & & \boldsymbol{x} \geq \boldsymbol{0} & \text{primal feasibility} \\ A^T\boldsymbol{y} + \boldsymbol{s} = \boldsymbol{c} & & \boldsymbol{s} \geq \boldsymbol{0} & \text{dual feasibility} \\ \boldsymbol{s}^T\boldsymbol{x} = 0 & (\Rightarrow & x_j s_j = 0 \quad \forall j) & \text{complementarity condition} \end{array}$$

Simplex method

- Uses a **combinatorial** approach
- Modifies a partition $\mathcal{B} \cup \mathcal{N}$ of variables until $\boldsymbol{s} \geq \boldsymbol{0}$
- **Moves along edges** of the feasible region
- **Terminates** at an optimal vertex

Satisfying the optimality conditions for LP

Lagrange multipliers $\boldsymbol{y}$ and $\boldsymbol{s}$ exist such that the following conditions hold

$$\begin{aligned} A\boldsymbol{x} &= \boldsymbol{b} & \boldsymbol{x} &\geq \boldsymbol{0} && \text{primal feasibility} \\ A^T\boldsymbol{y} + \boldsymbol{s} &= \boldsymbol{c} & \boldsymbol{s} &\geq \boldsymbol{0} && \text{dual feasibility} \\ \boldsymbol{s}^T\boldsymbol{x} &= 0 \quad (\Rightarrow & x_j s_j &= 0 \quad \forall j) && \text{complementarity condition} \end{aligned}$$

Simplex method

- Uses a **combinatorial** approach
- Modifies a partition $\mathcal{B} \cup \mathcal{N}$ of variables until $\boldsymbol{s} \geq \boldsymbol{0}$
- **Moves along edges** of the feasible region
- **Terminates** at an optimal vertex

Interior point methods (IPM)

- Use an **iterative** approach
- Reduce $x_j s_j = \mu$ to zero $\forall j$
- **Move through the interior** of the feasible region
- **Converge** to an optimal vertex

Simplex vs interior point methods

Simplex method

- Each iteration
  - Form $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
  - Each $[\, B \quad N \,]$ is a partition of A
  - $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ (and $\boldsymbol{r}_\pi$) are **sparse**

Simplex vs interior point methods

Simplex method

- Each iteration
  - Form $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
  - Each $[\, B \quad N \,]$ is a partition of A
  - $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ (and $\boldsymbol{r}_\pi$) are **sparse**

Interior point methods

- Each iteration
  - Form $(A\Theta A^T)^{-1}\boldsymbol{r}$ or $\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}^{-1} \boldsymbol{r}$
  - Diagonal matrix Θ is iteration-dependent
  - $\boldsymbol{r}$ is **full**

Simplex vs interior point methods

Simplex method

- Each iteration
  - Form $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
  - Each $[\, B \quad N \,]$ is a partition of A
  - $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ (and $\boldsymbol{r}_\pi$) are **sparse**

- Complexity
  - Theory: $O(2^n)$ iterations
  - Practice: $O(n)$ iterations

Interior point methods

- Each iteration
  - Form $(A\Theta A^T)^{-1}\boldsymbol{r}$ or $\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}^{-1} \boldsymbol{r}$
  - Diagonal matrix Θ is iteration-dependent
  - $\boldsymbol{r}$ is **full**

Simplex vs interior point methods

Simplex method

- Each iteration
  - Form $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
  - Each $[\, B \quad N \,]$ is a partition of A
  - $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ (and $\boldsymbol{r}_\pi$) are **sparse**

- Complexity
  - Theory: $O(2^n)$ iterations
  - Practice: $O(n)$ iterations

Interior point methods

- Each iteration
  - Form $(A\Theta A^T)^{-1}\boldsymbol{r}$ or $\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}^{-1} \boldsymbol{r}$
  - Diagonal matrix Θ is iteration-dependent
  - $\boldsymbol{r}$ is **full**

- Complexity
  - Theory: $O(\sqrt{n}\log n)$ iterations
  - Practice: $O(\log n)$ iterations

Simplex vs interior point methods

Simplex method

- Each iteration
  - Form $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
  - Each $[\, B \quad N \,]$ is a partition of A
  - $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ (and $\boldsymbol{r}_\pi$) are **sparse**
- Complexity
  - Theory: $O(2^n)$ iterations
  - Practice: $O(n)$ iterations

Interior point methods

- Each iteration
  - Form $(A\Theta A^T)^{-1}\boldsymbol{r}$ or $\begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}^{-1} \boldsymbol{r}$
  - Diagonal matrix Θ is iteration-dependent
  - $\boldsymbol{r}$ is **full**
- Complexity
  - Theory: $O(\sqrt{n}\log n)$ iterations
  - Practice: $O(\log n)$ iterations

Questions

- Why is the simplex method still competitive?
- What are the computational challenges?

