PCSDP 1.0 User's Guide

Ivan D. Ivanov

February 1, 2007

Introduction

PCSDP is a parallel software package based on CSDP5.0 [1] solver for semidefinite optimization (SDO) problems. The algorithm is an infeasible start primal–dual interior–point method using predictor–corrector strategy, see for details Helmberg et al. [3]. A more detailed description of the algorithm behind PCSDP can be found in [4]. PCSDP is written in ANSI C and is designed for distributed memory multiprocessor computers or (Beowulf[1]) computer clusters. PCSDP makes use of MPI, BLACS and ScaLAPACK[2] libraries for its parallel computations. Additionally the code uses optimized linear algebra routines from the LAPACK and BLAS libraries.

PCSDP similarly to CSDP works with general symmetric matrices and with matrices that have defined block diagonal structure. It handles constraint matrices with general sparse structure. The code exploits sparsity and rank-one structure in constraint matrices.

PCSDP is a stand alone solver program that can run on Linux or UNIX-like operating system. It requires as an input SDO instance written in the SDPA sparse format [2]. In case the rank-one feature is going to be used, then the problem should be in a modified SDPA sparse format. We describe this modified format later in this user guide.

This document gives brief description of how to use PCSDP solver. For detailed instructions on how to compile and install PCSDP see the INSTALL file in the main directory.

[1]http://www.beowulf.org/
[2]http://www.netlib.org/scalapack/

The Semidefinite Optimization Problem

Here we introduce in more details the notion of the semidefinite optimization. The goal of SDO is to maximize the linear objective

$$p := \mathbf{Tr}(CX),$$

which consists of the trace (denoted by '**Tr**') of the product of two symmetric $n \times n$ matrices, a given data matrix C and a variable matrix X, subject to a set of linear and nonlinear constraints. The linear constraints are:

$$\mathbf{Tr}(A_i X) = b_i, \quad i = 1, ..., m,$$

where the A_i's are given $n \times n$ symmetric matrices, and the b_i's are known scalars. The nonlinear constraint is that the matrix variable X must be symmetric positive semidefinite[3], denoted by $X \succeq 0$.

Hence, the semidefinite optimization problem has the following form, also called primal:

$$\begin{array}{lccccc} \text{(SDP)} & \max\limits_{X} & \mathbf{Tr}(CX) & & \\ & \text{s.t.} & \mathcal{A}(X) & = & b, \\ & & X & \succeq & 0, \end{array} \tag{1}$$

where the linear operator $\mathcal{A}(\cdot)$ is defined as

$$\mathcal{A}(X) = \begin{bmatrix} \mathbf{Tr}(A_1 X) \\ \mathbf{Tr}(A_2 X) \\ \vdots \\ \mathbf{Tr}(A_m X) \end{bmatrix}. \tag{2}$$

All constraint matrices A_i (*i.e.* are symmetric $n \times n$) for $i = 1, \cdots, m$, as is the matrix C. Thus n denotes the size of the matrix variables and m the number of equality constraints. The dual of the (SDP) problem is given by

$$\begin{array}{lccccc} \text{(SDD)} & \min\limits_{y,Z} & b^T y & & \\ \text{s.t.} & & \mathcal{A}^*(y) - Z & = & C, \\ & & Z & \succeq & 0, \end{array}$$

where Z is also $n \times n$ symmetric matrix (required to be positive semidefinite), and

$$\mathcal{A}^*(y) = \sum_{i}^{m} y_i A_i,$$

is the adjoint of $\mathcal{A}(\cdot)$ with respect to the usual trace inner product.

[3]A symmetric matrix X is positive semidefinite if all of its eigenvalues are nonnegative, or equivalently, if $s^T X s \geq 0$ for all $s \in \mathbb{R}^n$.

Some authors define this primal–dual pair in slightly different way, i.e, they define the primal problem as minimization and the dual as maximization. Some semidefinite optimization packages use this alternative way of defining the primal–dual pair.

The default termination criteria of PCSDP are

$$\begin{array}{rcl} \dfrac{\mid \mathbf{Tr}(CX) - b^T y \mid}{1 + |b^T y|} & < & 10^{-7}, \\ \dfrac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} & < & 10^{-7}, \\ \dfrac{\|\mathcal{A}^*(y) - Z - C\|_F}{1 + \|C\|_F} & < & 10^{-7}, \\ Z, X & \succeq & 0, \end{array} \tag{3}$$

where $\|\cdot\|_2$ denotes the Euclidean vector norm.

