Comput Optim Appl (2016) 64:379–406
DOI 10.1007/s10589-015-9813-x

Clustering-based preconditioning for stochastic programs

Yankai Cao[1] · Carl D. Laird[1] · Victor M. Zavala[2]

Received: 16 December 2014 / Published online: 21 December 2015

Abstract We present a clustering-based preconditioning strategy for KKT systems arising in stochastic programming within an interior-point framework. The key idea is to perform adaptive clustering of scenarios (inside-the-solver) based on their influence on the problem at hand. This approach thus contrasts with existing (outside-the-solver) approaches that cluster scenarios based on problem data alone. We derive spectral and error properties for the preconditioner and demonstrate that scenario compression rates of up to 94 % can be obtained, leading to dramatic computational savings. In addition, we demonstrate that the proposed preconditioner can avoid scalability issues of Schur decomposition in problems with large first-stage dimensionality.

Keywords Preconditioning · Interior-point · Stochastic · Large-scale · Clustering

✉ Victor M. Zavala
victor.zavala@wisc.edu

Yankai Cao
cao142@purdue.edu

Carl D. Laird
lairdc@purdue.edu

[1] School of Chemical Engineering, Purdue University, 480 Stadium Mall Drive, West Lafayette, IN 47907, USA

[2] Department of Chemical and Biological Engineering, University of Wisconsin-Madison, 1415 Engineering Hall, Madison, WI 53706-1691, USA

1 Preliminaries

We consider two-stage stochastic programs of the form

$$\min \left(\frac{1}{2}x_0^T Q_0 x_0 + d_0^T x_0\right) + \sum_{s\in\mathcal{S}} p_s \left(\frac{1}{2}x_s^T Q_s x_s + d_s^T x_s\right) \tag{1a}$$

$$\text{s.t.} \qquad W_0 x_0 = b_0, \quad (y_0) \tag{1b}$$

$$T_s x_0 + W_s x_s = b_s, \;\; (y_s), \;\; s \in \mathcal{S} \tag{1c}$$

$$x_0 \geq 0, \qquad (\nu_0) \tag{1d}$$

$$x_s \geq 0, \qquad (\nu_s), \;\; s \in \mathcal{S}. \tag{1e}$$

Here, $\mathcal{S} := \{1 \ldots S\}$ is the scenario set and S is the number of scenarios. The problem variables are $x_0, \nu_0 \in \Re^{n_0}$, $x_s, \nu_s \in \Re^{n_s}$, $y_0 \in \Re^{m_0}$, and $y_s \in \Re^{m_s}$. The total number of variables is $n := n_0 + \sum_{s\in\mathcal{S}} n_s$, of equality constraints is $m := m_0 + \sum_{s\in\mathcal{S}} m_s$, and of inequalities is n. We refer to (x_0, y_0, ν_0) as the first-stage variables and to (x_s, y_s, ν_s), $s \in \mathcal{S}$ as the second-stage variables. We refer to Eq. (1a) as the cost function. The *data* defining problem (1) is given by the cost coefficients d_0, Q_0, Q_s, d_s; the right-hand side coefficients b_0, b_s; the matrix coefficients T_s, W_s; and the scenario probabilities as $p_s \in \Re_+$. We refer to $p_s, Q_s, d_s, b_s, T_s, W_s$ as the *scenario data*.

As is typical in stochastic programming, the number of scenarios can be large and limits the scope of existing off-the-shelf solvers. In this work, we present strategies that cluster scenarios at the linear algebra level to mitigate complexity. We begin by presenting some basic notation. We scale the cost coefficient matrices and vectors as $Q_s \leftarrow p_s Q_s$ and $d_s \leftarrow p_s d_s$. Consequently, the Lagrange function of (1) can be expressed as:

$$\begin{aligned}\mathcal{L}(x, y, \nu) = &\frac{1}{2}x_0^T Q_0 x_0 + d_0^T x_0 + y_0^T(W_0 x_0 - b_0) - \nu_0^T x_0\\ &+ \sum_{s\in\mathcal{S}} \left(\frac{1}{2}x_s^T Q_s x_s + d_s^T x_s + y_s^T(T_s x_0 + W_s x_s - b_s) - \nu_s^T x_s\right).\end{aligned} \tag{2}$$

Here, $x := [x_0^T, x_1^T, \ldots, x_S^T]$, $y^T := [y_0^T, y_1^T, \ldots, y_S^T]$, and $\nu^T := [\nu_0^T, \nu_1^T, \ldots, \nu_S^T]$. In a primal-dual interior-point (IP) setting we seek to solve nonlinear systems of equations of the form

$$\nabla_{x_0}\mathcal{L} = 0 = Q_0 x_0 + d_0 + W_0^T y_0 - \nu_0 + \sum_{s\in\mathcal{S}} T_s^T y_s \tag{3a}$$

$$\nabla_{x_s}\mathcal{L} = 0 = Q_s x_s + d_s + W_s^T y_s - \nu_s, \;\; s \in \mathcal{S} \tag{3b}$$

$$\nabla_{y_0}\mathcal{L} = 0 = W_0 x_0 - b_0 \tag{3c}$$

$$\nabla_{y_s}\mathcal{L} = 0 = T_s x_0 + W_s x_s - b_s, \;\; s \in \mathcal{S} \tag{3d}$$

$$0 = X_0 V_0 e_{n_0} - \mu e_{n_0} \tag{3e}$$

$$0 = X_s V_s e_S - \mu e_{n_s}, \;\; s \in \mathcal{S}, \tag{3f}$$

with the implicit condition $x_0, \nu_0, x_s, \nu_s \geq 0$. Here, $\mu \geq 0$ is the barrier parameter and $e_{n_0} \in \Re^{n_0}, e_{n_s} \in \Re^{n_s}$ are vectors of ones. We define the diagonal matrices $X_0 := \text{diag}(x_0)$, $X_s := \text{diag}(x_s)$, $V_0 := \text{diag}(\nu_0)$, and $V_s := \text{diag}(\nu_s)$. We also define $\alpha_0 := X_0 V_0 e - \mu e_{n_0}$ and $\alpha_s := X_s V_s e - \mu e_{n_s},\ s \in \mathcal{S}$. The search step is obtained by solving the linear system

$$\begin{aligned}
Q_0 \Delta x_0 + W_0^T \Delta y_0 + \sum_{s \in \mathcal{S}} T_s^T \Delta y_s - \Delta \nu_0 &= -\nabla_{x_0} \mathcal{L} && \text{(4a)}\\
Q_s \Delta x_s + W_s^T \Delta y_s - \Delta \nu_s &= -\nabla_{x_s} \mathcal{L},\ s \in \mathcal{S} && \text{(4b)}\\
W_0 \Delta x_0 &= -\nabla_{y_0} \mathcal{L} && \text{(4c)}\\
T_s \Delta x_0 + W_s \Delta x_s &= -\nabla_{y_s} \mathcal{L},\ s \in \mathcal{S} && \text{(4d)}\\
X_0 \Delta \nu_0 + V_0 \Delta x_0 &= -\alpha_0 && \text{(4e)}\\
X_s \Delta \nu_s + V_s \Delta x_s &= -\alpha_s,\ s \in \mathcal{S}. && \text{(4f)}
\end{aligned}$$

After eliminating the step for the bound multipliers from the linear system we obtain

$$\begin{aligned}
H_0 \Delta x_0 + W_0^T \Delta y_0 + \sum_{s \in \mathcal{S}} T_s^T \Delta y_s &= r_{x_0} && \text{(5a)}\\
H_s \Delta x_s + W_s^T \Delta y_s &= r_{x_s},\ s \in \mathcal{S} && \text{(5b)}\\
W_0 \Delta x_0 &= r_{y_0} && \text{(5c)}\\
T_s \Delta x_0 + W_s \Delta x_s &= r_{y_s},\ s \in \mathcal{S}, && \text{(5d)}
\end{aligned}$$

where

$$\begin{aligned}
H_0 &:= Q_0 + X_0^{-1} V_0 && \text{(6a)}\\
H_s &:= Q_s + X_s^{-1} V_s,\ s \in \mathcal{S}. && \text{(6b)}
\end{aligned}$$

We also have that $r_{x_0} := -(\nabla_{x_0} \mathcal{L} + X_0^{-1} \alpha_0)$, $r_{x_s} := -(\nabla_{x_s} \mathcal{L}_s + X_s^{-1} \alpha_s)$, $r_{y_0} := -\nabla_{y_0} \mathcal{L}$, and $r_{y_s} := -\nabla_{y_s} \mathcal{L}$. The step for the bound multipliers can be recovered from

$$\begin{aligned}
\Delta \nu_0 &= -X_0^{-1} V_0 \Delta x_0 - X_0^{-1} \alpha_0 && \text{(7a)}\\
\Delta \nu_s &= -X_s^{-1} V_s \Delta x_s - X_s^{-1} \alpha_s,\ s \in \mathcal{S}. && \text{(7b)}
\end{aligned}$$

System (5) has the arrowhead form

$$\begin{bmatrix} K_1 & & & & B_1 \\ & K_2 & & & B_2 \\ & & \ddots & & \vdots \\ & & & K_S & B_S \\ B_1^T & B_2^T & \dots & B_S^T & K_0 \end{bmatrix} \begin{bmatrix} \Delta w_1 \\ \Delta w_2 \\ \vdots \\ \Delta w_S \\ \Delta w_0 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_S \\ r_0 \end{bmatrix}, \qquad (8)$$

where $\Delta w_0^T := [\Delta x_0^T, \Delta y_0^T]$, $\Delta w_s^T := [\Delta x_s^T, \Delta y_s^T]$, $r_0^T := [-r_{x_0}^T, -r_{y_0}^T]$, $r_s^T := [-r_{x_s}^T, -r_{y_s}^T]$, and

$$K_0 := \begin{bmatrix} H_0 & W_0^T \\ W_0 & 0 \end{bmatrix}, \; K_s := \begin{bmatrix} H_s & W_s^T \\ W_s & 0 \end{bmatrix}, \; B_s := \begin{bmatrix} 0 & 0 \\ T_s & 0 \end{bmatrix}. \tag{9}$$

We refer to the linear system (8) as the *KKT system* and to its coefficient matrix as the *KKT matrix*. We assume that each scenario block matrix $K_s,\; s \in \mathcal{S}$ is nonsingular.