Simplex method and hyper-sparsity

Simplex method and hyper-sparsity

- Major computational cost each iteration is forming $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
- Vectors $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ are always sparse
- Vector $\boldsymbol{r}_\pi$ may be sparse

Simplex method and hyper-sparsity

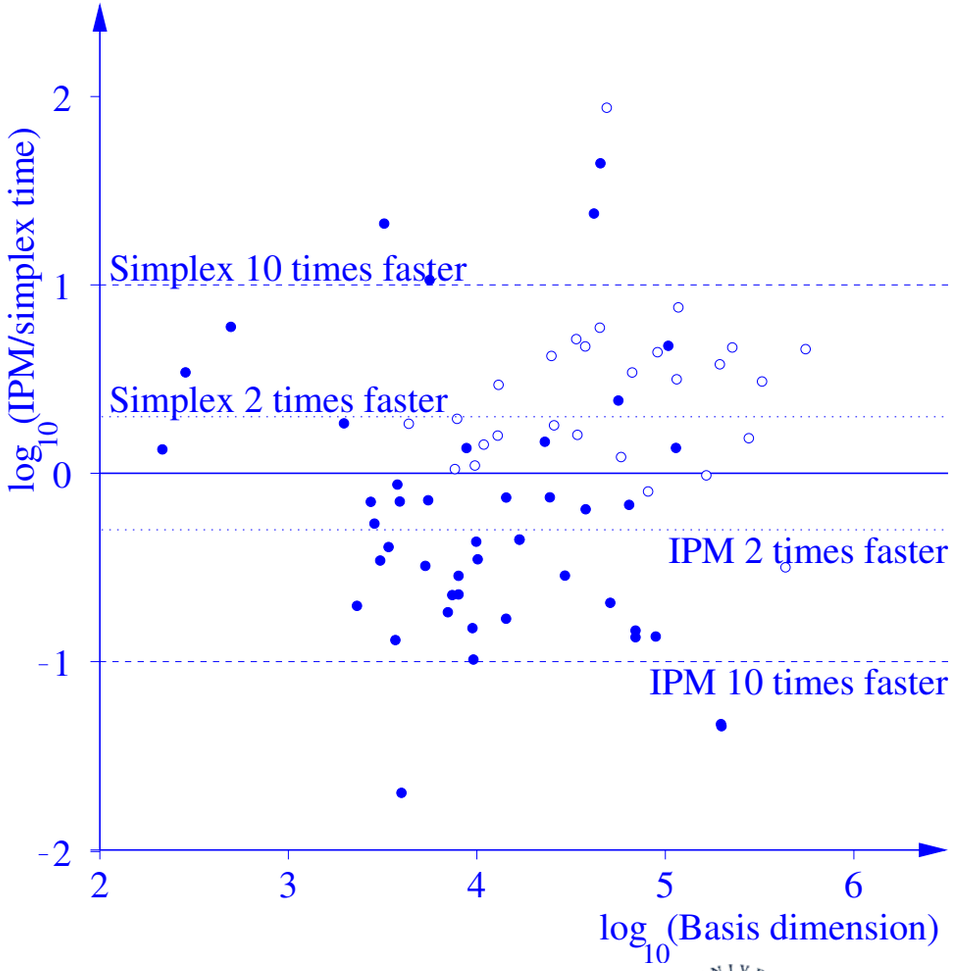
- Major computational cost each iteration is forming $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
- Vectors $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ are always sparse
- Vector $\boldsymbol{r}_\pi$ may be sparse
- If the results of these operations are (usually) sparse then LP is said to be **hyper-sparse**

Simplex method and hyper-sparsity

- Major computational cost each iteration is forming $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
- Vectors $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ are always sparse
- Vector $\boldsymbol{r}_\pi$ may be sparse
- If the results of these operations are (usually) sparse then LP is said to be **hyper-sparse**
- Essence: LP structure means B^{-1} **is sparse**

Simplex method and hyper-sparsity

- Major computational cost each iteration is forming $B^{-1}\boldsymbol{r}_F$, $\boldsymbol{r}_B^T B^{-1}$ and $\boldsymbol{r}_\pi^T N$
- Vectors $\boldsymbol{r}_F$, $\boldsymbol{r}_B$ are always sparse
- Vector $\boldsymbol{r}_\pi$ may be sparse
- If the results of these operations are (usually) sparse then LP is said to be **hyper-sparse**
- Essence: LP structure means B^{-1} **is sparse**
- When exploiting hyper-sparsity
  - For hyper-sparse LP problems (○)
    Simplex method typically better than interior point methods
  - For other LPs (●)
    Interior point methods frequently better than simplex method



Exploiting hyper-sparsity

- Traditional technique for solving $B\boldsymbol{x} = \boldsymbol{b}$ by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$:

```
do k = 1, r
    b_{p_k} := b_{p_k} / η_k
    b := b − b_{p_k} η_k
end do
```

Exploiting hyper-sparsity

- Traditional technique for solving $B\boldsymbol{x} = \boldsymbol{b}$ by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$:

```
do k = 1, r
   b_{p_k} := b_{p_k} / η_k
   b := b − b_{p_k} η_k
end do
```

- When $\boldsymbol{b}$ is sparse there is no need to apply $\boldsymbol{\eta}_k$ if b_{p_k} is zero:

```
do k = 1, r
   if (b_{p_k} .ne. 0) then
      b_{p_k} := b_{p_k} / η_k
      b := b − b_{p_k} η_k
   end if
end do
```

