# A Semidefinite Programming Approach for the Nearest Correlation Matrix Problem

Miguel F. Anjos* Nicholas J. Higham†
Pawoumodom L. Takouda ‡ Henry Wolkowicz§

September 16, 2003

University of Waterloo
Department of Combinatorics & Optimization
Waterloo, Ontario N2L 3G1, Canada
Preliminary Research Report 2003

**Keywords:** Closest Correlation Matrix, Semidefinite Programming, Large Sparse Problems.

**Abstract**

The nearest correlation matrix problem is to find a positive semidefinite matrix with unit diagonal that is nearest in the Frobenius norm to a given symmetric matrix $A$. This problem can be formulated as an optimization problem with a quadratic objective function and semidefinite programming constraints. Using such a formulation, we derive and test a primal-dual interior-exterior-point (p-d i-e-p) algorithm designed specifically for robustness and handling the case where $A$ is

*Faculty of Mathematical Studies, University of Southampton, Southampton, SO17 1BJ, UK. Email anjos@stanfordalumni.org

†Department of Mathematics, University of Manchester, Manchester, M13 9PL, UK. Email higham@ma.man.ac.uk

‡Laboratoire MIP, Université Paul Sabatier, 118 rte de Narbonne 31062 Toulouse Cedex 4, France. Email takouda@mip.ups-tlse.fr

§Research supported by The Natural Sciences and Engineering Research Council of Canada. Email hwolkowica@uwaterloo.ca

[0] URL for paper: http://orion.math.uwaterloo.ca/~hwolkowi/henry/reports/ABSTRACTS.html

sparse. The algorithm is novel in several respects. First, instead of solving the so-called normal equations to obtain the search direction at each iteration, our algorithm eliminates the linear feasibility equations from the start. This means that we maintain exact primal and dual feasibility during the course of the algorithm, and that we use a single bilinear equation to linearize for the search direction at each iteration. Second, the search direction is found using an inexact Gauss-Newton method rather than a Newton method on a symmetrized system, and is computed using a preconditioned conjugate-gradient-type method. We consider two types of preconditioner, an optimal diagonal preconditioner and a block diagonal preconditioner obtained from a partial Cholesky factorization. Finally, once the current iterate is sufficiently close to the optimal solution, we apply a crossover technique that sets the barrier parameter to 0 and does not maintain interiority of the iterates. The result is a robust algorithm with asymptotic quadratic convergence and the ability to handle *warm starts* simply. Preliminary computational results illustrate the robustness of the algorithm and show that sparsity can be successfully exploited.

# Contents

# 1 Introduction

We are interested in finding the nearest correlation matrix to a given symmetric matrix $A \in \mathcal{S}^n$, that is, solving the problem

$$\mu^* = \min \frac{1}{2}\|A - X\|_F^2 \quad \text{subject to} \quad \operatorname{diag} X = e,\ X \in \mathcal{S}^n,\ X \succeq 0. \tag{1.1}$$

Here, $e$ is the vector of ones, $\mathcal{S}^n$ is the space of symmetric matrices in $\mathbb{R}^{n\times n}$, $X \succeq 0$ denotes positive semidefiniteness, $\operatorname{diag} X$ is the vector formed from the diagonal of $X$, and $\|A\|_F = \operatorname{trace}(A^TA)^{1/2}$ is the Frobenius norm. This problem is of interest in statistical applications in which a computed correlation matrix is found not to be a correlation matrix, the underlying causes of which can include measurement errors, rounding errors, and missing data. A number of statistical software packages discuss this situation in their documentation (see, for example, `http://www.ssicentral.com/lisrel/posdef.htm`). A particular application is in the finance industry, where complete stock data is often not available over a given period and currently used techniques for dealing with missing data can result in computed correlation matrices having negative eigenvalues [7].

Higham [7] develops an alternating projections method for solving a more general version of (1.1) with a weighted Frobenius norm. The method is capable of solving problems arising in practice, where $n$ is currently in the low thousands. However, its convergence is linear, and hence possibly very slow (although high accuracy is not required in the finance application, where the original data is accurate only to a few significant figures).

The nearest correlation matrix problem is related to several other problems in the research literature. In particular, the structure of the set of correlation matrices (or elliptope), has been extensively studied and much of its structure is well documented in the book [3]. Also, this problem is a special case of the more general problem of projecting a given point, in finite-dimensional Euclidean space, onto the intersection of a closed convex cone with an affine subspace. The formulation of this problem, and of the correlation matrix problem in particular, as a convex differentiable optimization problem has been very recently proposed in [9].