We use the following notation to define a block-diagonal matrix M composed of blocks $M_1, M_2, M_3, \ldots$:

$$M = \text{blkdiag}\{M_1, M_2, M_3, \ldots\}. \tag{10}$$

In addition, we use the following notation to define a matrix B that stacks (row-wise) the blocks $B_1, B_2, B_3 \ldots$:

$$B = \text{rowstack}\{B_1, B_2, B_3, \ldots\}. \tag{11}$$

We apply the same rowstack notation for vectors. We use the notation $v^{(k)}$ to indicate the k-th entry of vector v. We use $\text{vec}(M)$ to denote the row-column vectorization of matrix M and we define $\sigma_{min}(M)$ as the smallest singular value of matrix M. We use $\|\cdot\|$ to denote the Euclidean norm for vectors and the Frobenius norm for matrices, and we recall that $\|M\| = \|\text{vec}(M)\|$ for matrix M.

2 Clustering setting

In this section we review work on scenario reduction and highlight differences and contributions of our work. We then present our clustering-based preconditioner for the KKT system (8).

2.1 Related work and contributions

Scenario clustering (also referred to as aggregation) is a strategy commonly used in stochastic programming to reduce computational complexity. We can classify these strategies as outside-the-solver and inside-the-solver strategies. Outside-the-solver strategies perform clustering on the scenario data (right-hand sides, matrices, and gradients) prior to the solution of the problem [5,8,13,16]. This approach can provide lower bounds and error bounds for linear programs (LPs) and this feature can be exploited in branch-and-bound procedures [1,5,22,28].

Outside-the-solver clustering approaches give rise to several inefficiencies. First, several optimization problems might need to be solved in order to refine the solution. Second, these approaches focus on the problem data and thus *do not capture the effect of the data on the particular problem at hand*. Consider, for instance, the situation in which the same scenario data (e.g., weather scenarios) is used for two different problem classes (e.g., farm planning and power grid planning). Moreover, clustering scenarios based on data alone is inefficient because scenarios that are close in terms of data might have very different impact on the cost function (e.g., if they are close to the constraint boundary). Conversely, two scenarios that are distant in terms of data might have similar contributions to the cost function. We also highlight that many scenario generation procedures require knowledge of the underlying probability distributions [10,13] which are often not available in closed form (e.g., weather forecasting) [18,26].

In this work, we seek to overcome these inefficiencies by performing clustering adaptively inside-the-solver. In an interior-point setting this can be done by creating a preconditioner for the KKT system (8) by *clustering* the scenario blocks. A key advantage of this approach is that a single optimization problem is solved and the clusters are refined only if the preconditioner is not sufficiently accurate. In addition, this approach provides a mechanism to capture the influence of the data on the particular problem at hand. Another advantage is that it can enable *sparse preconditioning* of Schur complement systems. This is beneficial in situations where the number of first-stage variables is large and thus Schur complement decomposition is expensive. Moreover, our approach does not require any knowledge of the underlying probability distributions generating the scenario data. Consequently, it can be applied to problems in which simulators are used to generate scenarios (e.g., weather forecasting), and it can be applied to problem classes that exhibit similar structures such as support vector machines [11,14] and scenario-based robust optimization [4]. Our proposed clustering approach can also be used in combination with outside-the-solver scenario aggregation procedures, if desired.

Related work on inside-the-solver scenario reduction strategies includes stochastic Newton methods [3]. These approaches sample scenarios to create a smaller representation of the KKT system. Existing approaches, however, cannot handle constraints. Scenario and constraint reduction approaches for IP solvers have been presented in [6,7,20,24]. In [24], scenarios that have little influence on the step computation are eliminated from the optimality system. This influence is measured in terms of the magnitude of the constraint multipliers or in terms of the products $X_s^{-1}V_s$. In that work, it was found that a large proportion of scenarios or constraints can be eliminated without compromising convergence. The elimination potential can be limited in early iterations, however, because it is not clear which scenarios have strong or weak influence on the solution. In addition, this approach eliminates the scenarios from the problem formulation, and thus special safeguards are needed to guarantee convergence. Our proposed clustering approach does not eliminate the scenarios from the problem formulation; instead, the scenario space is compressed to construct preconditioners.

In [20] preconditioners for Schur systems are constructed by sampling the full scenario set. A shortcoming of this approach is that scenario outliers with strong influence might not be captured in the preconditioner. This behavior is handled more efficiently in the preconditioner proposed in [6] in which scenarios having strong influence on the Schur complement are retained and those that have weak influence are eliminated. A limitation of the Schur preconditioners proposed in [6,20] is that they require a dense preconditioner for the Schur complement, which hinders scalability in problems with many first-stage variables. Our preconditioning approach enables sparse preconditioning and thus avoids forming and factorizing dense Schur complements. In addition, compared with approaches in [6,20,24], our approach clusters scenarios instead of eliminating them (either by sampling or by measuring strong/weak influence). This enables us to handle scenario redundancies and outliers. In [7], scenarios are clustered to solve a reduced problem and the solution of this problem is used to warm-start the problem defined for the full scenario set. The approach can reduce the number of iterations of the full scenario problem; but the work per iteration is not reduced, as in our approach.

2.2 Clustering-based preconditioner

To derive our clustering-based preconditioner, we partition the full scenario set $\mathcal{S}$ into C clusters, where $C \leq S$. We define the cluster set $\mathcal{C} := \{1, \dots, C\}$ and a partition of the scenario set $\{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_C\}$ with $\omega_i := |\mathcal{S}_i|$, $i \in \mathcal{C}$. That is,

$$\bigcup_{i \in \mathcal{C}} \mathcal{S}_i = \mathcal{S} \tag{12a}$$

$$\mathcal{S}_i \bigcap \mathcal{S}_j = \emptyset, \; i, j \in \mathcal{C}, j \neq i. \tag{12b}$$

For each cluster $i \in \mathcal{C}$, we pick an index $c_i \in \mathcal{S}_i$ to represent the cluster and we use these indexes to define the compressed set $\mathcal{R} := \{c_1, c_2, \dots, c_C\}$. We note that $|\mathcal{R}| = C$ and that ω_i are cluster weights. We define the binary indicator $\kappa_{s,i}$, $s \in \mathcal{S}, i \in \mathcal{C}$, satisfying

$$\kappa_{s,i} = \begin{cases} 1 & \text{if } s \in \mathcal{S}_i \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Using this notation we have that for arbitrary vectors $v_{c_i}, v_s, i \in \mathcal{C}$, the following identities hold:

$$\sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} \|v_{c_i} - v_s\| = \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{C}} \kappa_{s,i} \|v_{c_i} - v_s\| \tag{14a}$$

$$\sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} v_s = \sum_{s \in \mathcal{S}} v_s \tag{14b}$$

$$\sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} v_{c_i} = \sum_{i \in \mathcal{C}} \omega_i v_{c_i}. \tag{14c}$$

At this point, we have yet to define appropriate procedures for obtaining the cluster information $\mathcal{R}$, $\mathcal{S}_i$, ω_i and $\kappa_{s,i}$. These will be discussed in Sect. 3.

Consider now the compact representation of the KKT system (8),

$$\underbrace{\begin{bmatrix} K_{\mathcal{S}} & B_{\mathcal{S}} \\ B_{\mathcal{S}}^T & K_0 \end{bmatrix}}_{:=K} \underbrace{\begin{bmatrix} q_{\mathcal{S}} \\ q_0 \end{bmatrix}}_{:=q} = \underbrace{\begin{bmatrix} t_{\mathcal{S}} \\ t_0 \end{bmatrix}}_{:=t}, \tag{15}$$

where

$$\begin{aligned} K_{\mathcal{S}} &:= \text{blkdiag}\,\{K_1, \ldots, K_S\} && (16a)\\ B_{\mathcal{S}} &:= \text{rowstack}\,\{B_1, \ldots, B_S\} && (16b)\\ q_{\mathcal{S}} &:= \text{rowstack}\,\{q_1, \ldots, q_S\} && (16c)\\ t_{\mathcal{S}} &:= \text{rowstack}\,\{t_1, \ldots, t_S\}\,. && (16d)\end{aligned}$$

Here, $(t_0, t_{\mathcal{S}})$ are arbitrary right-hand side vectors and $(q_0, q_{\mathcal{S}})$ are solution vectors.[1] If the solution vector $(q_0, q_{\mathcal{S}})$ does not exactly solve (15), it will induce a residual vector that we define as $\epsilon_r^T := [\epsilon_{r_0}^T, \epsilon_{r_{\mathcal{S}}}^T]$ with

$$\begin{aligned} \epsilon_{r_0} &:= K_0 q_0 + B_{\mathcal{S}}^T q_{\mathcal{S}} - t_0 && (17a)\\ \epsilon_{r_{\mathcal{S}}} &:= K_{\mathcal{S}} q_{\mathcal{S}} + B_{\mathcal{S}} q_0 - t_{\mathcal{S}}. && (17b)\end{aligned}$$

The Schur system of (15) is given by

$$\underbrace{(K_0 - B_{\mathcal{S}}^T K_{\mathcal{S}}^{-1} B_{\mathcal{S}})}_{:=Z}\, q_0 = \underbrace{t_0 - B_{\mathcal{S}}^T K_{\mathcal{S}}^{-1} t_{\mathcal{S}}}_{:=t_Z}\,. \qquad (18)$$

Because $K_{\mathcal{S}}$ is block-diagonal, we have that

$$\begin{aligned} Z &= K_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} B_s && (19a)\\ t_Z &= t_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} t_s. && (19b)\end{aligned}$$