Exploiting hyper-sparsity

- Traditional technique for solving $B\boldsymbol{x} = \boldsymbol{b}$ by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$:

```
do k = 1, r
   b_{p_k} := b_{p_k} / η_k
   b := b − b_{p_k} η_k
end do
```

- When $\boldsymbol{b}$ is sparse there is no need to apply $\boldsymbol{\eta}_k$ if b_{p_k} is zero:

```
do k = 1, r
   if (b_{p_k} .ne. 0) then
      b_{p_k} := b_{p_k} / η_k
      b := b − b_{p_k} η_k
   end if
end do
```

- When $\boldsymbol{x}$ is sparse, the dominant cost is the test for zero

Exploiting hyper-sparsity

- Traditional technique for solving $B\boldsymbol{x} = \boldsymbol{b}$ by transforming $\boldsymbol{b}$ into $\boldsymbol{x}$:

```
do k = 1, r
   b_{p_k} := b_{p_k} / η_k
   b := b − b_{p_k} η_k
end do
```

- When $\boldsymbol{b}$ is sparse there is no need to apply $\boldsymbol{\eta}_k$ if b_{p_k} is zero:

```
do k = 1, r
   if (b_{p_k} .ne. 0) then
      b_{p_k} := b_{p_k} / η_k
      b := b − b_{p_k} η_k
   end if
end do
```

- When $\boldsymbol{x}$ is sparse, the dominant cost is the test for zero
- Identify vectors $\boldsymbol{\eta}_k$ to be applied at a cost proportional to arithmetic operations
  - Gilbert and Peierls (1988)
  - Hall and McKinnon (2005)

Parallel simplex: major review by Hall (2006)

Parallel simplex: major review by Hall (2006)

Parallel standard (tableau) simplex method

- Good parallel efficiency achieved
- Totally uncompetitive with **serial** (revised) simplex method without prohibitive resources

Parallel simplex: major review by Hall (2006)

Parallel standard (tableau) simplex method

- Good parallel efficiency achieved
- Totally uncompetitive with **serial** (revised) simplex method without prohibitive resources

Data parallel (revised) simplex method

- Only immediate parallelism is in forming $\boldsymbol{r}_\pi^T N$
- When $n \gg m$, forming $\boldsymbol{r}_\pi^T N$ dominates the cost of iterations: significant speed-up achieved
  Bixby and Martin (2000)
- One "fully" parallel implementation: no speed-up achieved
  Shu (1995)

Task parallel revised simplex method

Components of different iterations are overlapped (within algorithmic constraints)

- **Wunderling (1996)**
  "Fully" parallel for only two processors: good results only when $n \gg m$
- **ASYNPLEX: Hall and McKinnon (1995)**
  Fully parallel (inefficient) revised simplex variant: speed-up of up to 5 but numerically **unstable**
- **PARSMI: Hall and McKinnon (1996)**
  Fully parallel revised simplex variant: speed-up of 2 but numerically **unstable**
- **SYNPLEX: Hall (2005)**
  "Fully" parallel revised simplex variant: speed-up of 3 and numerically **stable**