In this paper we solve (1.1) using semidefinite programming (SDP). In particular, we provide a stable algorithm that is particularly effective when the given matrix $A$ is large and sparse. Our algorithm specifically exploits the fact that the nearest correlation matrix problem involves optimizing over the cone of positive semidefinite matrices, and the ideas we present are promising for future applications in this area.

Our primal-dual interior-exterior-point (p-d i-e-p) approach is novel in several respects. First, the system of optimality conditions that we use is different from, and no larger than, those commonly used in the SDP literature. This is because instead of solving the so-called normal equations to obtain the search direction at each iteration, we substitute the primal and dual feasibility equations into the complementary slackness equation. This means that we maintain exact primal and dual feasibility during the course of the algorithm, we have a true primal-dual path following algorithm, and we have a single bilinear equation, in $n(n+1)/2$ variables, to linearize for the search direction at each iteration; see (2.4) below. Second, we use an inexact Gauss-Newton method on this bilinear equation, employing a preconditioned conjugate-gradient (PCG) method to solve the linearized system, $F'_\mu d = -F_\mu$, for the search direction $d$. These two innovations avoid the ill-conditioning that arises in current SDP algorithms.

In the PCG method, we consider two types of preconditioner: an optimal diagonal preconditioner and a block diagonal preconditioner obtained from a partial Cholesky factorization. The latter preconditioner is inspired by the structure of the operator $F'_\mu$, which naturally divides into two blocks.

We exploit the fact that the linearized system is full rank at optimality by applying a crossover technique. The idea is that once we are close enough to the optimum, we can switch to an affine scaling approach (that is, we set the barrier parameter to 0 and take steps of length one) and we do not maintain interiority. This crossover technique results in q-quadratic convergence [12]. In addition, during the complete course of the algorithm we take long steps to the boundary of positive semidefiniteness without backtracking to ensure sufficient definiteness.

Our algorithm is robust and benefits from asymptotic quadratic convergence. The robustness allow us to handle *warm starts* simply. (Since we do not have to guarantee positive definiteness after the crossover, when we are at or close to the optimum we can perturb the data and restart the algorithm from the current point.) Our computational results show that the sparsity in $A$ and the optimal matrix $X$ is fully exploited; in particular, the cpu time is

correlated with the density in $X$. (Though $A$ is not generally sparse, we use perturbations to originally solve problems with sparse $A$ and then use warm starts while increasing the number of nonzeros.)

## 1.1 Mixed-Cone Formulation

A direct approach for the nearest correlation matrix problem is obtained by formulating it as a mixed SDP and second-order (or Lorentz) cone problem:

$$\begin{array}{rlll} \sqrt{2\mu^*} = & \min & \alpha & \\ & \text{s.t.} & \operatorname{diag} X = e & \\ & & Y + X = A, \ \|Y\|_F \leq \alpha & \\ & & X, Y \in \mathcal{S}^n \, , X \succeq 0. & \end{array} \tag{1.2}$$

Several public domain software packages can solve (1.2). Many of them can be accessed via NEOS [4] at `http://www-neos.mcs.anl.gov/` (see also C. Helmberg's SDP page at `http://www.zib.de/helmberg/semidef.html`). The main work per iteration for solving this problem is to form and solve the (usually dense) normal equations for the (Newton) search direction. The size of this system is determined by the $n + \binom{n+1}{2}$ equality constraints, and thus it is order $n^2$. There are many complications when forming and solving this system, since it is usually ill-conditioned at the solution. Computational tests comparing this approach to our specialized SDP algorithm are presented in Section 4.

Note that our approach in this paper also involves solving a linear system of order $n^2$. (We solve a least squares problem of size $n^2 \times n(n+1)/2$ using PCG.) However, our system is sparse, does not need to be formed explicitly, and has full column rank in the limit.

## 1.2 Notation

For a general rectangular matrix $M = [m_1 \ m_2 \ \ldots \ m_n] \in \mathbb{R}^{m \times n}$,

$$v = \operatorname{vec}(M) := \begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{pmatrix} \in \mathbb{R}^{mn}$$

forms a vector from its columns. The inverse mapping, $\operatorname{vec}^{-1}$, and the adjoint mapping, $\operatorname{vec}^*$, are given by $\operatorname{Mat} = \operatorname{vec}^{-1} = \operatorname{vec}^*$, the adjoint formula

following from $\langle \mathrm{vec}\,(M), u\rangle = \langle M, \mathrm{vec}^{\,*}(u)\rangle$. We use the *trace inner product* $\langle M, N\rangle := \mathrm{trace}\, M^T N$, which induces the Frobenius norm. With this inner product, Mat (and vec) is an isometry.