We now define the following:

$$\begin{aligned} K_{\mathcal{R}}^{\omega} &:= \text{blkdiag}\left\{\omega_1 K_{c_1}, \omega_2 K_{c_2}, \ldots, \omega_C K_{c_C}\right\} && (20a)\\ K_{\mathcal{R}}^{1/\omega} &:= \text{blkdiag}\left\{1/\omega_1 K_{c_1}, 1/\omega_2 K_{c_2}, \ldots, 1/\omega_C K_{c_C}\right\} && (20b)\\ B_{\mathcal{R}} &:= \text{rowstack}\left\{B_{c_1}, B_{c_2}, \ldots, B_{c_C}\right\} && (20c)\\ t_{\mathcal{R}} &:= \text{rowstack}\left\{t_{c_1}, t_{c_2}, \ldots, t_{c_C}\right\}. && (20d)\end{aligned}$$

In other words, $K_{\mathcal{R}}^{\omega}$ is a block-diagonal matrix in which each block entry K_{c_i} is weighted by the scalar weight ω_i and $K_{\mathcal{R}}^{1/\omega}$ is a block-diagonal matrix in which each block entry K_{c_i} is weighted by $1/\omega_i$. Note that

$$(K_{\mathcal{R}}^{1/\omega})^{-1} = (K_{\mathcal{R}}^{-1})^{\omega}, \qquad (21)$$

[1] We make a slight remark regarding notation: $K_{\mathcal{S}}$ is a block diagonal matrix while K_S in an entry of such block matrix. A similar observation applies to matrices $B_{\mathcal{S}}$ and vectors $q_{\mathcal{S}}, t_{\mathcal{S}}$ with corresponding entries B_S, q_S, t_S.

where

$$(K_{\mathcal{R}}^{-1})^{\omega} := \text{blkdiag}\left\{\omega_1 K_{c_1}^{-1}, \omega_2 K_{c_2}^{-1}, \ldots, \omega_C K_{c_C}^{-1}\right\}. \tag{22}$$

We now present the *clustering-based preconditioner* (CP),

$$\begin{bmatrix} K_{\mathcal{R}}^{1/\omega} & B_{\mathcal{R}} \\ B_{\mathcal{R}}^T & K_0 \end{bmatrix} \begin{bmatrix} \cdot \\ q_0 \end{bmatrix} = \begin{bmatrix} t_{\mathcal{R}} \\ t_0 + t_{CP} \end{bmatrix} \tag{23a}$$

$$K_s q_s = t_s - B_s q_0, \quad i \in \mathcal{C}, \; s \in \mathcal{S}_i, \tag{23b}$$

where

$$t_{CP} := \sum_{i \in \mathcal{C}} \omega_i B_{c_i}^T K_{c_i}^{-1} t_{c_i} - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} t_s \tag{24}$$

is a correction term that is used to achieve consistency between CP and the KKT system. In particular, the Schur system of (23a) is

$$\begin{aligned} \bar{Z} q_0 &= t_0 + t_{CP} - B_{\mathcal{R}}^T \left(K_{\mathcal{R}}^{1/\omega}\right)^{-1} t_{\mathcal{R}} \\ &= t_0 + t_{CP} - \sum_{i \in \mathcal{C}} \omega_i B_{c_i}^T K_{c_i}^{-1} t_{c_i} \\ &= t_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} t_s \\ &= t_Z, \end{aligned} \tag{25}$$

with

$$\begin{aligned} \bar{Z} &:= K_0 - \sum_{i \in \mathcal{C}} \omega_i B_{c_i}^T K_{c_i}^{-1} B_{c_i} \\ &= K_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_{c_i}^T K_{c_i}^{-1} B_{c_i}. \end{aligned} \tag{26}$$

Consequently, the Schur system of the preconditioner and of the KKT system have the same right-hand side. This property is key to establish spectral and error properties for the preconditioner. In particular, note that the solution of CP system (23a)-(23b) solves the perturbed KKT system,

$$\underbrace{\begin{bmatrix} K_{\mathcal{S}} & B_{\mathcal{S}} \\ B_{\mathcal{S}}^T & K_0 + E_Z \end{bmatrix}}_{:=\bar{K}} \begin{bmatrix} q_{\mathcal{S}} \\ q_0 \end{bmatrix} = \begin{bmatrix} t_{\mathcal{S}} \\ t_0 \end{bmatrix}, \tag{27}$$

where

$$E_Z := \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} B_s - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_{c_i}^T K_{c_i}^{-1} B_{c_i}, \tag{28}$$

is the Schur error matrix and satisfies $Z + E_Z = \bar{Z}$. The mathematical equivalence between CP system (23a)–(23b) and (27) can be established by constructing the Schur system of (27) and noticing that it is equivalent to (25). Moreover, the second-stage steps are the same. Consequently, *applying preconditioner CP is equivalent to using the perturbed matrix $\bar{K}$ as a preconditioning matrix for the KKT matrix K.*

The main idea behind preconditioner CP (we will use CP for short) is to compress the KKT system (15) into the smaller system (23a) which is cheaper to factorize. We solve this smaller system to obtain q_0, and we recover $q_{\mathcal{S}}$ from (23b) by factorizing the individual blocks K_s. We refer to the coefficient matrix of (23a) as the *compressed matrix.*

In the following, we assume that the Schur complements Z and $\bar{Z}$ are nonsingular. The nonsingularity of Z together with the assumption that all the blocks K_s are nonsingular implies (from the Schur complement theorem) that matrix K defined in (15) is nonsingular and thus the KKT system has a unique solution. The nonsingularity of $\bar{Z}$ together with the assumption that all the blocks K_s are nonsingular implies that the compressed matrix is nonsingular and thus CP has a unique solution. Note that we could have also assumed nonsingularity of matrix K directly and this, together with the nonsingularity of the blocks K_s, would imply nonsingularity of Z (this also from the Schur complement theorem). The same applies if we assume nonsingularity of the compressed matrix, which would imply nonsingularity of $\bar{Z}$.

Schur decomposition is a popular approach for solving structured KKT systems on parallel computers but it suffers from poor scalability with the dimension of q_0. The reason is that the Schur complement needs to be formed (this requires as many backsolves with the factors of K_s as the dimension of q_0) and factored (this requires a factorization of a nearly dense matrix of dimension q_0). We elaborate on these scalability issues in Sect. 4. We thus highlight that the Schur system representations are used only for analyzing CP.

Our preconditioning setting is summarized as follows. At each IP iteration k, we compute a step by solving the KKT system (8). We do so by using an iterative linear algebra solver such as GMRES, QMR, or BICGSTAB. Each minor iteration of the iterative linear algebra solver is denoted by $\ell = 0, 1, 2, \ldots,$. We denote the initial guess of the solution vector of (8) as $(\Delta w_0^\ell, \Delta w_{\mathcal{S}}^\ell)$ with $\ell = 0$. At each minor iterate ℓ, the iterative solver will request the application of CP to a given vector $(t_0^\ell, t_{\mathcal{S}}^\ell)$, and the solution vectors $(q_0^\ell, q_{\mathcal{S}}^\ell)$ of (23) are returned to the iterative linear algebra solver. *Perfect preconditioning* occurs when we solve (8) instead of (23) with the right-hand sides $(t_0^\ell, t_{\mathcal{S}}^\ell)$.

3 Preconditioner properties

In this section we establish properties for CP and we use these to guide the design of appropriate clustering strategies. Because the CP system (23) and the perturbed KKT system (27) are equivalent, we can establish the following result.

Lemma 1 *The preconditioned matrix $\bar{K}^{-1}K$ has $(n+m-n_0-m_0)$ unit eigenvalues, and the remaining (n_0+m_0) eigenvalues are bounded as*

$$|\lambda(\bar{K}^{-1}K)-1| \leq \frac{1}{\sigma_{min}(\bar{Z})}\|E_Z\|.$$

Proof The eigenvalues λ and eigenvectors $w := (w_{\mathcal{S}}, w_0)$ of $\bar{K}^{-1}K$ satisfy $\bar{K}^{-1}Kw = \lambda w$, and thus $Kw = \lambda \bar{K}w$. Consequently,

$$\begin{aligned} K_{\mathcal{S}}w_{\mathcal{S}} + B_{\mathcal{S}}w_0 &= \lambda(K_{\mathcal{S}}w_{\mathcal{S}} + B_{\mathcal{S}}w_0) \\ B_{\mathcal{S}}^T w_{\mathcal{S}} + K_0 w_0 &= \lambda B_{\mathcal{S}}^T w_{\mathcal{S}} + \lambda(K_0 + E_Z)w_0. \end{aligned}$$

From the first relationship we have $n+m-n_0-m_0$ unit eigenvalues. Applying Schur decomposition to the eigenvalue system, we obtain

$$\begin{aligned} Zw_0 &= \lambda(Z + E_Z)w_0 \\ &= \lambda \bar{Z}w_0. \end{aligned}$$

We can thus express the remaining n_0+m_0 eigenvalues of $\bar{K}^{-1}K$ as $\lambda = 1 + \epsilon_Z$ to obtain

$$\begin{aligned} |\epsilon_Z| &= \frac{\|E_Z w_0\|}{\|\bar{Z}w_0\|} \\ &\leq \frac{1}{\sigma_{min}(\bar{Z})}\|E_Z\|. \end{aligned}$$

The proof is complete. □

The above lemma is a direct consequence of Theorem 3.1 in [9]. From the definition of E_Z we note that the following bound holds:

$$|\lambda(\bar{K}^{-1}K)-1| \leq \frac{1}{\sigma_{min}(\bar{Z})}\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i}\left\|B_s^T K_s^{-1}B_s - B_{c_i}^T K_{c_i}^{-1}B_{c_i}\right\|. \quad (29)$$

Lemma 1 states that we can improve the spectrum of $\bar{K}^{-1}K$ by choosing clusters that minimize $\|E_Z\|$. This approach, however, would require expensive matrix operations. An interesting and tractable exception occurs when $Q_s = Q$, $W_s = W$, and $T_s = T$, $i \in \mathcal{C}, s \in \mathcal{S}_i$. This case is quite common in applications and arises when the scenario data is only defined by the right-hand sides b_s and the cost coefficients d_s of

(1). We refer to this case as the *special data case*. In this case we have that E_Z reduces to

$$E_Z = \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B^T \left(K_s^{-1} - K_{c_i}^{-1} \right) B. \tag{30}$$

We also have that K_s and K_{c_i} differ only in the diagonal matrices $X_s^{-1} V_s$ and $X_{c_i}^{-1} V_{c_i}$. We thus have,

$$K_s - K_{c_i} = \begin{bmatrix} \left(X_s^{-1} V_s - X_{c_i}^{-1} V_{c_i} \right) & 0 \\ 0 & 0 \end{bmatrix}. \tag{31}$$

If we define the vectors,

$$\gamma_s = \text{vec}\left(X_s^{-1} V_s \right), i \in \mathcal{C}, s \in \mathcal{S}_i \tag{32a}$$

$$\gamma_{c_i} = \text{vec}\left(X_{c_i}^{-1} V_{c_i} \right), i \in \mathcal{C}, \tag{32b}$$

we can establish the following result.

