A Scalable Interior Point Solver for Stochastic Programming

Mihai Anitescu

Mathematics and Computer Science Division
Argonne National Laboratory

INFORMS 2010
With Cosmin Petra, Victor Zavala, and Emil Constantinescu
MOTIVATION
Optimal management under uncertainty of energy systems

- Applications @Argonne – Anitescu, Constantinescu, Zavala
  - Stochastic Unit Commitment with Wind Power Generation
  - Energy management of Co-generation
  - Economic Optimization of a Building Energy System

- Sources of large dimensional uncertainty in complex energy systems
  - Weather
  - Consumer Demand
  - Market prices
Creating the weather uncertainty

• Use the Weather Research Forecast (WRF) Code as a Black-Box
• Hidden Markov Model (HMM) with an enormous spatial domain.
• Approximate the background correlations by the ergodic hypothesis, fit variance from data, propagate and “track” variance.

\[
Y_{[i,j]} := \chi_i(t_{\ell} + j) = \underbrace{\mathcal{M}(\mathcal{M}(...\mathcal{M}(\chi_i(t_{\ell}))))}_{j \text{ times}}
\]

\[
E[Y] \approx \bar{Y} := \frac{1}{NS} \sum_{i=1}^{NS} Y_{[i,:]}\]

\[
V \approx \frac{1}{NS - 1} \sum_{i=1}^{NS} (Y_{[i,:]} - \bar{Y})(Y_{[i,:]} - \bar{Y})^T
\]

Validation Results, Pittsburgh Area 2006

5 Day Forecast and +/- 3s Intervals

Building applications; August 1st-5th: hourly
Sampling Atmospheric Conditions

- **Focus:** Uncertainty quantification for large-scale problems
  - Extremely large scale hidden Markov model
  - (Spatio-temporal uncertainty)
  - Physics-based scalable approximate sampling algorithms

- **Application:** Weather and Climate Prediction with UQ
  - Aeolus: scalable *multiscale* UQ framework for weather prediction – wind power prediction
  - Uncertainty propagation approximated by ensembles; a multiscale approach makes the problem tractable.
  - Operational quality UQ is very computationally Intensive: wind/temperature **24 hours** [simulation time] in **one hour** [real time] with **30 members** needs **500 processors** on Argonne’s Jazz cluster.

\[\text{Wind speed at 80m, forecast with assimilated data, Midwest}\]
Stochastic Unit Commitment with Wind Power (SAA)

\[
\begin{align*}
\min \quad & \text{COST} = \frac{1}{N_s} \sum_{s \in S} \left( \sum_{j \in N} \sum_{k \in T} c^p_{sjk} + c^u_{jk} + c^d_{jk} \right) \\
\text{s.t.} \quad & \sum_{j \in N} p_{sjk} + \sum_{j \in N_{\text{wind}}} p^{\text{wind}}_{sjk} = D_k, s \in S, k \in T \\
& \sum_{j \in N} \overline{p}_{sjk} + \sum_{j \in N_{\text{wind}}} p^{\text{wind}}_{sjk} \geq D_k + R_k, s \in S, k \in T \\
& \text{ramping constr., min. up/down constr.}
\end{align*}
\]

- Wind Forecast – WRF(Weather Research and Forecasting) Model
  - Real-time grid-nested 24h simulation
  - 30 samples require 1h on 500 CPUs (Jazz@Argonne)
Wind power forecast and stochastic programming

- Unit commitment & energy dispatch with uncertain wind power generation for the State of Illinois, assuming 20% wind power penetration, using the same windfarm sites as the one existing today.

- Full integration with 10 thermal units to meet demands. Consider dynamics of start-up, shutdown, set-point changes

- Does uncertainty matter? ... Yes. The solution is only 1% more expensive than the one with exact information. Solution on average infeasible at 10%.
Management under uncertainty paradigm: stochastic programming.