We define several operators on vectors and matrices. Let $x = \mathrm{us2vec}\, X \in \mathbb{R}^{\binom{n}{2}}$ be $\sqrt{2}$ times the vector obtained columnwise from the strictly upper triangular part of $X$. Here $\binom{n}{2} = n(n-1)/2$ and $\sqrt{2}$ guarantees that the mapping is an isometry. (Define vectors $a, s$ similarly, $a = \mathrm{us2vec}\, A, s = \mathrm{us2vec}\, S$.) Let $\mathrm{us2Mat} := \mathrm{us2vec}^{-1}$ denote the inverse mapping into $\mathcal{S}^n$, i.e., the one-one mapping between $\mathbb{R}^{\binom{n}{2}}$ and the matrices with zero diagonal in $\mathcal{S}^n$. The adjoint operator $\mathrm{us2Mat}^{\,*} = \mathrm{us2vec}$, since

$$\begin{aligned} \langle \mathrm{us2Mat}\,(v), S\rangle = \mathrm{trace}\, \mathrm{us2Mat}\,(v) S &= \mathrm{trace}\, \mathrm{us2Mat}\,(v) \mathrm{offDiag}\,(S) \\ &= v^T \mathrm{us2vec}\,(S) = \langle \mathrm{us2vec}\,(S), v\rangle\,. \end{aligned}$$

Here,

$$\mathrm{offDiag}\,(S) := S - \mathrm{Diag}\,(\mathrm{diag}\,(S)),$$

where $\mathrm{diag}\,(S)$ denotes the diagonal of $S$ and $\mathrm{Diag}\,(v)$ is the adjoint operator, i.e., the diagonal matrix with diagonal elements from the vector $v$. Therefore

$$\mathrm{us2Mat}\,\mathrm{us2Mat}^{\,*}(S) = \mathrm{offDiag}\,(S) = \mathrm{offDiag}^{\,*}(S),$$

is the orthogonal projection onto the subspace of matrices with zero diagonal.

Our algorithm uses the following operators which map $\mathcal{S}^n \to \mathcal{M}^n$, the space of $n \times n$ matrices. Let

$$X := A + \mathrm{us2Mat}\,(s) + I, \quad S^y := \mathrm{us2Mat}\,(s) + \mathrm{Diag}\,(y),$$

for appropriate vectors, $s$ and $y$. Define the linear operators

$$\boxed{\mathcal{X}_u(\cdot) := X\mathrm{us2Mat}\,(\cdot);\ \mathcal{X}_d(\cdot) := X\mathrm{Diag}\,(\cdot);\ \mathcal{S}(\cdot) := \mathrm{us2Mat}\,(\cdot)S^y.}$$

Thus

$$\mathcal{X}_u : \mathbb{R}^{\binom{n}{2}} \to \mathcal{M}^n; \quad \mathcal{X}_d : \mathbb{R}^n \to \mathcal{M}^n; \quad \mathcal{S} : \mathbb{R}^{\binom{n}{2}} \to \mathcal{M}^n\,.$$

The calculations for the various adjoints are given in Appendix A.