Theorem 1 *Assume that $Q_s = Q$, $W_s = W$, and $T_s = T$, $i \in \mathcal{C}, s \in \mathcal{S}_i$ holds. Let vectors γ_s, γ_{c_i} be defined as in (32). The preconditioned matrix $\bar{K}^{-1}K$ has $(n + m - n_0 - m_0)$ unit eigenvalues, and there exists a constant $c_K > 0$ such that the remaining $(n_0 + m_0)$ eigenvalues are bounded as*

$$|\lambda(\bar{K}^{-1}K) - 1| \leq \frac{c_K}{\sigma_{min}(\bar{Z})} \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{C}} \kappa_{s,i} \|\gamma_{c_i} - \gamma_s\|.$$

Proof From Lemma 1 we have that $n_0 + m_0$ eigenvalues λ of $\bar{K}^{-1}K$ are bounded as $|\lambda - 1| \leq \frac{1}{\sigma_{min}(\bar{Z})} \|E_Z\|$. We define the error matrix,

$$E_s := K_s - K_{c_i}, \ i \in \mathcal{C}, s \in \mathcal{S}_i$$

and use (30) and (31) to obtain the bound,

$$\begin{aligned} \|E_Z\| &\leq \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} \|B^T B\| \|K_s^{-1} - K_{c_i}^{-1}\| \\ &= \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} \|B^T B\| \|(K_{c_i} + E_s)^{-1} - K_{c_i}^{-1}\|. \end{aligned}$$

We have that

$$\begin{aligned} (K_{c_i} + E_s)^{-1} - K_{c_i}^{-1} &= -(K_{c_i} + E_s)^{-1} E_s K_{c_i}^{-1} \\ &= -K_s^{-1} E_s K_{c_i}^{-1}. \end{aligned}$$

This can be verified by multiplying both sides by $K_{c_i} + E_s$. We thus have

$$\begin{aligned}\|E_Z\| &\leq \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i} \|B^T B\| \|(K_{c_i} + E_s)^{-1} - K_{c_i}^{-1}\| \\ &\leq \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i} \|B^T B\| \|K_s^{-1}\| \|K_{c_i}^{-1}\| \|E_s\| \\ &\leq c_K \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i} \|\mathrm{vec}(X_{c_i}^{-1} V_{c_i}) - \mathrm{vec}(X_s^{-1} V_s)\|,\end{aligned}$$

with $c_K := \max_{i\in\mathcal{C}} \max_{s\in\mathcal{S}_i} \|B^T B\| \|K_s^{-1}\| \|K_{c_i}^{-1}\|$. The existence of c_K follows from the nonsingularity of K_s and K_{c_i}. The proof is complete. □

We now develop a bound of the preconditioning error for the *general data case* in which the scenario data is also defined by coefficient matrices. Notably, this bound does not require the minimization of the error $\|E_Z\|$. The idea is to bound the error induced by CP relative to the exact solution of the KKT system (15) (perfect preconditioner). This approach is used to characterize inexact preconditioners such as multigrid and nested preconditioned conjugate gradient [23]. We express the solution of CP obtained from (23) as $q^T = [q_{\mathcal{S}}^T, q_0^T]$ and that of the KKT system (15) as $q^{*T} = [q_{\mathcal{S}}^{*T}, q_0^{*T}]$. We define the error between q and q^* as $\epsilon := q - q^*$ and we seek to bound ϵ. If we decompose the error as $\epsilon^T = [\epsilon_{\mathcal{S}}^T, \epsilon_0^T]$, we have that $\epsilon_0 = q_0 - q_0^*$ and $\epsilon_{\mathcal{S}} = q_{\mathcal{S}} - q_{\mathcal{S}}^*$.

We recall that the Schur systems of (15) and of (23) and their respective solutions satisfy

$$Zq_0^* = t_Z \tag{33a}$$

$$\bar{Z}q_0 = t_Z. \tag{33b}$$

If we define the vectors,

$$\gamma_s = \left(B_s^T K_s^{-1} B_s\right) t_Z,\ i \in \mathcal{C}, s \in \mathcal{S}_i \tag{34a}$$

$$\gamma_{c_i} = \left(B_{c_i}^T K_{c_i}^{-1} B_{c_i}\right) t_Z,\ i \in \mathcal{C}. \tag{34b}$$

we can establish the following bound on the error $\epsilon = q - q^*$.

Lemma 2 *Assume that there exists $c_T > 0$ such that $\|(Z - \bar{Z})Z^{-1}t_Z\| \leq c_T\|(Z - \bar{Z})t_Z\|$ holds; then there exists $c_{ZK} > 0$ such that the preconditioner error ϵ is bounded as*

$$\|\epsilon\| \leq c_{ZK}\|(Z - \bar{Z})t_Z\|.$$

Proof From $\epsilon_0 = q_0 - q_0^*$ we have $\bar{Z}\epsilon_0 = \bar{Z}q_0 - \bar{Z}q_0^*$. From (33) we have $\bar{Z}q_0 = Zq_0^* = t_Z$ and thus $\bar{Z}\epsilon_0 = Zq_0^* - \bar{Z}q_0^*$. We thus have,

$$\begin{aligned}\bar{Z}\epsilon_0 &= Zq_0^* - \bar{Z}q_0^*\\ &= t_Z - \bar{Z}q_0^*\\ &= t_Z - \bar{Z}Z^{-1}t_Z\\ &= t_Z - (Z + (\bar{Z} - Z))Z^{-1}t_Z\\ &= (Z - \bar{Z})Z^{-1}t_Z.\end{aligned}$$

We recall that

$$\begin{aligned}q_{\mathcal{S}}^* &= K_{\mathcal{S}}^{-1}(t_{\mathcal{S}} - B_{\mathcal{S}}q_0^*)\\ q_{\mathcal{S}} &= K_{\mathcal{S}}^{-1}(t_{\mathcal{S}} - B_{\mathcal{S}}q_0)\end{aligned}$$

and thus

$$\begin{aligned}\epsilon_{\mathcal{S}} &= K_{\mathcal{S}}^{-1}B_{\mathcal{S}}(q_0^* - q_0)\\ &= -K_{\mathcal{S}}^{-1}B_{\mathcal{S}}\epsilon_0.\end{aligned}$$

We thus have

$$\begin{aligned}\|\epsilon_0\| &\le c_Z\|(Z - \bar{Z})t_Z\|\\ \|\epsilon_{\mathcal{S}}\| &\le c_{K_{\mathcal{S}}}\|\epsilon_0\|,\end{aligned}$$

with $c_Z := \|\bar{Z}^{-1}\|c_T$ and $c_{K_{\mathcal{S}}} := \|K_{\mathcal{S}}^{-1}B_{\mathcal{S}}\|$. The existence of c_Z follows from the assumption that $\bar{Z}$ is nonsingular. The existence of $c_{K_{\mathcal{S}}}$ follows from the assumption that the blocks K_s are nonsingular and thus $K_{\mathcal{S}}$ is nonsingular. The result follows from $\|\epsilon\| \le \|\epsilon_0\| + \|\epsilon_{\mathcal{S}}\|$ and by defining $c_{ZK} := c_Z(1 + c_{K_{\mathcal{S}}})$. □

The assumption that there exists $c_T > 0$ such that $\|(Z-\bar{Z})Z^{-1}t_Z\| \le c_T\|(Z-\bar{Z})t_Z\|$ holds is trivially satisfied when Z^{-1} and $\bar{Z}$ commute (i.e., $\bar{Z}Z^{-1}$ is a symmetric matrix). In this case we have that $c_T = \|Z^{-1}\|$. The matrices also commute in the limit $\bar{Z} \to Z$ because $\bar{Z}Z^{-1} = ZZ^{-1} + (\bar{Z} - Z)Z^{-1}$ and thus $\bar{Z}Z^{-1} \to I$. When Z and $\bar{Z}$ do not commute we require that $\|(Z - \bar{Z})Z^{-1}t_Z\|$ decreases when $\|(Z - \bar{Z})t_Z\|$ does. We validate this condition empirically in Sect. 4.

Theorem 2 *Let vectors γ_s, γ_{c_i} be defined as in (34). The preconditioner error ϵ is bounded as*

$$\|\epsilon\| \le c_{ZK}\sum_{s\in\mathcal{S}}\sum_{i\in\mathcal{C}} k_{s,i}\|\gamma_{c_i} - \gamma_s\|,$$

with c_{ZK} defined in Lemma 2.