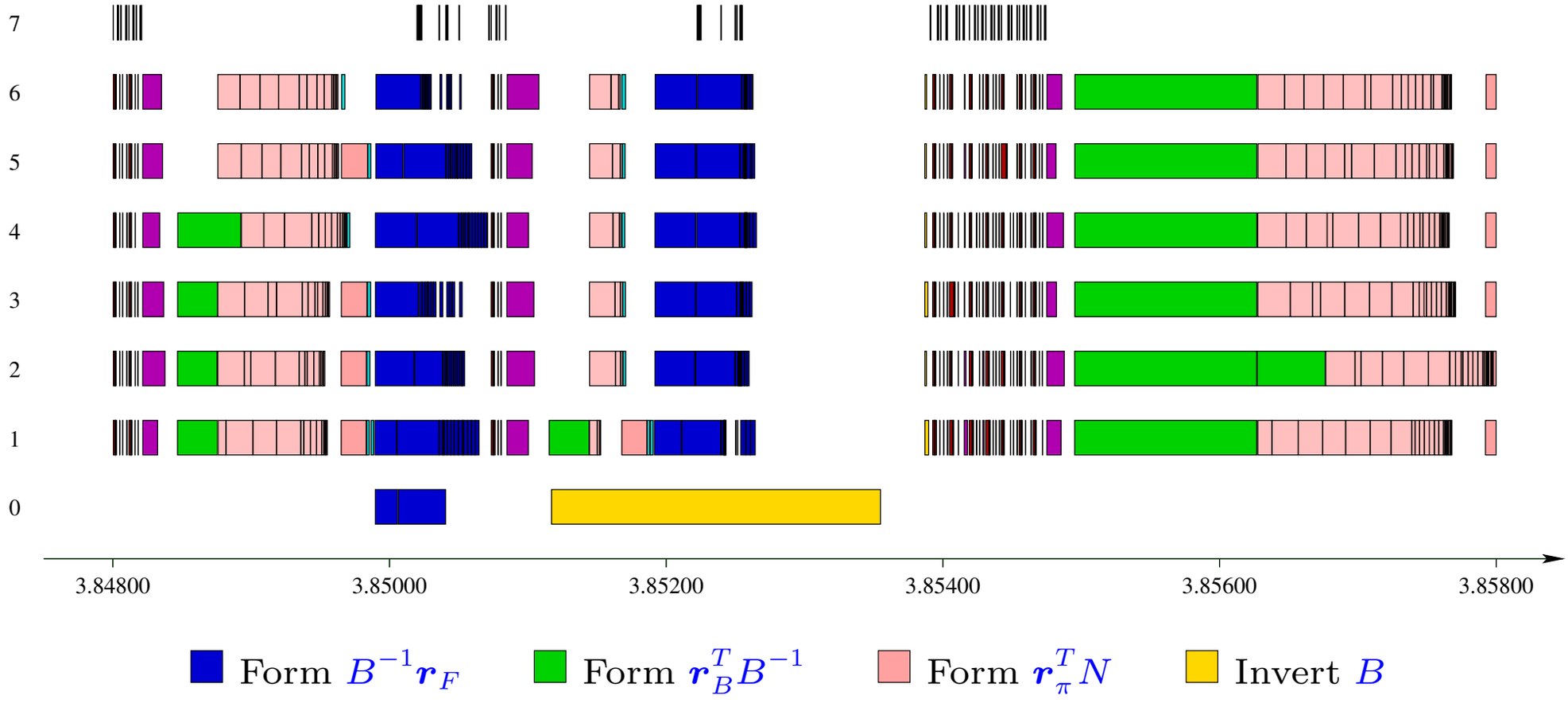
Task parallel revised simplex method

Components of different iterations are overlapped (within algorithmic constraints)

- **Wunderling (1996)**
  "Fully" parallel for only two processors: good results only when $n \gg m$
- **ASYNPLEX: Hall and McKinnon (1995)**
  Fully parallel (inefficient) revised simplex variant: speed-up of up to 5 but numerically **unstable**
- **PARSMI: Hall and McKinnon (1996)**
  Fully parallel revised simplex variant: speed-up of 2 but numerically **unstable**
- **SYNPLEX: Hall (2005)**
  "Fully" parallel revised simplex variant: speed-up of 3 and numerically **stable**

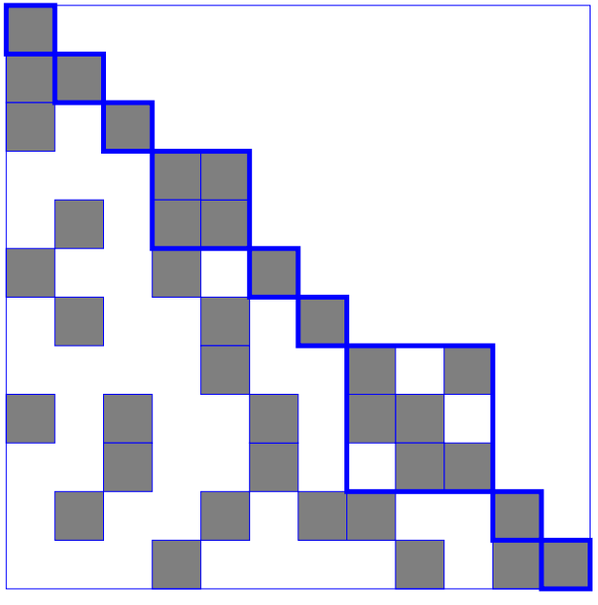
All data and task parallel implementations compromised by serial inversion of B

Need to parallelise inversion of B for SYNPLEX



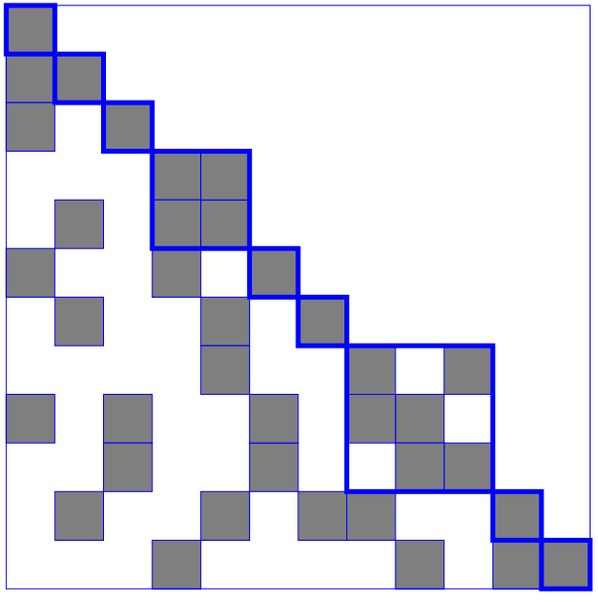
Price of numerical stability and serial inversion is processor idleness