- Two-stage stochastic programming with recourse ("here-and-now")
  \[
  \min_{x_0} \left\{ f_0(x_0) + \mathbb{E} \left[ \min_x f(x, \omega) \right] \right\}
  \]
  subj. to. \[
  A_0 x_0 = b_0 \\
  A(\omega)x_0 + B(\omega)x = b(\omega) \\
  x_0 \geq 0, \quad x(\omega) \geq 0
  \]

- \( \xi(\omega) := (A(\omega), B(\omega), b(\omega), Q(\omega), c(\omega)) \)

Sample average approximation (SAA)

\[
\min_{x_0, x_1, x_2, \ldots, x_s} \left\{ f_0(x) + \frac{1}{S} \sum_{i=1}^{s} f_i(x_i) \right\}
\]
subj. to. \[
A_0 x_0 = b_0 \\
A_k x_0 + B_k x_k = b_k, \\
x_0 \geq 0, \quad x_k \geq 0, \quad k = 1, \ldots, S
\]
INTERIOR POINT METHODS FOR STOCHASTIC PROGRAMMING
Linear Algebra of Primal-Dual Interior-Point Methods

Convex quadratic problem
\[ \begin{align*}
\text{Min} & \quad \frac{1}{2} x^T Q x + c^T x \\
\text{subj. to} & \quad A x = b \\
x & \geq 0
\end{align*} \]

IPM Linear System
\[
\begin{bmatrix}
Q + \Lambda & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix}
= rhs
\]

Multi-stage SP

Two-stage SP

*nested* arrow-shaped linear system (via a permutation)

Mihai Anitescu -- Stochastic Programming
The Direct Schur Complement Method (DSC)

- Uses the arrow shape of $H$

$$
\begin{bmatrix}
H_1 & G_1^T \\
H_2 & G_2^T \\
\vdots & \vdots \\
G_1 & G_2 & \ldots & G_S & H_0 \\
\end{bmatrix}
= \begin{bmatrix}
L_1 & \ & \ & \ & \ \\
\ & L_2 & \ & \ & \ \\
\ & \ & \ddots & \ & \ \\
\ & \ & \ & L_{i_0} & L_{i_1} & \ldots & L_{i_S} & L_e \\
\end{bmatrix}
\begin{bmatrix}
D_1 & \ & \ & \ & \ \\
\ & D_2 & \ & \ & \ \\
\ & \ & \ddots & \ & \ \\
\ & \ & \ & D_N & D_e \\
\end{bmatrix}
\begin{bmatrix}
L_1^T & \ & \ & \ & \ \\
\ & L_2^T & \ & \ & \ \\
\ & \ & \ddots & \ & \ \\
\ & \ & \ & L_{i_0}^T & L_{i_1}^T & \ldots & L_{i_S}^T & L_e^T \\
\end{bmatrix}
$$

- Solving $Hz=r$

$$L_iD_iL_i^T = H_i, \quad L_{i_0} = G_iL_i^{-T}D_i^{-1}, \quad i = 1, \ldots, S,$$

$$C = H_0 - \sum_{i=1}^{S} G_iH_i^{-1}G_i^T,$$

$$L_cD_cL_c^T = C.$$
High performance computing with DSC

- Gondzio (OOPS) 6-stages 1 billion variables; Zavala et.al., 2007 (in IPOPT)
- Our experiments (PIPS) – strong scaling is investigated;
- We do only two stage but we have much larger physical layer.
  - Building energy system
    - Almost linear scaling
  - Unit commitment
    - Relaxation solved
    - Largest instance
  - 28.9 millions variables
  - 1000 cores
Scalability of DSC

Unit commitment
76.7% efficiency ...

...but not always the case, since first stage calculations can keep everyone blocked. Issue: Size of first stage.

Large number of 1st stage variables: 38.6% efficiency
BOTTLENECK SOLUTION 1: STOCHASTIC PRECONDITIONER
Preconditioned Schur Complement (PSC)

\[ L_i D_i L_i^T = H_i, \quad L_{i0} = G_i L_i^{-T} D_i^{-1}, \quad i = 1, \ldots, N, \]
\[ C = H_0 - \sum_{i=1}^{N} G_i H_i^{-1} G_i^T \]
\[ L_M D_M L_M^T = M \]
(separate process)

\[ w_i = L_i^{-1} r_i \]
\[ \tilde{r}_0 = \left( r_0 - \sum_{i=1}^{N} L_{Ni} w_i \right) \]
\[ v_i = D_i^{-1} w_i, \quad i = 0, \ldots, N \]
\[ z_i = L_i^{-T} \left( v_i - L_{i0} z_0 \right) \]

\[ z_0 = \text{Krylov}(C, M, \tilde{r}_0) \]
The Stochastic Preconditioner

- The exact structure of $C$ is

$$C = \begin{bmatrix} S_S \\ \tilde{Q}_0 + \frac{1}{S} \sum_{i=1}^{S} A_i^T \left( B_i \tilde{Q}_i^{-1} B_i^T \right)^{-1} A_i \end{bmatrix} A_0^T.$$