The duality gap for SDO is defined as

$$\mathbf{Tr}(CX) - b^T y = \mathbf{Tr}((Z - \textstyle\sum_{i=1}^m y_i A_i)X) - \sum_{i=1}^m y_i \mathbf{Tr}(A_i X) = \mathbf{Tr}(ZX).$$

For feasible solution (X, y, Z) the *weak duality* property holds in semidefinite optimization, namely

$$\mathbf{Tr}(ZX) \geq 0.$$

Hence, the first of these criteria insures that the duality gap is small. In practice, there are cases when the duality gap may even become negative. Because of this ambiguity, PCSDP uses the $\mathbf{Tr}(XZ)$ gap instead of the difference between the primal and dual objective functions. As an option in the parameters file `param.pcsdp` the user may switch to using the difference of the objective functions instead of the $\mathbf{Tr}(XZ)$ gap.

Since the matrices X and Z are considered to be positive definite, then their Cholesky factorizations can be computed. In this version of the solver we do not perform Cholesky of X and Z in parallel.

Similarly to CSDP5.0 solver, PCSDP provides information about the DIMACS error measures [5] in its output.

To check for primal infeasibility, PCSDP tests the inequality

$$\frac{-b^T y}{\|\mathcal{A}^T(y) - Z\|_F} > 1.0 \times 10^7. \tag{4}$$

If the problem is primal infeasible, PCSDP announces it in its output. It also gives a certificate: a dual solution with $b^T y = -1$, and $\|\mathcal{A}^T(y) - Z\|$ very small.

Similarly, a test for dual infeasibility is done by checking

$$\frac{\mathbf{Tr}(CX)}{\|\mathcal{A}(X)\|_2} > 1.0 \times 10^7. \tag{5}$$

In case a problem is dual infeasible, again PCSDP announces this in its output and returns a certificate: a primal solution with $\mathbf{Tr}(CX) = 1$, and $\|\mathcal{A}(X)\|$ small.

All tolerances for primal and dual feasibility as well as the duality gap can be adjusted by editing PCSDP's parameter file. See the following section on using the rank-one option.

Using the rank-one option

PCSDP, similarly to CSDP, works with block-structured matrices. For example, assume the constraint matrices $A_i, i = 1, ..., m$ consist of the matrix blocks $A_i^{(1)}, ..., A_i^{(k)}$ (we will omit further their subscript i for simplicity). Hence,

$$A_i := \begin{bmatrix} A^{(1)} & & & \mathbf{0} \\ & A^{(2)} & & \\ & & \ddots & \\ \mathbf{0} & & & A^{(k)} \end{bmatrix}.$$

Assume that the first j matrix blocks in each A_i have the form $A^{(p)} = a_p {a_p}^T$, for $p = 1, ..., j$. We will refer to this type of structure as *rank-one*. It appears in many large scale problems coming from combinatorial optimization problems.

Taking a full advantage of this construction inside PCSDP involves a direct access to vectors a_p. Therefore, the user should use in these cases a **modified SDPA sparse input** data format. The only difference with the original SDPA sparse format is that instead of specifying all elements of the rank-one matrix blocks $A^{(p)}$, we write only the vectors a_p $(p = 1, ..., j)$ in their main diagonal of the corresponding blocks.

Example: Suppose we have a problem in standard SDPA sparse format:

```
>cat test2-200.dat-s
201
3
100 100 -2
.
.
.
```

In this case

$$A_i = \begin{bmatrix} A^{(1)} & & \mathbf{0} \\ & A^{(2)} & \\ \mathbf{0} & & A^{(3)} \end{bmatrix}, \qquad i = 1, ..., 201,$$

where $A^{(1)}, A^{(2)}$ are symmetric 100×100 square matrix blocks, and $A^{(3)}$ is 2×2 diagonal matrix. Assume next that $A^{(1)}$ and $A^{(2)}$ are rank-one matries, i.e.,

$A^{(1)} = a_1{a_1}^T$ and $A^{(2)} = a_2{a_2}^T$. Normally, in standard SDPA format such matrices, as $A^{(1)}$ and $A^{(2)}$, are described as

$$A^{(1)} = \begin{bmatrix} A^{(1)}_{11} & A^{(1)}_{12} & A^{(1)}_{13} & A^{(1)}_{14} & A^{(1)}_{15} \\ 0 & A^{(1)}_{22} & A^{(1)}_{23} & A^{(1)}_{24} & A^{(1)}_{25} \\ 0 & 0 & A^{(1)}_{33} & A^{(1)}_{34} & A^{(1)}_{35} \\ 0 & 0 & 0 & A^{(1)}_{44} & A^{(1)}_{45} \\ 0 & 0 & 0 & 0 & A^{(1)}_{55} \end{bmatrix}.$$