**SUMMARY**

$$
\begin{aligned}
\mathcal{X}_d(\cdot) &= X\mathrm{Diag}\,(\cdot)\\
\mathcal{X}_u(\cdot) &= X\mathrm{us2Mat}\,(\cdot)\\
\mathcal{S}(\cdot) &= \mathrm{us2Mat}\,(\cdot)S^y\\
\mathcal{X}_d{}^*(W) &= (W\circ X)^T e = \mathrm{diag}\,(W^TX)\\
\mathcal{X}_u{}^*(W) &= \tfrac{1}{2}\mathrm{us2vec}\,(XW+W^TX)\\
\mathcal{S}^*(W) &= \tfrac{1}{2}\mathrm{us2vec}\,(WS^y+S^yW^T)\\
\mathcal{X}_d{}^*\mathcal{X}_d(z) &= \mathrm{diag}\,(\mathrm{Diag}\,(z)X^2)\\
\mathcal{X}_d{}^*\mathcal{X}_u(v) &= \mathrm{diag}\ (\mathrm{us2Mat}\,(v)X^2)\\
\mathcal{X}_d{}^*\mathcal{S}(v) &= \mathrm{diag}\ (S^y\mathrm{us2Mat}\,(v)X)\\
\mathcal{X}_u{}^*\mathcal{X}_d(z) &= \tfrac{1}{2}\mathrm{us2vec}\ (X^2\mathrm{Diag}\,(z)+\mathrm{Diag}\,(z)X^2)\\
\mathcal{X}_u{}^*\mathcal{X}_u(v) &= \tfrac{1}{2}\mathrm{us2vec}\ (X^2\mathrm{us2Mat}\,(v)+\mathrm{us2Mat}\,(v)X^2)\\
\mathcal{X}_u{}^*\mathcal{S}(v) &= \tfrac{1}{2}\mathrm{us2vec}\ (X\mathrm{us2Mat}\,(v)S^y+S^y\mathrm{us2Mat}\,(v)X)\\
\mathcal{S}^*\mathcal{X}_u(v) &= \tfrac{1}{2}\mathrm{us2vec}\ (X\mathrm{us2Mat}\,(v)S^y+S^y\mathrm{us2Mat}\,(v)X)\\
\mathcal{S}^*\mathcal{X}_d(z) &= \tfrac{1}{2}\mathrm{us2vec}\ (X\mathrm{Diag}\,(z)S^y+S^y\mathrm{Diag}\,(z)X)\\
\mathcal{S}^*\mathcal{S}(v) &= \tfrac{1}{2}\mathrm{us2vec}\ (\mathrm{us2Mat}\,(v)(S^y)^2+(S^y)^2\mathrm{us2Mat}\,(v))
\end{aligned}
$$

# 2 Duality and Optimality Conditions

The diagonal of the variable $X$ in (1.1) is constant and so without loss of generality we assume that $\mathrm{diag}\,(A)=0$. Then (using $X=\mathrm{us2Mat}\,(x)+I$) an equivalent problem to (1.1) is:

$$\mu^* := \min \frac{1}{2}\|x-a\|_2^2 \quad \text{subject to} \quad \mathrm{us2Mat}\,(x)+I\succeq 0,\ x\in\mathbb{R}^{\binom{n}{2}}. \tag{2.1}$$

To obtain optimality conditions we use a dual problem. Slater's condition (strict feasibility) holds for (2.1), which implies that we have strong duality with the Lagrangian dual

$$\mu^*=\nu^* := \max_{S\succeq 0}\min_x \frac{1}{2}||x-a||^2 - \mathrm{trace}\,S(\mathrm{us2Mat}\,(x)+I).$$

We change this to the Wolfe dual by noting that the inner problem is a convex unconstrained problem and its optimal solutions are characterized by stationarity:

$$0=(x-a)-\mathrm{us2Mat}^*(S)=(x-a)-\mathrm{us2vec}\,(S), \tag{2.2}$$

since trace $S(\text{us2Mat}\,(x)+I) = x^T\,\text{us2vec}\,(S) + \text{trace}\,S$. Thus we obtain the equivalent dual problem:

$$\begin{array}{rlll} \mu^* = & \max & \frac{1}{2}||x-a||^2 - \text{trace}\,S(\text{us2Mat}\,(x)+I) & \\ & \text{subject to} & x - \text{us2vec}\,(S) = a & (2.3) \\ & & S \succeq 0. & \end{array}$$

Since Slater's condition is satisfied for both primal and dual programs we get the following optimality conditions.

**Theorem 2.1** *The optimal values $\mu^* = \nu^*$ and the primal-dual pair $x, (y,s)$ are optimal for* (2.1) *and* (2.3) *if and only if*

$$\begin{array}{rcll} X & := & \text{us2Mat}\,(x) + I \succeq 0 & \textit{(primal feasibility)} \\ x & = & a+s, \quad S^y := \text{us2Mat}\,(s) + \text{Diag}\,(y) \succeq 0 & \textit{(dual feasibility)} \\ XS^y & = & 0 & \textit{(complementary slackness)} \end{array}$$