Basis matrix inversion: the nature of the challenge



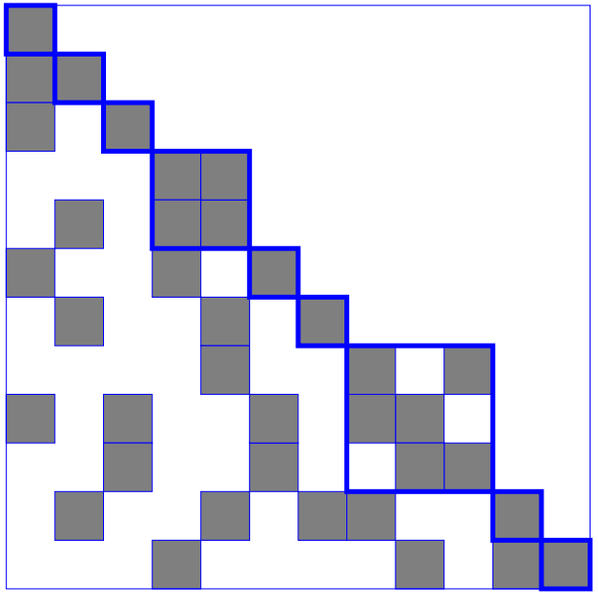
- Consider optimal block triangular form (Tarjan) of B
  - All diagonal blocks are 1×1 (trivial) or irreducible
  - For LU factors of B, elimination is restricted to diagonal blocks

Basis matrix inversion: the nature of the challenge



- Consider optimal block triangular form (Tarjan) of B
  - All diagonal blocks are 1×1 (trivial) or irreducible
  - For LU factors of B, elimination is restricted to diagonal blocks
- Total cost of inverting B depends on
  - Cost of finding irreducible blocks
  - Cost of factorizing irreducible blocks

Basis matrix inversion: the nature of the challenge



- Consider optimal block triangular form (Tarjan) of B
  - All diagonal blocks are 1×1 (trivial) or irreducible
  - For LU factors of B, elimination is restricted to diagonal blocks
- Total cost of inverting B depends on
  - Cost of finding irreducible blocks
  - Cost of factorizing irreducible blocks
- Relative number/size of 1×1 and irreducible blocks is closely related to hyper-sparsity of the LP

Tomlin INVERT (1972)

- Matrix inversion procedure designed for the revised simplex method

Tomlin INVERT (1972)

- Matrix inversion procedure designed for the revised simplex method
- Operates in two phases

  Triangularisation:
  - Active row/column **singleton** entries are identified as pivots
    until all active rows/columns have at least *two* active nonzeros
  - Corresponds to Markowitz pivot selection so long as count is zero
  - Yields approximate Tarjan form at *much* lower cost

Tomlin INVERT (1972)

- Matrix inversion procedure designed for the revised simplex method
- Operates in two phases

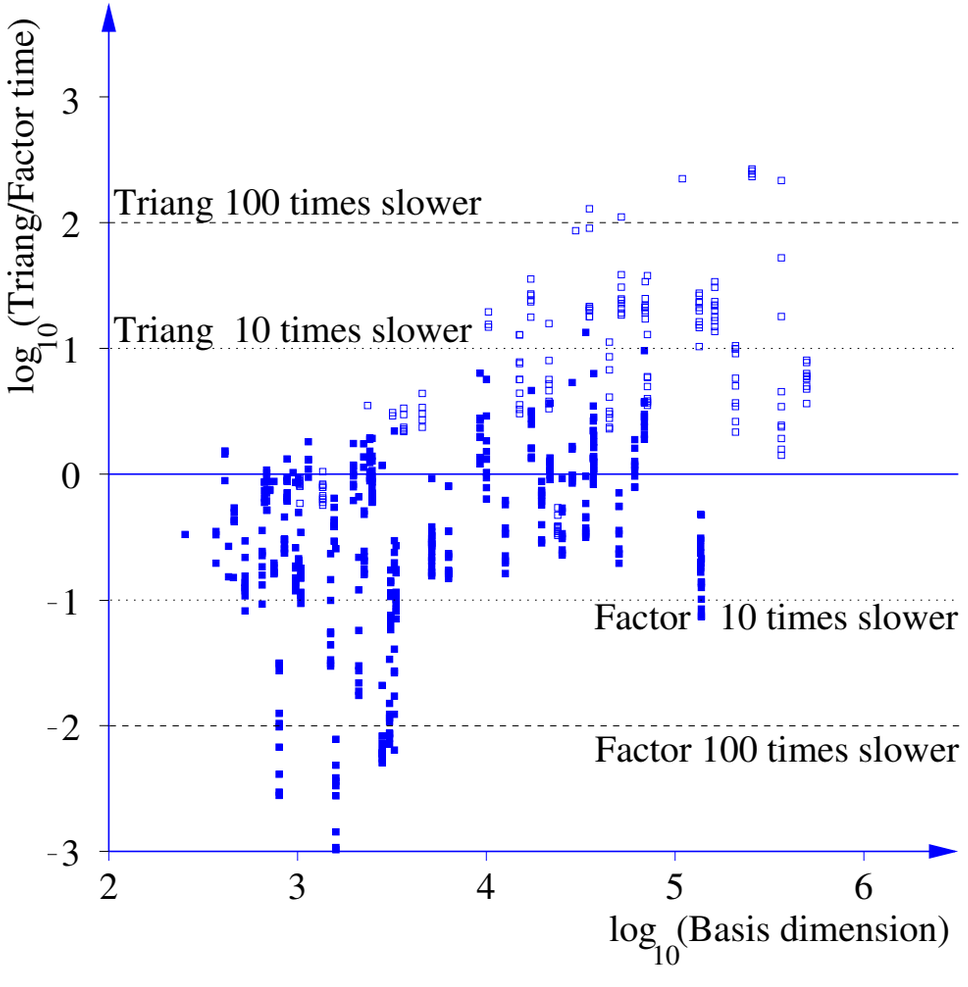
  Triangularisation:

  - Active row/column **singleton** entries are identified as pivots until all active rows/columns have at least *two* active nonzeros
  - Corresponds to Markowitz pivot selection so long as count is zero
  - Yields approximate Tarjan form at *much* lower cost

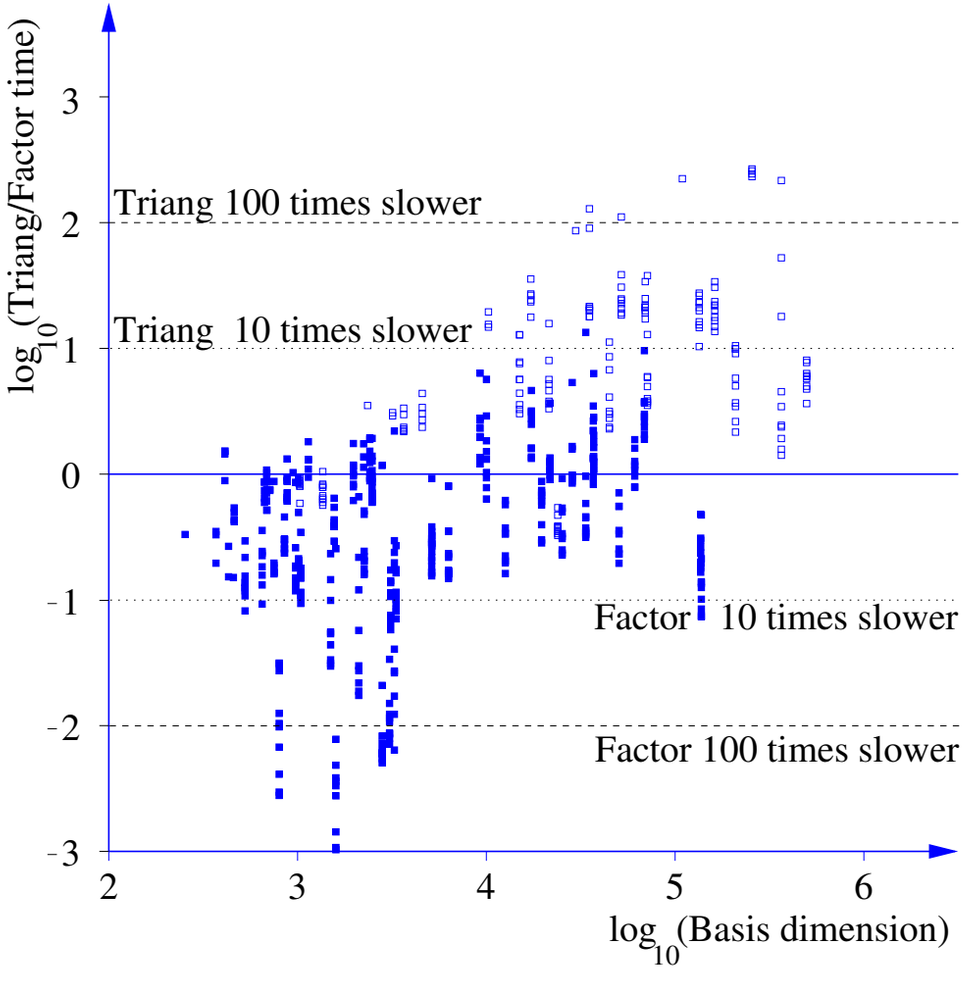
  Factorization:

  - Gaussian elimination applied the residual **bump**

Tomlin INVERT: relative time for triangularisation and factorization



Tomlin INVERT: relative time for triangularisation and factorization



- Triangularisation can be hugely dominant: particularly for large hyper-sparse LPs
- Bump factorization can be hugely dominant: particularly for smaller non hyper-sparse LPs

Parallel basis matrix inversion

For hyper-sparse LP problems

- Develop parallel triangularisation: Hall (2006-date)
  - Very encouraging results for serial simulation
  - Parallel implementation is in progress

Parallel basis matrix inversion

For hyper-sparse LP problems

- Develop parallel triangularisation: Hall (2006-date)
  - Very encouraging results for serial simulation
  - Parallel implementation is in progress

For other LP problems

- Perform parallel factorization of bump
  - Scope for collaboration with MUMPS group