Proof From (34) and (28) we have that

$$\begin{aligned}\bar{Z}t_Z - Zt_Z &= E_Zt_Z\\ &= \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i} B_s^TK_s^{-1}B_st_Z - \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}_i} B_{c_i}^TK_{c_i}^{-1}B_{c_i}t_Z\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} (B_s^T K_s^{-1} B_s t_Z - B_{c_i}^T K_{c_i}^{-1} B_{c_i} t_Z) \\
&= \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} (\gamma_s - \gamma_{c_i}).
\end{aligned}
$$

We bound this expression to obtain,

$$
\begin{aligned}
\|\bar{Z} t_Z - Z t_Z\| &= \left\| \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} (\gamma_s - \gamma_{c_i}) \right\| \\
&\leq \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} \|\gamma_{c_i} - \gamma_s\| \\
&= \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{C}} \kappa_{s,i} \|\gamma_{c_i} - \gamma_s\|.
\end{aligned}
$$

The result follows from Lemma 2. □

We can see that the properties of CP are related to a metric of the form

$$
\mathcal{D}_C := \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{C}} \kappa_{s,i} \|\gamma_{c_i} - \gamma_s\|. \tag{35}
$$

This is the *distortion metric* widely used in clustering analysis [2]. The distortion metric is (partially) minimized by K-means, K-medoids, and hierarchical clustering algorithms to determine $\kappa_{s,i}$ and γ_{c_i}. The vectors γ_s are called *features*, and γ_{c_i} is the centroid of cluster $i \in \mathcal{C}$ (we can also pick the scenario that is closest to the centroid if the centroid is not an element of the scenario set). The distortion metric is interpreted as the accumulated distance of the elements of the cluster relative to the centroid. If the distortion is small, then the scenarios in a cluster are similar. The distortion metric can be made arbitrarily small by increasing the number of clusters and is zero in the limit with $S = C$ because each cluster is given by one scenario. Consequently, we see that Theorems 1 and 2 provide the necessary insights to derive clusters using different sources of information of the scenarios.

Theorem 1 suggests that, in the special data case with features defined as $\gamma_s = \text{vec}(X_s^{-1} V_s)$, the spectrum of $\bar{K}^{-1} K$ can be made arbitrarily close to one if the distortion metric is made arbitrarily small. This implies that the *definition of the features is consistent.* We highlight, however, that the bounds of Theorem 1 assume that the clustering parameters are given (i.e., the sets $\mathcal{C}$ and $\mathcal{C}_i$ are fixed). Consequently, the constants c_K, and $\sigma_{min}(\bar{Z})$ change when the clusters are changed. Because of this, we cannot guarantee that reducing the distortion metric will indeed improve the quality of the preconditioner. The aforementioned constants depend in nontrivial ways on the clustering parameters and it is thus difficult to obtain bounds for them. In the next section we demonstrate empirically, however, that the constants c_K and $\sigma_{min}(\bar{Z})$ are insensitive to the clustering parameters. Consequently, reducing the distortion metric in fact improves the quality of the preconditioner. We leave the theoretical treatment of this issue as part of future work.

We can obtain useful insights from the special data case. First note that the scenarios are clustered at each IP iteration k because the matrices $X_s^{-1}V_s$ change along the search. The clustering approach is therefore adaptive, unlike outside-the-solver scenario clustering approaches. In fact, it is not possible to derive spectral and error properties for preconditioners based on clustering of problem data alone. Our approach focuses directly on the contributions $X_s^{-1}V_s$ and thus assumes that the problem data enters indirectly through the contributions $X_s^{-1}V_s$, which in turn affect the structural properties of the KKT matrix. The features $\gamma_s = \text{vec}(X_s^{-1}V_s)$ have an important interpretation: these reflect the contribution of each scenario to the logarithmic barrier function. From complementarity we have that $\|X_s\| \gg 0$ implies $\|V_s\| \approx 0$ and $\|X_s^{-1}V_s\| \approx 0$. In this case we say that there is weak activity in the scenario and we have from (6) that $H_s = Q_s + X_s^{-1}V_s \approx Q_s$. Consequently, the primal-dual term $X_s^{-1}V_s$ for a scenario with weak activity puts little weight on the barrier function. In the opposite case in which the scenario has strong activity we have that $\|V_s\| \gg 0$, $\|X_s\| \approx 0$, and $\|X_s^{-1}V_s\| \gg 0$. In this case we thus have that a scenario with strong activity puts a large weight on the barrier function. This reasoning is used in [12,24] to eliminate the scenarios with weak activity. In our case we propose to cluster scenarios with similar activities. Clustering allows us to eliminate redundancies in both active and inactive scenarios and to capture outliers. In addition, this strategy avoids the need to specify a threshold to classify weak and strong activity.

Theorem 2 provides a mechanism to obtain clusters for the general data case in which the scenario data is defined also by the coefficient matrices. The result states that we can bound the preconditioning error using the Schur complement error $E_Z = \bar{Z} - Z$ projected on the right-hand side vector t_Z. Consequently, the error can be bounded by the distortion metric with features defined in (34). This suggests that the error can be made arbitrarily small if the distortion is made arbitrarily small. Moreover, it is not necessary to perform major matrix operations. As in the special data case of Theorem 1, however, the bounding constant c_Z of Theorem 2 depends on the clustering parameters. Moreover, we need to verify that the term $\|(Z - \bar{Z})Z^{-1}t_Z\|$ decreases when $\|(Z - \bar{Z})t_Z\|$ does. In the next section we verify these two assumptions empirically.

The error bound of Theorem 2 requires that clustering tasks and the factorization of the compressed matrix be performed at each minor iteration ℓ of the iterative linear algebra solver. The reason is that the features (34) change with t_Z^ℓ. Performing these tasks at each minor iteration, however, is expensive. Consequently, we perform these tasks only at the first minor iteration $\ell = 0$. If the initial guess of the solution vector of the KKT system is set to zero ($\Delta w_0^\ell = 0$ and $\Delta w_{\mathcal{S}}^\ell = 0$) and if GMRES, QMR, or BICGSTAB schemes are used, this is equivalent to performing clustering using the features

$$\gamma_s = \left(B_s^T K_s^{-1} B_s\right) r_Z, i \in \mathcal{C}, s \in \mathcal{S}_i \tag{36a}$$

$$\gamma_{c_i} = \left(B_{c_i}^T K_{c_i}^{-1} B_{c_i}\right) r_Z, i \in \mathcal{C}, \tag{36b}$$

where

$$\begin{aligned} r_Z &= t_Z^0 \\ &= r_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_s^T K_s^{-1} r_s \end{aligned} \tag{37}$$

is the right-hand side of the Schur system of (8).

4 Numerical results

In this section we discuss implementation issues of CP and present numerical results for benchmark problems in the literature and a large-scale stochastic market clearing problem. We begin by summarizing the procedure for computing the step $(\Delta x_k, \Delta y_k, \Delta \nu_k)$ at each IP iteration k.

Step computation scheme

1. **Initialization**. Given iterate (x_k, y_k, ν_k), number of clusters C, tolerance τ_k, and maximum number of linear solver iterates m_{it}.
2. **Get clustering information.**
   - 2.0 Compute features $\gamma_s,\ s \in \mathcal{S}$ as in (32) or (36).
   - 2.1 Obtain $\kappa_{s,i}$ and γ_{c_i} using a clustering algorithm (e.g., K-means, hierarchical).
   - 2.2 Use $\kappa_{s,i}$ to construct $\mathcal{C}$, $\mathcal{R}$, and ω_i.
   - 2.3 Construct and factorize compressed matrix

     $$\begin{bmatrix} K_{\mathcal{R}}^{1/\omega} & B_{\mathcal{R}} \\ B_{\mathcal{R}}^T & K_0 \end{bmatrix}$$

     and factorize scenario matrices $K_s,\ i \in \mathcal{C}, s \in \mathcal{S}_i$.
3. **Get step.**
   - 3.1 Call iterative linear solver to solve KKT system (15) with right-hand sides $(r_0, r_{\mathcal{S}})$, set $\ell = 0$, and initial guess $\Delta w_0^\ell = 0$ and $\Delta w_{\mathcal{S}}^\ell = 0$. At each minor iterate $\ell = 0, 1, \ldots,$ of the iterative linear solver, DO:
     - 3.1.1 Use factorization of compressed matrix and of $K_{\mathcal{S}}$ to solve CP (23a)-(23b) for right-hand sides $(t_0^\ell, t_{\mathcal{S}}^\ell)$ and RETURN solution $(q_0^\ell, q_{\mathcal{S}}^\ell)$.
     - 3.1.2 From (17), get ϵ_r^ℓ using solution vector $(\Delta w_0^\ell, \Delta w_{\mathcal{S}}^\ell)$ and right-hand side vectors $(r_0, r_{\mathcal{S}})$. If $\|\epsilon_r^\ell\| \leq \tau_k$, TERMINATE.
     - 3.1.3 If $\ell = m_{it}$, increase C, and RETURN to Step 3.1.
   - 3.2 Recover $(\Delta x_k, \Delta y_k)$ from $(\Delta w_0^\ell, \Delta w_{\mathcal{S}}^\ell)$.
   - 3.3 Recover $\Delta \nu_k$ from (7).

We call our clustering-based IP framework `IP-CLUSTER`. The framework is written in C++ and uses MPI for parallel computations. In this implementation we use the primal-dual IP algorithm of Mehrotra [19]. We use the matrix templates and direct linear algebra routines of the `BLOCK-TOOLS` library [15]. This library is specialized to block matrices as those arising in this work and greatly facilitated the implementation. Within BLOCK-TOOLS, we use its MA57 interface to perform all direct linear algebra operations. We use the GMRES implementation within the PETSc library (http://www.mcs.anl.gov/petsc) to perform all iterative linear algebra operations. We have implemented serial and parallel versions of CP. We highlight that the parallel version performs the factorizations of (23b) in parallel and exploits the block-bordered-diagonal structure of the KKT matrix to perform matrix-vector operations in parallel as well. We use the K-means and hierarchical clustering implementations of the C-Clustering library (http://bonsai.hgc.jp/~mdehoon/software/cluster/software.htm). To implement the market clearing models of Sect. 4.2 we use an interface to AMPL to create individual instances (.nl files) for each scenario and indicate first-stage variables and constraints using the suffix capability.