■

For our p-d i-e-p algorithm we use

$$XS^y = \mu I \quad \text{perturbed complementary slackness.}$$

Furthermore, we can substitute the primal and dual feasibility equations into the perturbed complementary slackness equation and obtain a *single bilinear equation* in $s$ and $y$ that characterizes optimality for the perturbed log-barrier problem:

$$\begin{array}{rcl} & & F_\mu(s,y) : \mathbb{R}^{\binom{n+1}{2}} \to \mathcal{M}^n\,. \\ F_\mu(s,y) : & = & [A + \text{us2Mat}\,(s) + I]\,[\text{us2Mat}\,(s) + \text{Diag}\,(y)] - \mu I = 0, \end{array} \tag{2.4}$$

Note that the original closest correlation matrix problem has $\binom{n+1}{2}$ variables, $n$ equality constraints (on the diagonal of $X$) and the semidefiniteness constraint on $X$. Therefore, the dual problem has $n + \binom{n+1}{2}$ variables. Therefore, dual based algorithms do not reduce the size of the problem and standard primal-dual based algorithms have $n + 2\binom{n+1}{2}$ variables.

Viewing (2.4) as an overdetermined system of nonlinear equations, we solve it using an inexact Gauss-Newton method. Linearizing, we obtain a

linear system for the search direction $\Delta v = \binom{\Delta s}{\Delta y}$, where $v = \binom{s}{y}$:

$$\begin{aligned} -F_\mu(s,y) &= F'_\mu(s,y)\Delta v \\ &= [A + \text{us2Mat}\,(s) + I]\,(\text{us2Mat}\,(\Delta s) + \text{Diag}\,(\Delta y)) \\ &\qquad +\text{us2Mat}\,(\Delta s)S^y \\ &= (\mathcal{X}_u + \mathcal{S})\,(\Delta s) + \mathcal{X}_d(\Delta y). \end{aligned} \tag{2.5}$$

(where the vec is understood). This is a linear, full rank, overdetermined system. We use its least squares solution as the search direction $\Delta v$ in our algorithm. This solution is found in two ways: first, we use PCG with both diagonal and block diagonal preconditioners; second, we use a Lanczos method for the smallest eigenvalue of a parametrized system in order to find a regularized least squares solution. Note that $\Delta s \in \mathbb{R}^{\binom{n}{2}}$, but the cost of evaluating $(\mathcal{X}_u + \mathcal{S})\,(\Delta s)$ is (without considering any sparsity) equivalent to one matrix-matrix multiplication.

# 3 Primal-Dual Interior-Exterior-Point Algorithm

We can use equation (2.4) to develop a primal-dual interior-exterior-point (p-d i-e-p) algorithm, i.e., we linearize to find the search direction (assuming that we start feasible) using a linear least squares problem.

## 3.1 Framework

The p-d i-e-p framework that we use is different in several ways from the common framework for both linear and semidefinite programming, see e.g. [13, 10]. We have eliminated, in advance, the primal and dual linear feasibility equations. We work with an overdetermined nonlinear system rather than a square symmetrized system; thus we use an (inexact) Gauss-Newton approach [8]. We include a centering parameter $\sigma_k$ (instead of the customary predictor-corrector approach). We enforce positive semidefiniteness rather than definiteness in the steplengths. In addition, once we are *close enough* to the optimum we set the centering parameter $\sigma$ to zero (crossover step) and we no longer enforce interiority, i.e. we allow negative eigenvalues. This allows for (fast) asymptotic quadratic convergence.

At each iteration, we have available the iterate $v = \begin{pmatrix} s \\ y \end{pmatrix}$ and we find a new iterate by taking a step in the (inexact) Gauss-Newton search direction $\Delta v$. Up until the crossover, we ensure that the new iterate $v + \alpha \Delta v$ results in both $X, S^y$ sufficiently positive definite; then, we take $\alpha = 1$ after the crossover. By our construction, the iterates maintain both primal and dual feasibility.

We let $\mathcal{F}^0$ denote the set of strictly feasible primal-dual points; $F'$ denotes the derivative of the function of optimality conditions.