Future of parallel simplex

- Pure data parallel simplex
  - For hyper-sparse LPs: will be hard but parallel INVERT is an important start
  - For other LP problems: good prospects with parallel INVERT

Future of parallel simplex

- Pure data parallel simplex
  - For hyper-sparse LPs: will be hard but parallel INVERT is an important start
  - For other LP problems: good prospects with parallel INVERT
- Task/data parallel simplex
  - Parallel INVERT will improve performance of SYNPLEX

Future of parallel simplex

- Pure data parallel simplex
  - For hyper-sparse LPs: will be hard but parallel INVERT is an important start
  - For other LP problems: good prospects with parallel INVERT
- Task/data parallel simplex
  - Parallel INVERT will improve performance of SYNPLEX
- Parallelisation of (Kaul's) simplex variant for block-angular LPs
  - Potential demonstrated by Boduroğlu (1997)
  - Boduroğlu-Hall collaboration starts June 2007

Future of parallel simplex

- Pure data parallel simplex
  - For hyper-sparse LPs: will be hard but parallel INVERT is an important start
  - For other LP problems: good prospects with parallel INVERT
- Task/data parallel simplex
  - Parallel INVERT will improve performance of SYNPLEX
- Parallelisation of (Kaul's) simplex variant for block-angular LPs
  - Potential demonstrated by Boduroğlu (1997)
  - Boduroğlu-Hall collaboration starts June 2007
- Parallelisation of *sparse* standard (tableau) simplex method
  - Attempted once by Lentini *et al.* (1995)
  - Would allow hyper-sparsity in very large LPs to be studied

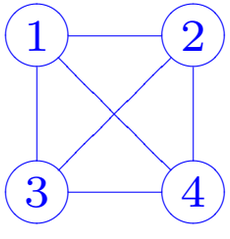
Interior point methods (IPM)

Interior point methods (IPM)

- IPM beat simplex for many problems
  - **Challenge:** maintain sparsity when factoring $A\Theta A^T$

Interior point methods (IPM)

- IPM beat simplex for many problems
  - **Challenge:** maintain sparsity when factoring $A\Theta A^T$
- **But:** $A\Theta A^T$ can fill in badly
  - Network LP problems are *very* sparse
  - Constraint (node-arc incidence) matrix for the graph

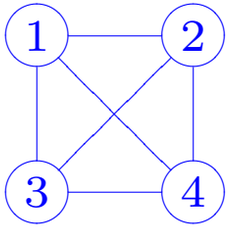


is $A = \begin{bmatrix} \bullet & & \bullet & \bullet & & \\ \bullet & \bullet & & & & \bullet \\ & \bullet & & \bullet & \bullet & \\ & & \bullet & & \bullet & \bullet \end{bmatrix}$ but $A\Theta A^T = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \end{bmatrix}$

  - Network(-like) LPs are hyper-sparse and simplex beats IPM

Interior point methods (IPM)

- IPM beat simplex for many problems
  - **Challenge:** maintain sparsity when factoring $A\Theta A^T$
- **But:** $A\Theta A^T$ can fill in badly
  - Network LP problems are *very* sparse
  - Constraint (node-arc incidence) matrix for the graph



is $A = \begin{bmatrix} \bullet & & \bullet & \bullet & \\ \bullet & \bullet & & & \bullet \\ & \bullet & & \bullet & \bullet \\ & & \bullet & & \bullet & \bullet \end{bmatrix}$ but $A\Theta A^T = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \end{bmatrix}$

  - Network(-like) LPs are hyper-sparse and simplex beats IPM

- **Alternative:** solve systems with $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$
  - **Challenge:** Factor K
  - **Challenge:** Design preconditioner for K and use conjugate gradients

Preconditioned CG for IPM

- $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$ is indefinite but PCG works with $\boldsymbol{x}^{(0)}$ in appropriate space

  Lukšan and Vlček (1998); Rozložník and Simoncini (2002)
- Applied to IPM by Al-Jeiroudi, Gondzio and Hall (2006)

Preconditioned CG for IPM

- $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$ is indefinite but PCG works with $\boldsymbol{x}^{(0)}$ in appropriate space

  Lukšan and Vlček (1998); Rozložník and Simoncini (2002)
- Applied to IPM by Al-Jeiroudi, Gondzio and Hall (2006)
  - Convergence of IPM yields simplex-like partition of $\Theta^{-1} = [\, {\Theta_B}^{-1} \quad {\Theta_N}^{-1} \,]$
  - Since ${\Theta_B}^{-1} \to 0$, partition A as $[\, B \quad N \,]$ and precondition K with

$$P = \begin{bmatrix} & & B^T \\ & -{\Theta_N}^{-1} & N^T \\ B & N & 0 \end{bmatrix}$$

Preconditioned CG for IPM

- $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$ is indefinite but PCG works with $\boldsymbol{x}^{(0)}$ in appropriate space

  Lukšan and Vlček (1998); Rozložník and Simoncini (2002)
- Applied to IPM by Al-Jeiroudi, Gondzio and Hall (2006)
  - Convergence of IPM yields simplex-like partition of $\Theta^{-1} = [\, {\Theta_B}^{-1} \quad {\Theta_N}^{-1} \,]$
  - Since ${\Theta_B}^{-1} \longrightarrow 0$, partition A as $[\, B \quad N \,]$ and precondition K with

$$P = \begin{bmatrix} & & B^T \\ & -{\Theta_N}^{-1} & N^T \\ B & N & 0 \end{bmatrix}$$