4.1 Benchmark problems

We consider stochastic variants of problems obtained from the CUTEr library and benchmark problems (SSN, GBD, LANDS, 20TERM) reported in [17]. The deterministic CUTEr QP problems have the form

$$\min \frac{1}{2}y^T Qy + d^T y, \text{ s.t. } Ay = b, \;\; y \geq 0. \tag{38}$$

We generate a stochastic version of this problem by defining b as a random vector. We create scenarios for this vector b_s, $s \in \mathcal{S}$ using the nominal value b as mean and a standard deviation $\pm\sigma = 0.5b$. We then formulate the two-stage stochastic program:

$$\min e^T y_0 + \sum_{s\in\mathcal{S}} p_s \left(\frac{1}{2}y_s^T Qy_s + d^T y_s\right) \tag{39a}$$

$$\text{s.t.} \quad Ay_s = b_s, s \in \mathcal{S} \tag{39b}$$

$$y_s + y_0 \geq 0, \;\; s \in \mathcal{S} \tag{39c}$$

$$y_0 \geq 0. \tag{39d}$$

Here, we set $p_s = 1/|\mathcal{S}|$. We first demonstrate the quality of CP in terms of the number of GMRES iterations. For all cases, we assume a scenario compression rate of 75 % (only 25 % of the scenarios are used in the compressed matrix), and we solve the problems to a tolerance of 1×10^{-6}. We use the notation x % to indicate the compression rate (i.e., the preconditioner CP uses 100-x % of the scenarios to define the compressed matrix). A compression rate of 0 % indicates that the entire scenario set is used for the preconditioner (ideal). A compression rate of 100 % indicates that no preconditioner is used. We set the maximum number of GMRES iterations m_{it} to 100.

For this first set of results we cluster the scenarios using a hierarchical clustering algorithm with the features (34). The results are presented in Table 1. As can be seen, the performance of CP is satisfactory in all instances, requiring fewer than 20 GMRES iterations per interior-point iteration (this is labeled as LAit/IPit). We attribute this to

Table 1 Performance of naive and preconditioner CPs in benchmark problems

Problem	S	n	m	NPI (75 %)			CP (75 %)			NPII (75 %)		
				IPit	LAit	LAit/IPit	IPit	LAit	LAit/IPit	IPit	LAit	LAit/IPit
HS53	100	1010	800	19	152	8	19	113	5	20	112	5
LOTSCHD	100	1212	700	27	911*	33	25	203	8	26	178	6
HS76	100	707	300	24	626*	26	23	98	4	24	107	4
HS118	100	5959	4400	47	1499*	31	47	409	8	50	484	9
QPCBLEND	100	11,514	7400	57	258	4	57	253	4	55	273	4
ZECEVIC2	100	606	400	27	451*	16	29	111	3	26	107	4
QPTEST	100	505	300	23	569*	24	23	108	4	26	109	4
SSN	100	70,689	8600	114	738*	6	114	1857	16	114	2506	21
GBD	1000	10,017	5000	24	627*	26	24	144	6	24	92	4
LANDS	1000	12,004	7003	29	481*	16	29	115	3	29	122	4
20TERM	100	76,463	12,404	57	581*	10	57	976	17	58	905*	16

Asterisks indicate the instances in which the number of clusters needs to be increased next to the total number of GMRES iterations the particular structure of CP, which enable us to pose the preconditioning systems in the equivalent form (27) and to derive favorable spectral properties and error bounds. To support these observations, we have also experimented with a couple of *naive* preconditioners. The first naive preconditioner (NPI) has the form:

$$\begin{bmatrix} \bar{K}_{\mathcal{S}} & \bar{B}_{\mathcal{S}} \\ \bar{B}_{\mathcal{S}}^T & K_0 \end{bmatrix} \begin{bmatrix} q_{\mathcal{S}} \\ q_0 \end{bmatrix} = \begin{bmatrix} t_{\mathcal{S}} \\ t_0 \end{bmatrix}, \tag{40}$$

where

$$\bar{K}_{\mathcal{S}} := \text{blkdiag}\left\{K_{c_1}, \ldots, K_{c_1}, K_{c_2}, \ldots, K_{c_2}, \ldots, K_{c_C}, \ldots, K_{c_C}\right\} \tag{41a}$$

$$\bar{B}_{\mathcal{S}} := \text{rowstack}\left\{B_{c_1}, \ldots, B_{c_1}, B_{c_2}, \ldots, B_{c_2}, \ldots, B_{c_C}, \ldots, B_{c_C}\right\}. \tag{41b}$$

We can see that NPI replaces the block matrix elements of the cluster with those of the scenario representing the cluster. This, in fact, is done implicitly by the compressed system (23a). Note also that the right-hand side of NPI is consistent with that of the KKT system. The design of NPI seems reasonable at first sight but it has several structural deficiencies. We highlight these by noticing that the Schur system of NPI has the form

$$\bar{Z} q_0 = t_0 - \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}_i} B_{c_i}^T K_{c_i}^{-1} t_s. \tag{42}$$

This system has the same Schur matrix as that of the Schur system of CP (25) but does not have the same right-hand side. Moreover, the second-stage steps obtained from NPI are

$$K_{c_i} q_s = t_s - B_{c_i} q_0, \quad i \in \mathcal{C}, \; s \in \mathcal{S}_i. \tag{43}$$

By comparing (43) with (23b) we can see that the recovery of the second-stage steps in NPI does not use the second-stage matrices K_s, B_s corresponding to each scenario (as is done in CP). We also consider an alternative naive preconditioner (NPII) to analyze the impact of the second-stage step (23b). This preconditioner computes q_0 using (40) as in NPI but computes the second-stage steps using (43) as in CP. Consequently, NPII and CP only differ in the way q_0 is computed. It is not difficult to verify that the solution of NPII is equivalent to the solution of the system:

$$\begin{bmatrix} K_{\mathcal{S}} & B_{\mathcal{S}} \\ B_{\mathcal{S}}^T & K_0 + E_Z \end{bmatrix} \begin{bmatrix} q_{\mathcal{S}} \\ q_0 \end{bmatrix} = \begin{bmatrix} t_{\mathcal{S}} \\ t_0 + t_{CP} \end{bmatrix}. \tag{44}$$

By comparing the equivalent system (27) of CP and the equivalent system (44) of NPII we can see that NPII introduces the additional perturbation t_{CP} on the right-hand side.

The structural deficiencies of NPI and NPII prevent us from obtaining the error bounds of Theorems 1 and 2 and highlight the importance of the structure of CP. In

Table 2 Performance of preconditioned and unpreconditioned strategies

Problem	S	n	Compress	IPit	LAit	LAit/IPit
HS53	100	1010	100 %	19	12861	676
			75 % (NPI)	19	152	8
			75 % (CP)	19	113	5
			75 % (NPII)	20	112	5

Table 1 we compare the performance of the different preconditioners. We can see that the performance of NPI is not competitive. In particular, CP outperforms NPI in nine instances out of eleven. Moreover, in all instances except HS53 and QPCBLEND, it was necessary to refine preconditioner NPI in several iterations (this is done by increasing the number of clusters). We highlight instances in which this occurs using a star next to the total number of GMRES iterations. The performance of NPII is highly competitive with that of CP. In fact, NPII performs slightly better than CP in several instances. For problem 20TERM, however, it was necessary to increase the number of clusters for NPII in some iterations. We can thus conclude that CP has in general better performance and is more robust. Moreover, we can conclude that the second-stage step (23b) plays a key role.

In Table 2 we compare the performance of CP with that of the unpreconditioned strategy (compression rate of 100 %) and with that of the naive strategies. We only show results for a single instance to illustrate that the matrices of the benchmark problems are nontrivial and preconditioning is indeed needed.

We note that the instances reported in Tables 1 and 2 are small ($n < 100,000$). In most of these small instances we found that the solution times obtained with full factorization are shorter than those obtained with CP. This is because the overhead introduced by the iterative linear solver is not sufficient to overcome the gains obtained by compressing the linear system. We illustrate this behavior in Table 3 where we compare the performance of full factorization (0 % compression rate) with that of preconditioner CP for problem 20TERM. We clearly see that the total solution times (denoted as θ_{tot}) obtained with full factorization are significantly shorter than those obtained with CP. Most notably, this trend holds for problems with up to 600,000 variables and the times scale linearly with the number of scenarios. These results illustrate that sparse direct factorization codes such as MA57 can efficiently handle certain large-scale problems. As we show in Sect. 4.2, this efficiency enables us to overcome scalability bottlenecks of Schur decomposition. Full factorization, however, will eventually become expensive as we increase the problem size and, at this point, the use of CP becomes beneficial. We illustrate this in Table 6 where we compare the performance of CP with that of full factorization for two large instances. Instance QSC2015 has 63,717 variables, while instance AUG3DC has 131,682 variables. We use θ_{fact} to denote the time spent in the factorization of the compressed matrix and of the block matrices, θ_{clus} to denote the time spent performing clustering operations, and θ_{gmres} to denote the time spent in GMRES iterations (without considering factorization operations in the preconditioner). As can be seen, the solution times of full factorization are dramatically reduced by using CP.

Table 3 Effect of compression rates on 20TERM problem

S	n	Compress (%)	IPit	LAit/IPit	θ_{tot}
100	76,463	0	54	–	16
		50	57	10	54
		75	57	19	79
		87	57	17	69
200	152,863	0	69	–	44
		50	72	9	137
		75	72	14	166
		87	72	18	185
400	305,663	0	87	–	108
		50	92	20	578
		75	92	21	555
		87	92	23	570
800	611,263	0	88	–	232
		50	97	25	1440
		75	97	25	1427
		87	97	27	1417

Table 4 Performance of different clustering strategies for benchmark problems (Theorem 1)

Problem	Compress (%)	c_K	$\sigma_{min}(\bar{Z})$	$\|Z - \bar{Z}\|$	$\mathcal{D}_C$
LANDS	50	$6.8 \times 10^{+2}$	3.7×10^{-3}	3.9×10^{-2}	6.5×10^{-2}
	75	$6.8 \times 10^{+2}$	3.8×10^{-3}	1.8×10^{-1}	5.8×10^{-1}
GBD	50	$4.5 \times 10^{+12}$	6.8×10^{-2}	6.6×10^{-3}	5.0×10^{-4}
	75	$4.5 \times 10^{+12}$	6.7×10^{-2}	6.1×10^{-2}	1.6×10^{-3}