**Algorithm 3.1** *(p-d i-e-p framework:)*

- **Initialization:**

  •• **Input data:** *a real symmetric* $n \times n$ *matrix* $A$ *(set* $\mathrm{diag}\,(A) = 0$*)*

  •• **Positive tolerances:** $\epsilon_1$ *(stopping),* $\epsilon_2$ *(lss accuracy),* $\epsilon_3$ *(crossover).*

  •• **Find initial strictly feasible points:**

  *both* $S^0, X^0 := (\mathrm{offDiag}\,(S + A) + I) \succ 0$*;* $\mu$ *small*

  •• **Set initial parameters:**

  $$\mathrm{gap} = \mathrm{trace}\, S^0 X^0; \;\; \mu = \mathrm{gap}\,/n; \;\; \mathrm{objval} = 0.5||X^0 - A||_F^2; \;\; k = 0.$$

- **while** $\min\{\frac{\mathrm{gap}}{\mathrm{objval}+1}, \mathrm{objval}\} > \epsilon_1$

  •• **solve lss for search direction** *(accuracy* $\epsilon_2 \min\{\mu, 1\}$

  $$F'_{\sigma\mu}(v^k)\left(\Delta v^k\right) = -F_{\sigma\mu}(v^k),$$

  *where* $\sigma_k$ *centering,* $\mu_k = \frac{1}{n}\mathrm{trace}\, S^k(\mathrm{offDiag}\,(S^{k+1} + A) + I)$

  $$S^{k+1} = S^k + \alpha_k \Delta S^k, \quad \alpha_k > 0,$$

  *so that both* $S^{k+1}, \mathrm{offDiag}\,(S^{k+1} + A) + I \succeq 0$

  *(*$\alpha_k = 1$ *after the crossover.)*

  •• **update**

  $$k \leftarrow k + 1 \quad \textit{and then}$$

  $$\mu_k, \sigma_k \quad \left(\textit{set } \sigma_k = 0 \textit{ if } \min\{\frac{\mathrm{gap}}{\mathrm{objval}+1}, \mathrm{objval}\} < \epsilon_3\right)$$

- **end (while).**
- **On completion:** $X \approx \mathrm{us2Mat}\,(s) + A + I$

## 3.2 Preconditioning

Preconditioning is essential for efficient solution of the least squares problem (2.5). We find operators $P_s, P_y$ and find the least squares solution of

$$(\mathcal{X}_u + \mathcal{S})\, P_s^{-1}(\widehat{\Delta s}) + \mathcal{X}_d P_y^{-1}(\widehat{\Delta y}) = -F_\mu(s,y)$$

where

$$\widehat{\Delta s} = P_s(\Delta s), \quad \widehat{\Delta y} = P_y(\Delta y).$$

The inverses are not found explicitly. The two operators $P_s, P_y$ have simple structure so that the linear systems can be solved efficiently.

### 3.2.1 Diagonal Preconditioning

Optimal diagonal scaling has been studied in, e.g., [6, Sect. 10.5], and [2, Prop. 2.1(v)]. In the latter reference, it was shown that for a full rank matrix $A \in \mathbb{R}^{m\times n}$, $m \geq n$, and using the condition number $\omega(K) := n^{-1}\mathrm{trace}\,(K)/\det(K)^{1/n}$, the optimal scaling, i.e. the solution of the optimization problem

$$\min \omega((AD)^T(AD)) \quad \text{subject to } D \text{ positive diagonal matrix,} \tag{3.1}$$

is given by $d_{ii} = 1/\|A_{:i}\|_2, i = 1,\ldots,n$.

Therefore, the two operators $P_s, P_y$ are diagonal and are evaluated using the columns of the operator $F'_\mu(s,y)$. The columns are ordered using $k = 1,2,\ldots$ where $k$ represents $(i,j)$, $1 \leq i < j \leq n$ for the strictly upper triangular part of $S$ and then $i = 1,\ldots,n$ for the elements of $y$. Let $X = A + \mathrm{us2Mat}\,(s) + I$ and $S = \mathrm{us2Mat}\,(s) + \mathrm{Diag}\,(y)$.

For the first two operators in the left part of $F'$, we get:

$$\begin{aligned}
\mathcal{X}_u(e_k) &= X\,\mathrm{us2Mat}\,(e_k) \\
&= \tfrac{1}{\sqrt{2}} X\left(e_i e_j^T + e_j e_i^T\right) \\
&= \tfrac{1}{\sqrt{2}}\left(X_{:i} e_j^T + X_{:j} e_i^T\right);
\end{aligned}$$

$$\begin{aligned}
\mathcal{S}(e_k) &= \mathrm{us2Mat}\,(e_k)(S + \mathrm{Diag}\,(y)) \\
&= \tfrac{1}{\sqrt{2}}\left(e_i e_j^T + e_j e_i^T\right)(S + \mathrm{Diag}\,(y)) \\
&= \tfrac{1}{\sqrt{2}}\left\{(e_i(S + \mathrm{Diag}\,(y))_{j:} + e_j(S + \mathrm{Diag}\,(y))_{i:})\right\}.
\end{aligned}$$