To exploit efficiently in PCSDP such rank-one matrices, we introduced the **modified SDPA sparse** data format. It stores the rank-one blocks as diagonal matrices, namely

$$A^{(1)} = \begin{bmatrix} a^{(1)}_1 & & & & \mathbf{0} \\ & a^{(1)}_2 & & & \\ & & a^{(1)}_3 & & \\ & & & a^{(1)}_4 & \\ \mathbf{0} & & & & a^{(1)}_5 \end{bmatrix}. \tag{6}$$

In order to avoid confusion about the correct type of the blocks, users have to pay attention of the following:

1) Each rank-one block should be written as in (6) and declared **always** in the data file as a matrix type (in the example above '100 100 -2', not as '-100 -100 -2');

2) The parameter *rank1* in 'param.csdp' file must be set to j, i.e., the total number of rank-one blocks. In the example above, this parameter should be set to 'rank1=2'.

Note that using *rank1* parameter with the positive value j, we state that **the first** j blocks in $A_i, i = 1, ..., m$ are rank-one. If this is not the case, then the SDPA sparse input file will be interpreted in a wrong way. See the following section for a description of all parameters in the 'param.pcsdp' file.

Using the solver

PCSDP solves SDO problems that have been written in the SDPA sparse format. The program is started on the compute nodes of the cluster using the Sun Grid Engine (SGE[4]) batch queueing system or the 'prun' user interface.

The usage with prun is as follows:

[4]http://docs.sun.com/app/docs/coll/1017.3

```
 prun -v -np N [-o <output file>] pcsdp <problem file> ...
                               ... [<final solution>] [<initial solution>]
```

where `-v` option reports host allocation, `-np` specifies the number `N` of CPUs to be used, `-o` writes the output from each of the parallel processes into a separate `<output file>`, `<problem file>` is the name of a file containing the SDO problem in SDPA sparse format, `final solution` is the optional name of a file in which to save the final solution, and `initial solution` is the optional name of a file from which to take the initial solution.

The following example shows how PCSDP would be used to solve a problem, on 4 CPUs and writes the solution in `outputfile`.

```
$ prun -v -np 4 -o solutionfile pcsdp testprob/theta5.dat-s

$ cat outputfile.0
 Iter:  0 Ap: 0.00e+00 Pobj:  3.6611652e+05 Ad: 0.00e+00 Dobj:  0.0000000e+00
 Iter:  1 Ap: 8.96e-01 Pobj:  6.2953741e+05 Ad: 8.51e-01 Dobj:  1.4900931e+02
 Iter:  2 Ap: 8.44e-01 Pobj:  2.8720606e+04 Ad: 1.00e+00 Dobj:  1.9199634e+02
 Iter:  3 Ap: 8.66e-01 Pobj:  1.4970884e+03 Ad: 1.00e+00 Dobj:  2.0092645e+02
 Iter:  4 Ap: 8.77e-01 Pobj:  7.8171197e+01 Ad: 1.00e+00 Dobj:  2.1007836e+02
 Iter:  5 Ap: 9.58e-01 Pobj:  1.6149600e+01 Ad: 1.00e+00 Dobj:  2.1641386e+02
 Iter:  6 Ap: 1.00e+00 Pobj:  8.4599603e+00 Ad: 1.00e+00 Dobj:  1.7625506e+02
 Iter:  7 Ap: 1.00e+00 Pobj:  2.2888970e+01 Ad: 1.00e+00 Dobj:  1.1477064e+02
 Iter:  8 Ap: 1.00e+00 Pobj:  3.8407284e+01 Ad: 1.00e+00 Dobj:  7.0032152e+01
 Iter:  9 Ap: 1.00e+00 Pobj:  5.1479243e+01 Ad: 1.00e+00 Dobj:  6.1392496e+01
 Iter: 10 Ap: 1.00e+00 Pobj:  5.5259658e+01 Ad: 1.00e+00 Dobj:  5.7848116e+01
 Iter: 11 Ap: 1.00e+00 Pobj:  5.6900184e+01 Ad: 1.00e+00 Dobj:  5.7305727e+01
 Iter: 12 Ap: 1.00e+00 Pobj:  5.7198444e+01 Ad: 1.00e+00 Dobj:  5.7239034e+01
 Iter: 13 Ap: 1.00e+00 Pobj:  5.7229356e+01 Ad: 1.00e+00 Dobj:  5.7232858e+01
 Iter: 14 Ap: 9.92e-01 Pobj:  5.7232049e+01 Ad: 1.00e+00 Dobj:  5.7232350e+01
 Iter: 15 Ap: 9.67e-01 Pobj:  5.7232285e+01 Ad: 1.00e+00 Dobj:  5.7232310e+01
 Iter: 16 Ap: 8.68e-01 Pobj:  5.7232304e+01 Ad: 9.58e-01 Dobj:  5.7232307e+01
 0 Success: SDP solved
 Primal objective value: 5.7232304e+01
 Dual objective value: 5.7232307e+01
 Relative primal infeasibility: 4.29e-15
 Relative dual infeasibility: 2.15e-10
 Real Relative Gap: 2.67e-08
 XZ Relative Gap: 2.68e-08
 DIMACS error measures: 4.29e-15 0.00e+00 2.70e-08 0.00e+00 2.67e-08 2.68e-08
```