From Table 3 we can also see that the performance of CP deteriorates as we increase the compression rate. This is because the distortion metric increases as we increase the compression rate and thus the quality of the preconditioner deteriorates, as suggested by Theorems 1 and 2. We recall, however, that the bounds provided in these theorems depend on constants that change with the clustering parameters. Consequently, it is not obvious that reducing the distortion metric will improve the quality of the preconditioner. We designed a numerical experiment to gain more insight into this issue. In the experiment we compute the constants and metrics of Theorems 1 and 2 for two additional instances (GBD and LANDS) and for two different compression rates (50 and 75 %). We only report the results at a single iteration because we observed similar behavior at other iterations. The results are summarized in Tables 4 and 5. As can be seen in Table 4, the constants c_K and $\sigma_{min}(\bar{Z})$ of Theorem 1 are insensitive to the compression rate. The distortion metric $\mathcal{D}_C$, on the other hand, changes by an order of magnitude. We also report the Schur complement error $\|E_Z\| = \|\bar{Z} - Z\|$ and we see that this error changes by an order of magnitude as well. In Table 5

Table 5 Performance of different clustering strategies for benchmark problems (Theorem 2)

Problem	Compress (%)	c_Z	$\mathcal{D}_C$	$\|(\bar{Z}-Z)Z^{-1}t_Z\|$	$\|(\bar{Z}-Z)t_Z\|$
LANDS	50	$5.7 \times 10^{+2}$	$1.0 \times 10^{+1}$	$2.0 \times 10^{+0}$	$6.1 \times 10^{+2}$
	75	$5.8 \times 10^{+2}$	$1.0 \times 10^{+2}$	$2.9 \times 10^{+1}$	$8.5 \times 10^{+3}$
GBD	50	$9.2 \times 10^{+0}$	1.0×10^{-1}	5.6×10^{-3}	9.2×10^{-2}
	75	$9.2 \times 10^{+0}$	3.5×10^{-1}	2.0×10^{-2}	7.1×10^{-1}

Table 6 Performance of different clustering strategies for benchmark problems

Problem	Compress (%)	Clustering	IPit	θ_{tot}	θ_{fact}	θ_{clus}	θ_{gmres}	LAit	LAit/IPit
QSC205	0		110	1331	1321				
	50	$X^{-1}V$	110	220	157	5	42	747	6
	75	$X^{-1}V$	110	91	25	6	45	933	8
	50	r_Z	110	229	161	5	43	747	6
	75	r_Z	110	89	24	5	43	924	8
AUG3DC	0		11	1427	1423				
	50	$X^{-1}V$	11	96	84	0.3	6	26	2
	75	$X^{-1}V$	11	24	13	0.3	6	27	2
	50	r_Z	11	93	80	0.3	6	26	2
	75	r_Z	11	25	13	0.3	6	27	2

we can see that the constant c_Z of Theorem 2 is insensitive to the compression rate but the distortion is rather sensitive as well. Moreover, we can see that metrics $\|(\bar{Z}-Z)Z^{-1}t_Z\|$, $\|(\bar{Z}-Z)t_Z\|$, and $\mathcal{D}_C$ change significantly with the compression rate. We highlight that the distortion metric of Theorem 1 is defined using the features (32) while the distortion metric of Theorem 2 is defined using the features (34). *From these results we can conclude that the distortion metrics proposed are indeed appropriate indicators of preconditioning quality and can thus be used to guide the construction of the preconditioners.*

From Table 3 we can also see that the deterioration of performance due to increasing compression rates becomes less pronounced as we increase the number of scenarios. The reason is that more redundancy is observed as we increase the number of scenarios and, consequently, compression potential increases. This behavior has been found in several instances and indicates that it is possible to deal with problems with a large number of scenarios.

In Table 6 we compare the performance of different clustering strategies. To this end, we perform clustering using features (32) (we label this as $X^{-1}V$) and using (36) (we label this as r_Z). As can be seen, the performance of both clustering strategies is very similar. This demonstrates that the design of the features (32) and (34) is consistent.

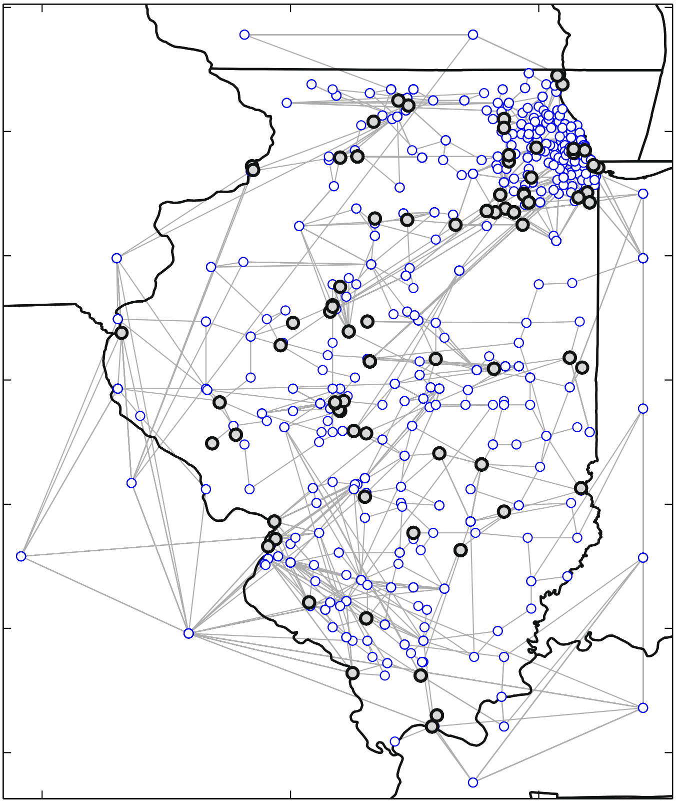

Fig. 1 Illinois transmission network. *Dark dots* are supply nodes and *blue dots* are demand nodes (Color figure online)

4.2 Stochastic market clearing problem

We demonstrate the computational efficiency of the preconditioner by solving a stochastic market-clearing model for the entire Illinois power grid system [21,25,27]. The system is illustrated in Fig. 1. The stochastic programming formulation is given by

$$\min_{x_i, X_i(s)} \sum_{i\in\mathcal{G}} \left(\beta_i x_i + \sum_{s\in\mathcal{S}} p_s \left[\beta_i^+ (X_i(s) - x_i)_+ - \beta_i^- (X_i(s) - x_i)_- \right] \right)$$

$$\text{s.t. } \tau_n(f) + \sum_{i\in\mathcal{G}(n)} x_i = d_n,\ n \in \mathcal{N} \tag{45a}$$

$$\tau_n(F(s)) - \tau_n(f) + \sum_{i\in\mathcal{G}(n)} (X_i(s) - x_i) = 0,\ \ n \in \mathcal{N}, s \in \mathcal{S} \tag{45b}$$

$$f, F(s) \in \mathcal{F},\ \ s \in \mathcal{S} \tag{45c}$$

Table 7 Performance of Schur decomposition approach

S	n	IPit	θ_{tot}	$\theta_{factschur}$	$\theta_{formschur}$	$\theta_{factblock}$
1	30,472	55	31280	27236	4023	4

$$(x_i, X_i(s)) \in \mathcal{X}_i(s), \quad i \in \mathcal{G}, s \in \mathcal{S}. \tag{45d}$$

Here, $\mathcal{N}$ denotes the set of network nodes and the set of all suppliers is denoted by $\mathcal{G}$. Subsets $\mathcal{G}(n)$ denote the set of suppliers connected to node $n \in \mathcal{N}$. The forward (first-stage) dispatched quantities for players are x_i, and the spot (second-stage) quantities under scenario s are $X_i(s)$. Symbol f denotes the vector of all line flows and $\tau_n(\cdot)$ are flow injections into node $n \in \mathcal{N}$. Similarly, $F(s)$ denotes the vector of line flows for each scenario s. The demand is assumed to be deterministic and inelastic and is represented by $d_n, \ n \in \mathcal{N}$. The sets $\mathcal{F}$ and $\mathcal{X}_i(s)$ are polyhedral and define lower and upper bounds for the flows and dispatch quantities. The objective of the market clearing problem is to minimize the forward dispatch cost plus the expected recourse dispatch cost. Here $[y]_+ = \max\{y, 0\}$ and $[y]_- = \max\{-y, 0\}$. The coefficients β_i denote the supply price bids, and β_i^+ and β_i^- are price bids for corrections of the suppliers. A supplier i asks $\beta_i^+ > \beta_i$ to sell additional power or asks $\beta_i^- < \beta_i$ to buy power from the system (e.g., reduces output). The scenarios $s \in \mathcal{S}$ characterize the randomness in the model due to unpredictable supply capacities (in this case wind power).

The market clearing model has *large first-stage dimensionality*. The Schur complement has a dimension of 64,199 and has a large dense block. In Table 7 we present the solution times for this problem using a Schur decomposition strategy for *a single scenario*. Here, θ_{tot} is the total solution time, $\theta_{factschur}$ is the time spent factorizing the Schur complement, $\theta_{formschur}$ is the time spent forming the Schur complement, and $\theta_{factblock}$ is the time spent factorizing the scenario blocks (in this case just one block). All times are reported in seconds. The solution time for this problem is 8.7 hr, with 13 % of the time spent forming the Schur complement and 87 % spent factorizing the Schur complement. Note that if more scenarios are added, the time spent forming and factorizing the Schur complement will dominate (even if the scenarios can be parallelized). Iterative strategies applied to the Schur complement system can avoid the time spent forming the Schur complement but not the factorization time because a preconditioner with a large dense block still needs to be factored [20].