  - Exploit hyper-sparsity when identifying full-rank B
  - Use Tomlin INVERT to factor B

Preconditioned CG for IPM

- $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$ is indefinite but PCG works with $\boldsymbol{x}^{(0)}$ in appropriate space

  Lukšan and Vlček (1998); Rozložník and Simoncini (2002)
- Applied to IPM by Al-Jeiroudi, Gondzio and Hall (2006)
  - Convergence of IPM yields simplex-like partition of $\Theta^{-1} = [\, {\Theta_B}^{-1} \quad {\Theta_N}^{-1} \,]$
  - Since ${\Theta_B}^{-1} \longrightarrow 0$, partition A as $[\, B \quad N \,]$ and precondition K with

$$P = \begin{bmatrix} & & B^T \\ & -{\Theta_N}^{-1} & N^T \\ B & N & 0 \end{bmatrix}$$

  - Exploit hyper-sparsity when identifying full-rank B
  - Use Tomlin INVERT to factor B
- Spin-off application to matrix rank identification proposed by Hall and Saunders

Preconditioned CG for IPM

- $K = \begin{bmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{bmatrix}$ is indefinite but PCG works with $\boldsymbol{x}^{(0)}$ in appropriate space

  Lukšan and Vlček (1998); Rozložník and Simoncini (2002)
- Applied to IPM by Al-Jeiroudi, Gondzio and Hall (2006)
  - Convergence of IPM yields simplex-like partition of $\Theta^{-1} = [\, {\Theta_B}^{-1} \quad {\Theta_N}^{-1} \,]$
  - Since ${\Theta_B}^{-1} \to 0$, partition A as $[\, B \quad N \,]$ and precondition K with

$$P = \begin{bmatrix} & & B^T \\ & -{\Theta_N}^{-1} & N^T \\ B & N & 0 \end{bmatrix}$$

  - Exploit hyper-sparsity when identifying full-rank B
  - Use Tomlin INVERT to factor B
- Spin-off application to matrix rank identification proposed by Hall and Saunders
- Link with work on preconditioners for indefinite systems when solving elliptic PDEs

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Simplex method

- iterations are **cheap**
- terminates quickly given near-optimal partition $\mathcal{B} \cup \mathcal{N}$

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Simplex method

- iterations are **cheap**
- terminates quickly given near-optimal partition $\mathcal{B} \cup \mathcal{N}$

- **Challenge:** devise methods combining the best of both worlds

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Simplex method

- iterations are **cheap**
- terminates quickly given near-optimal partition $\mathcal{B} \cup \mathcal{N}$

- **Challenge:** devise methods combining the best of both worlds
- **Hard:** would revolutionise linear optimization

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Simplex method

- iterations are **cheap**
- terminates quickly given near-optimal partition $\mathcal{B} \cup \mathcal{N}$

- **Challenge:** devise methods combining the best of both worlds
- **Hard:** would revolutionise linear optimization
- Steps towards this "holy grail"
  - LP-DASA true hybrid method: Davis and Hager (2005)
  - Use simplex techniques in PCG for IPM (above)
  - Use simplex techniques in direct methods for IPM: Gondzio, Hall and Hogg (current)

Fusion of IPM and simplex techniques

Interior point methods

- require **few** iterations
- get estimate of optimal partition $\mathcal{B} \cup \mathcal{N}$ quickly

Simplex method

- iterations are **cheap**
- terminates quickly given near-optimal partition $\mathcal{B} \cup \mathcal{N}$

- **Challenge:** devise methods combining the best of both worlds
- **Hard:** would revolutionise linear optimization
- Steps towards this "holy grail"
  - LP-DASA true hybrid method: Davis and Hager (2005)
  - Use simplex techniques in PCG for IPM (above)
  - Use simplex techniques in direct methods for IPM: Gondzio, Hall and Hogg (current)
- Gondzio-Hall collaboration offers scope for further progress

Summary and conclusions

- Exploiting hyper-sparsity has had a major impact on simplex-IPM competition
- Parallelising simplex is hard without parallel INVERT
- Iterative solution of equations in IPM is fruitful
- Demonstrated scope for valuable future developments in important areas of computational linear optimization:
  - Parallel simplex
  - Fusion of IPM and simplex techniques
- Important advances will require collaboration between major groups

`http://www.maths.ed.ac.uk/hall/Talks`

Thank you

Summary and conclusions

- Exploiting hyper-sparsity has had a major impact on simplex-IPM competition
- Parallelising simplex is hard without parallel INVERT
- Iterative solution of equations in IPM is fruitful
- Demonstrated scope for valuable future developments in important areas of computational linear optimization:
  - Parallel simplex
  - Fusion of IPM and simplex techniques
- Important advances will require collaboration between major groups

`http://www.maths.ed.ac.uk/hall/Talks`

Thank you