- IID subset of $n$ scenarios: $\mathcal{K} = \{k_1, k_2, \ldots, k_n\}$

- The stochastic preconditioner (Petra & Anitescu, 2010)

$$S_n = \tilde{Q}_0 + \frac{1}{n} \sum_{i=1}^{n} A_{k_i}^T \left( B_{k_i} \tilde{Q}_{k_i}^{-1} B_{k_i}^T \right)^{-1} A_{k_i}.$$  

- For $C$ use the constraint preconditioner (Keller et. al., 2000)

$$M = \begin{bmatrix} S_n & A_0^T \\ A_0 & 0 \end{bmatrix}.$$
Quality of the Stochastic Preconditioner

\[ S_n = \tilde{Q}_0 + \frac{1}{n} \sum_{i=1}^{n} \left[ A_{k_i}^T \left( B_{k_i} \tilde{Q}_{k_i}^{-1} B_{k_i}^T \right)^{-1} A_{k_i} \right] \]

\[ S_S = \tilde{Q}_0 + \frac{1}{S} \sum_{i=1}^{S} \left[ A_i^T \left( B_i \tilde{Q}_i^{-1} B_i^T \right)^{-1} A_i \right] \]

- “Exponentially” better preconditioning \cite{Petra2010} (Petra & Anitescu 2010)
  \[
  \Pr(|\lambda(S_n^{-1}S_S)^{-1} - 1| \geq \varepsilon) \leq 2p^4 \exp\left(-\frac{n\varepsilon^2}{2p^4L^2 \| S_S \|_{\text{max}}^2}\right)
  \]

- Proof: Hoeffding inequality (p is dim on S; L is a bound on data)

- Assumptions on the problem’s random data
  1. Boundedness
  2. Uniform full rank of \( A(\omega) \) and \( B(\omega) \)

\[ \text{not restrictive (=> L)} \]
Quality of the Constraint Preconditioner

\[
M = \begin{bmatrix}
S_n & A_0^T \\
A_0 & 0 \\
\end{bmatrix}
\]
\[
C = \begin{bmatrix}
S_S & A_0^T \\
A_0 & 0 \\
\end{bmatrix}
\]

- $M^{-1}C$ has an eigenvalue 1 with order of multiplicity $2r$.
- The rest of the eigenvalues satisfy
  \[
  0 < \lambda_{\min}(S_n^{-1}S_S) \leq \lambda(M^{-1}C) \leq \lambda_{\max}(S_n^{-1}S_S).
  \]
- Proof: based on Bergamaschi et al., 2004.
The Krylov Methods Used for $Cz_0 = r_0$

\[
\begin{bmatrix}
S_S & A_0^T \\
A_0 & 0
\end{bmatrix}
\begin{bmatrix}
x_0 \\
y_0
\end{bmatrix} =
\begin{bmatrix}
r_0^1 \\
r_0^2
\end{bmatrix}
\]

- **BiCGStab** using constraint preconditioner $M$

- **Preconditioned Projected CG (PPCG)** (Gould *et al.*, 2001)
  - Preconditioned projection onto the $\text{Ker}A_0$.
    \[
P = Z_0 \left( Z_0^T S_n Z_0 \right)^{-1} Z_0^T
    \]
  - Does not compute the basis $Z_0$ for $\text{Ker}A_0$. Instead,
    \[
g = \text{Pr} \text{ is computed from } \begin{bmatrix}
S_n & A_0^T \\
A_0 & 0
\end{bmatrix}
\begin{bmatrix}
g \\
u
\end{bmatrix} =
\begin{bmatrix}
r \\
0
\end{bmatrix}.
    \]
  - $y_0 = (A_0 A_0^T)^{-1} A_0 \left( r_0^1 - S_N x_0 \right)$
Performance of the preconditioner