We now assess the serial performance of CP. By comparing Tables 7 and 8 we can see that the full factorization approach will be as efficient as Schur decomposition for problems with up to 64 scenarios. In other words, it would be faster to factorize the full sparse KKT system than forming and factorizing the large Schur complement. The fast growth in solution time of the full factorization approach is remarkable, however. We attribute this to the tight connectivity induced by the network constraints which introduce significant fill-in. CP reduces the solution times of full factorization by a factor of 2 for the problem with 32 scenarios and by a factor of 11 for the problem with 64 scenarios. We highlight that CP is highly effective, requiring on average 6–11 GMRES iterations per IP iteration for compression rates of 50 % and 12–18 iterations

Table 8 Serial performance of preconditioner CP against full factorization for stochastic market clearing problem

S	n	Compress (%)	Cluster.	IPit	θ_{tot}	θ_{fact}	θ_{clus}	θ_{gmres}	LAit	LAit/IPit
16	309,187	0		57	473	452				
		50	$X^{-1}V$	57	544	119	0.4	325	631	11
		50	r_Z	57	508	117	0.15	296	519	9
32	606,483	0		65	3480	3414				
		50	$X^{-1}V$	65	1477	661	8	606	574	8
		75	$X^{-1}V$	65	1479	145	8	1141	1194	18
		50	r_Z	65	1347	672	3	459	398	6
		75	r_Z	65	1131	150	3	769	804	12
64	1,201,075	0		64	28022	27883				
		50	$X^{-1}V$	64	5163	3513	29	1292	660	10
		75	$X^{-1}V$	64	2878	656	29	1844	902	14
		87	$X^{-1}V$	64	2499	135	29	1990	1040	16
		50	r_Z	64	5238	3492	12	1349	666	10
		75	r_Z	64	3003	659	12	1924	937	14
		87	r_Z	64	2440	115	12	1944	1147	17

Table 9 Parallel performance of preconditioner CP against full factorization for stochastic market clearing problem

S	n	MPI Proc.	Compress (%)	IPit	θ_{tot}	θ_{fact}	θ_{clus}	$\theta_{factblock}$	θ_{gmres}	LAit	LAit/IPit
64	1,201,075	1	0	64	28022	27883					
		1	87	64	2440	115	12	288	1944	1147	17
		2	87	64	1211	116	12	147	892	1025	16
		4	87	64	817	134	12	80	592	919	14
		8	87	64	658	152	12	44	398	905	14
		1	94	64	3223	49	12	327	2764	1489	23
		2	94	64	1558	43	12	151	1306	1471	22
		4	94	64	993	49	12	84	801	1420	22
		8	94	64	733	54	12	46	570	1409	22

for compression rates of 75 %. We also observe that the performance of different clustering strategies is similar.

For the problem with 64 scenarios we can see that the solution time of CP is not significantly reduced further as we increase the compression rate from 75 to 87 % (even if the factorization time is dramatically reduced). This is because the time spent in GMRES to perform backsolves and matrix-vector operations dominates the factorization time. We mitigate this by using the parallel implementation of CP. The results are presented in Table 9. We can see that the solution time spent in GMRES to perform backsolves and matrix-vector operations is dramatically reduced by exploiting the block-bordered-diagonal structure of the KKT matrix. *This enables us to solve a market clearing problem with over 1.2 million variables in 10 min, as opposed to 9 hours using the full factorization approach. This represents a speed up factor of 42*. By comparing the parallel results with those of Table 7 we can also see that the Schur complement approach is not competitive because of the time needed to form and factorize the Schur complement (this holds even for a single scenario).

From Table 9 we can see that scalability slows down as we increase the number of processes. This is because the remaining serial components (beyond backsolves and matrix-vector operations) of CP start dominating. This overhead includes operations inside the GMRES algorithm itself. We are currently investigating ways to parallelize these operations.

In Table 9 we also present experiments using a compression rate of 94 %. We performed these experiments to explore the performance limit of CP. We can see that the performance of CP deteriorates in terms of total solution time because the number of GMRES iterations (and thus time) increases. Consequently, it does not pay off to cluster the KKT system further. We highlight, however, that the deterioration of CP in terms of GMRES iterations is *graceful*. It is remarkable that, on average, the linear system can be solved in 22 GMRES iterations when only four scenarios are used in the compressed matrix. This behavior again indicates that the computation of the second-stage variables in (23b) plays a key role in the performance of CP.

We emphasize on the efficiency gains obtained from parallelization with respect to the computation of the second-stage steps (23b). This step requires a factorization of all the block matrices K_s prior to calling the iterative linear solver. When the factorizations of the blocks are performed serially, the total solution time grows linearly with the number of scenarios. This can be observed from the block factorization times (denoted as $\theta_{factblock}$) reported in Table 9. In particular, the time spent in the factorization of the block matrices in the serial implementation (one processor) is a significant component of the total time. This overhead is eliminated using the parallel implementation (with almost perfect scaling).

5 Conclusions and future work

We have presented a preconditioning strategy for stochastic programs using clustering techniques. This inside-the-solver clustering strategy can be used as an alternative to (or in combination with) outside-the-solver scenario aggregation and clustering strategies. Practical features of performing inside-the-solver clustering is that no information on probability distributions is required and the effect of the data on the problem at hand is better captured. We have demonstrated that the preconditioners can be implemented in sparse form and dramatically reduce computational time compared to full factorizations of the KKT system. We have also demonstrated that the sparse form enables the solution of problems with large first-stage dimensionality that cannot be addressed with Schur decomposition. Scenario compression rates of up to 94 % have been observed in large problem instances. As part of future work, we will investigate the performance of the preconditioner in a nonlinear programming setting and we will investigate extensions to multi-stage stochastic programs.

Acknowledgments Victor M. Zavala acknowledges funding from the DOE Office of Science under the Early Career program. Carl Laird and Yankai Cao acknowledge support by the National Science Foundation CAREER Grant CBET #0955205. The authors thank Jacek Gondzio for providing feedback on a previous version of the manuscript.

References

1. Birge, J.: Aggregation bounds in stochastic linear programming. Math. Progr. **31**, 25–41 (1985)
2. Bishop, C.M., et al.: Pattern recognition and machine learning, vol. 4. Springer, New York (2006)
3. Byrd, R.H., Chin, G.M., Neveitt, W., Nocedal, J.: On the use of stochastic Hessian information in optimization methods for machine learning. SIAM J. Optim. **21**(3), 977–995 (2011)
4. Calafiore, G.C., Campi, M.C.: The scenario approach to robust control design. IEEE Trans. Autom. Control **51**(5), 742–753 (2006)
5. Casey, M.S., Sen, S.: The scenario generation algorithm for multistage stochastic linear programming. Math. Oper. Res. **30**(3), 615–631 (2005)
6. Chiang, N., Grothey, A.: Solving security constrained optimal power flow problems by a structure exploiting interior point method. Optim. Eng. pp. 1–23 (2012)
7. Colombo, M., Gondzio, J., Grothey, A.: A warm-start approach for large-scale stochastic linear programs. Math. Progr. **127**(2), 371–397 (2011)
8. de Oliveira, W.L., Sagastizábal, C., Penna, D., Maceira, M., Damázio, J.M.: Optimal scenario tree reduction for stochastic streamflows in power generation planning problems. Optim. Methods Softw. **25**(6), 917–936 (2010)
9. Dollar, H.S.: Constraint-style preconditioners for regularized saddle point problems. SIAM J. Matrix Anal. Appl. **29**(2), 672–684 (2007)
10. Dupačová, J., Gröwe-Kuska, N., Römisch, W.: Scenario reduction in stochastic programming. Math. Progr. **95**(3), 493–511 (2003)
11. Ferris, M.C., Munson, T.S.: Interior-point methods for massive support vector machines. SIAM J. Optim. **13**(3), 783–804 (2002)
12. Gondzio, J., Grothey, A.: Reoptimization with the primal-dual interior point method. SIAM J. Optim. **13**, 842–864 (2003)
13. Heitsch, H., Römisch, W.: Scenario tree reduction for multistage stochastic programs. Comput. Manag. Sci. **6**, 117–133 (2009)
14. Jung, J., Oleary, D.P., Tits, A.L.: Adaptive constraint reduction for training support vector machines. Electron. Trans. Numer. Anal. **31**, 156–177 (2008)
15. Kang, J., Cao, Y., Word, D.P., Laird, C.D.: An interior-point method for efficient solution of block-structured NLP problems using an implicit Schur-complement decomposition. Comput. Chem. Eng. (2014, in press)
16. Latorre, J.M., Cerisola, S., Ramos, A.: Clustering algorithms for scenario tree generation: application to natural hydro inflows. Eur. J. Oper. Res. **181**(3), 1339–1353 (2007)
17. Linderoth, J., Shapiro, A., Wright, S.: The empirical behavior of sampling methods for stochastic programming. Ann. Oper. Res. **142**(1), 215–241 (2006)
18. Lubin, M., Petra, C.G., Anitescu, M., Zavala, V.M.: Scalable stochastic optimization of complex energy systems. In: IEEE international conference for high performance computing, networking, storage and analysis (SC). pp. 1–10 (2011)
19. Mehrotra, S.: On the implementation of a primal-dual interior point method. SIAM J. Optim. **2**, 575–601 (1992)
20. Petra, C., Anitescu, M.: A preconditioning technique for Schur complement systems arising in stochastic optimization. Comput. Optim. Appl. **52**, 315–344 (2012)
21. Pritchard, G., Zakeri, G., Philpott, A.: A single-settlement, energy-only electric power market for unpredictable and intermittent participants. Oper. Res. **58**(4–part–2), 1210–1219 (2010)
22. Shetty, C.M., Taylor, R.W.: Solving large-scale linear programs by aggregation. Comput. Oper. Res. **14**(5), 385–393 (1987)
23. Szyld, D.B., Vogel, J.A.: Fqmr: a flexible quasi-minimal residual method with inexact preconditioning. SIAM J. Sci. Comput. **23**(2), 363–380 (2001)
24. Tits, A., Absil, P., Woessner, W.: Constraint reduction for linear programs with many inequality constraints. SIAM J. Optim. **17**(1), 119–146 (2006)

25. Zavala, V.M., Botterud, A., Constantinescu, E.M., Wang, J.: Computational and economic limitations of dispatch operations in the next-generation power grid. In: IEEE conference on innovative technologies for and efficient and reliable power supply (2010)
26. Zavala, V.M., Constantinescu, E.M., Krause, T., Anitescu, M.: On-line economic optimization of energy systems using weather forecast information. J. Process Control **19**(10), 1725–1736 (2009)
27. Zavala, V.M., Kim, K., Anitescu, M., Birge, J.: A stochastic market clearing formulation with consistent pricing properties. Technical Report ANL/MCS-P5110-0314, Argonne National Laboratory (2015)
28. Zipkin, P.H.: Bounds for row-aggregation in linear programming. Oper. Res. **28**(4), 903–916 (1980)