Each iteration is reported via one line of output and includes: the iteration number, primal step size `(Ap)`, primal objective value `(Pobj)`, dual step size `(Ad)`, and dual objective value `(Dobj)`.

At the end, PCSDP displays the primal and dual optimal objective values, the XZ duality gap, the actual duality gap, the relative primal and dual infeasibility in the optimal solution. The last line gives the DIMACS error measures.

PCSDP uses a parameter file named "param.pcsdp", that should be in the current directory as the binary. IN case no such file exists, then default values for all of PCSDP's parameters are used. If there exists a parameter file, then PCSDP reads and uses the parameter values from "param.pcsdp". An example file containing the default parameter values is given below.

```
axtol=1.0e-7
atytol=1.0e-7
objtol=1.0e-7
pinftol=1.0e7
dinftol=1.0e7
maxiter=100
minstepfrac=0.90
maxstepfrac=0.97
minstepp=1.0e-8
minstepd=1.0e-8
usexzgap=1
tweakgap=0
affine=0
printlevel=1
perturbobj=1
fastmode=0
rank1=0
```

The first three parameters, `axtol, atytol,` and `objtol` are the tolerances for primal feasibility, dual feasibility, and relative duality gap. The parameters `pinftol` and `dinftol` are tolerances used in determining primal and dual infeasibility.

The maximum number of iterations PCSDP as a stopping criteria is given by the `maxiter`. The `minstepfrac` and `maxstepfrac` parameters determine the distance to the edge of the feasible region. In case the primal or the dual step is shorter than `minstepp` or `minstepd`, then PCSDP declares a line search failure.

By default `usexzgap=0`, so PCSDP will use the objective function duality gap instead of the $\mathrm{tr}(XZ)$ gap. In case `tweakgap` is set to 1, and `usexzgap` is set to 0, then PCSDP will attempt to "fix" cases of negative duality gaps. If parameter `affine` is set to 1, then PCSDP will take only primal–dual affine steps and not make use of the barrier term. This might be useful for problems with no feasible solutions that are strictly in the interior of the cone of semidefinite matrices.

The `printlevel` parameter determines the amount of debugging information printed as output. If `printlevel=0`, no output is printed and `printlevel=1` is for normal output. Higher values of `printlevel` will generate more debugging information.

The `perturbobj` parameter determines whether the objective function will be perturbed to help deal with problems that have unbounded optimal solution sets. If `perturbobj=0`, then the objective will not be perturbed. If `perturbobj=1`, then the objective function will be perturbed by a default amount. Larger values of `perturbobj` (e.g. 100.0 or more) increase the size of the perturbation. This is helpful in solving some difficult problems. The `fastmode` parameter determines whether or not PCSDP will skip certain time consuming operations that slightly improve the accuracy of the solutions. When `fastmode=1`, then PCSDP may be faster, but also probably less accurate.

The `rank1` parameter is used only in cases when an explicit rank-one structure is present in the constraints. The default value is 0, which means a normal mode of operation with input the standard SDPA file format. If `rank1` is set to 1, then PCSDP will treat the first matrix block of all constraint matrices as rank-one. Similarly, `rank1=2` means that the first 2 matrix blocks of all constraint matrices are rank-one. In order to use this feature, one should use the *modified SDPA sparse format*, explained in the previous section.

References

[1] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods & Software*, 11-2(1-4):613 – 623, 1999.

[2] K. Fujisawa, M. Kojima, K. Nakata, and M. Yamashita. SDPA (semidefinite programming algorithm) users manual - version 6.00. Technical Report B–308, Tokyo Institute of Technology, 1995.

[3] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342 – 361, May 1996.

[4] I.D. Ivanov and E. de Klerk. Parallel implementation of a semidefinite programming solver based on csdp in a distributed memory cluster. Technical report, 2007.

[5] H. D. Mittelmann. An independent benchmarking of SDP and SOCP solvers. *Mathematical Programming*, 95(2):407 – 430, February 2003.