- Eigenvalues clustering & Krylov iterations

- Affected by the well-known ill-conditioning of IPMs.

\[ S_n \approx S_N \quad \text{and} \quad S_N \approx \mathbb{E}[S(\omega)], \quad \text{where} \]

\[ S(\omega) = (Q_0 + D_0) + \left[ A^T(\omega) \left( B(\omega)(Q(\omega) + D(\omega))^{-1} B^T(\omega) \right)^{-1} A(\omega) \right] \]
The “Ugly” Unit Commitment Problem; PSC gets further

- DSC on P processes vs PSC on P+1 process

Optimal use of PSC – linear scaling

- 120 scenarios - # cores used for preconditioner

- Conclusion: PSC hides the latency well, but it eventually hits a memory wall as well.

Factorization of the preconditioner can not be hidden anymore; we need to accelerate it as well; cannot solve larger problems where improvement would likely be larger.
SOLUTION 2: PARALLELIZATION OF STAGE 1 LINEAR ALGEBRA
Parallelizing the 1st stage linear algebra

- We distribute the 1st stage Schur complement system.
  \[
  C = \begin{bmatrix}
  \tilde{Q} & A_0^T \\
  A_0 & 0
  \end{bmatrix}, \quad \tilde{Q} \text{ dense symm. pos. def., } A_0 \text{ sparse full rank.}
  \]
- C is treated as dense.
- Alternative to PSC for problems with large number of 1st stage variables.
- Removes the memory bottleneck of PSC and DSC.
- We investigated ScaLapack, Elemental (successor of PLAPACK)
  - None have a solver for symmetric indefinite matrices (Bunch-Kaufman);
  - LU or Cholesky only.
  - So we had to think of modifying either.
Cholesky-based $LDL^T$-like factorization

\[
\begin{bmatrix}
\tilde{Q} & A^T \\
A & 0
\end{bmatrix} = \begin{bmatrix}
L & 0 \\
AL^{-T} & L
\end{bmatrix}
\begin{bmatrix}
I \\
-I
\end{bmatrix}
\begin{bmatrix}
L^T & L^{-1}A^T \\
0 & \tilde{L}^T
\end{bmatrix}, \text{ where } LL^T = \tilde{Q}, \quad \tilde{L}L^T = A\tilde{Q}^{-1}A^T
\]

- Can be viewed as an “implicit” normal equations approach.
- In-place implementation inside Elemental: no extra memory needed.
- Idea: modify the Cholesky factorization, by changing the sign after processing $p$ columns.
- It is much easier to do in Elemental, since this distributes elements, not blocks.
- Twice as fast as LU
- Works for more general saddle-point linear systems, i.e., pos. semi-def. (2,2) block.
Distributing the 1st stage Schur complement matrix

- All processors contribute to all of the elements of the (1,1) dense block

\[
\tilde{Q} = \tilde{Q}_0 + \frac{1}{S} \sum_{i=1}^{S} A_i^T \left( B_i \tilde{Q}_i^{-1} B_i^T \right)^{-1} A_i 
\]

- A large amount of inter-process communication occurs.

- Possibly more costly than the factorization itself.

- Solution: use buffer to reduce the number of messages when doing a Reduce_scatter.

- \( LDL^T \) approach also reduces the communication by half – only need to send lower triangle.
Large-scale performance

- Comparison of ScaLapack (LU), Elemental(LU), and $LDL^T$ (1024 cores)

<table>
<thead>
<tr>
<th>Units (Q+A)</th>
<th>1st Stage Size</th>
<th>Factor (Sec.)</th>
<th>Reduce (Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$LU(S)$ $LU(E)$ $LDL^T$</td>
<td>$LU$ $LDL^T$</td>
</tr>
<tr>
<td>300</td>
<td>23436+1224</td>
<td>16.59 20.04 6.71</td>
<td>54.32 26.35</td>
</tr>
<tr>
<td>640</td>
<td>49956+2584</td>
<td>60.67 83.24 36.77</td>
<td>256.95 128.59</td>
</tr>
<tr>
<td>1000</td>
<td>78030+4024</td>
<td>173.67 263.53 90.82</td>
<td>565.36 248.22</td>
</tr>
</tbody>
</table>

- Strong scaling
  - 90.1% from 64 to 1024 cores;
  - 75.4% from 64 to 2048 cores.
  - > 4,000 scenarios.

SAA problem: 189 million variables
A Parallel Interior-Point Solver for Stochastic Programming (PIPS)

- Convex QP SAA SP problems
- Input: users specify the scenario tree
- Object-oriented design based on OOQP
- Linear algebra: tree vectors, tree matrices, tree linear systems
- Scenario based parallelism
  - tree nodes (scenarios) are distributed across processors
  - inter-process communication based on MPI
  - dynamic load balancing
- Mehrotra predictor-corrector IPM
Conclusions

- The DSC method offers a good parallelism for SP in an IPM framework.
- The PSC method improves the scalability, by tackling the latency, though not the memory wall.
- Parallel direct linear algebra eliminates the memory wall; we will investigate their combination to reduce both.
- PIPS – solver for SP problems.
- PIPS is ready for larger problems: 100,000 cores.
Future work

- New math / stat
  - Asynchronous optimization
  - SAA error estimate

- New scalable methods for a more efficient software
  - Better interconnect between (iterative) linear algebra and sampling
    - importance-based preconditioning
    - multigrid decomposition
  - Target: emerging exa architectures

- PIPS
  - IPM hot-start, parallelization of the nodes
  - Ensure compatibility with other paradigms: NLP, conic progr., MILP/MINLP solvers
  - A ton of other small enhancements

- Ensure computing needs for important applications
  - Unit commitment with transmission constraints & market integration (Zavala)
References


Thank you for your attention!

Questions?
**Multi-stage SAA SP Problems – Scenario formulation**

- Depth-first traversal of the scenario tree
- **Nested** half-arrow shaped Jacobian
- Block separable obj. func.

\[
\begin{align*}
\text{Min} & \quad \frac{1}{2} \sum_{i=0}^{7} x_i^T Q_i x_i + \sum_{i=0}^{7} c_i^T x_i \\
\text{s.t.} & \quad A_0 x_0 + B_1 x_1 \\
& \quad A_1 x_1 + B_2 x_2 \\
& \quad A_2 x_1 + B_3 x_3 \\
& \quad A_3 x_1 + B_4 x_4 \\
& \quad A_4 x_0 + B_5 x_5 \\
& \quad A_5 x_4 + B_6 x_6 \\
& \quad A_6 x_4 + B_7 x_7 \\
\end{align*}
\]

\[
\begin{align*}
x_0 & \geq 0 \\
x_1 & \geq 0 \\
x_2 & \geq 0 \\
x_3 & \geq 0 \\
x_4 & \geq 0 \\
x_5 & \geq 0 \\
x_6 & \geq 0 \\
x_7 & \geq 0 \\
\end{align*}
\]

\[
\begin{align*}
b_0 &= 0 \\
b_1 &= 0 \\
b_2 &= 0 \\
b_3 &= 0 \\
b_4 &= 0 \\
b_5 &= 0 \\
b_6 &= 0 \\
b_7 &= 0 \\
\end{align*}
\]

Mihai Anitescu -- Stochastic Programming
Tree Linear Algebra – Data, Operations & Linear Systems

- **Data**
  - Tree vector: b, c, x, etc
  - Tree symmetric matrix: Q
  - Tree general matrix: A

- **Operations**
  \[(u \otimes v)_n = u_n \otimes v_n, \forall n \in T\]
  \[(Qx)_n = Q_n x_n, \forall n \in T\]
  \[(Ax)_n = A_{a(n)} x_{a(n)} + B_n x_n, \forall n \in T^*\]
  \[(A^T y)_n = B_n^T y_n + \sum_{c \in C(n)} A_c^T y_c\]

- **Linear systems:** for each non-leaf node a two-stage problem is solved via Schur complement methods as previously described.

\[\text{Min } \frac{1}{2} x^T Q x + c^T x\]
subj. to. \[Ax = b\]
\[x \geq 0\]
Parallelization – Tree Distribution

- The tree is distributed across processes.
- Example: 3 processes

- Dynamic load balancing of the tree
  - Number partitioning problem --> graph partitioning --> METIS