Therefore

$$\begin{aligned}\|(\mathcal{X}_u+\mathcal{S})(e_k)\|_F^2 &= \tfrac{1}{2}\{\|(S+\mathrm{Diag}\,(y))_{:i}\|^2+\|(S+\mathrm{Diag}\,(y))_{:j}\|^2+\\ &\quad \|X_{:i}\|^2+\|X_{:j}\|^2+2(S+\mathrm{Diag}\,(y))_{jj}X_{ii}\\ &\quad +4(S+\mathrm{Diag}\,(y))_{ji}X_{ij}+2(S+\mathrm{Diag}\,(y))_{ii}X_{jj}\}\,.\end{aligned} \tag{3.2}$$

For this calculation we need the three Hadamard products $X\circ X, (S+\mathrm{Diag}\,(y))\circ(S+\mathrm{Diag}\,(y)), (S+\mathrm{Diag}\,(y))\circ X$ and the *vector* Kronecker product $\mathrm{Diag}\,((S+\mathrm{Diag}\,(y)))\otimes\mathrm{Diag}\,(X)$.

For the last operator in the right part of $F'$, we get

$$\mathcal{X}_d(e_i)=X\mathrm{Diag}\,(e_i).$$

Therefore

$$\|\mathcal{X}_d(e_i)\|_F^2=||X_{i,:}||^2. \tag{3.3}$$

The diagonal preconditioners are inexpensive to calculate. However, in general, they are not strong enough, e.g. [6].

### 3.2.2 Block-Diagonal Incomplete Cholesky Preconditioner

The equation for the search direction has a natural block structure:

$$[(\mathcal{X}_u+\mathcal{S})\ |\ \mathcal{X}_d]\begin{pmatrix}\Delta s\\ \Delta y\end{pmatrix}=-F_\mu,$$

and therefore the normal equations have the block structure

$$\left[\begin{array}{c|c}(\mathcal{X}_u{}^*+\mathcal{S}^*)(\mathcal{X}_u+\mathcal{S}) & (\mathcal{X}_u{}^*+\mathcal{S}^*)\mathcal{X}_d\\ \hline \mathcal{X}_d{}^*(\mathcal{X}_u+\mathcal{S}) & \mathcal{X}_d{}^*\mathcal{X}_d\end{array}\right]\begin{pmatrix}\Delta s\\ \Delta y\end{pmatrix}=-\begin{pmatrix}\mathcal{X}_u{}^*+\mathcal{S}^*\\ \mathcal{X}_d{}^*\end{pmatrix}F_\mu. \tag{3.4}$$

Given this block structure, it is natural to consider block-diagonal preconditioning. Following [6] and [1, Section 9.2] (see also [16, 15] and [14, Chapter 6]) we propose to use the preconditioner based on incomplete Cholesky factorizations of the diagonal blocks of the positive definite operator

$$\widetilde{P}^*\widetilde{P}=\left[\begin{array}{c|c}(\mathcal{X}_u{}^*+\mathcal{S}^*)(\mathcal{X}_u+\mathcal{S}) & 0\\ \hline 0 & \mathcal{X}_d{}^*\mathcal{X}_d\end{array}\right],$$

where

$$\begin{aligned}&(\mathcal{X}_u{}^*+\mathcal{S}^*)(\mathcal{X}_u+\mathcal{S})(v)=\\ &\quad \tfrac{1}{2}\mathrm{us2vec}\,[(X^2+(S^y)^2)\mathrm{us2Mat}\,(v)+\mathrm{us2Mat}\,(v)(X^2+(S^y)^2)]\\ &\qquad +\mathrm{us2vec}\,[X\mathrm{us2Mat}\,(v)S^y+S^y\mathrm{us2Mat}\,(v)X]\,.\end{aligned} \tag{3.5}$$

By complementary slackness, $XS^y \to 0$ as $\mu \to 0$. Therefore,

$$\begin{aligned}\|X\text{us2Mat}\,(v)S^y + S^y\text{us2Mat}\,(v)X\|^2 = \text{trace}\, S^y\text{us2Mat}\,(v)XX\text{us2Mat}\,(v)S^y + \\ \text{trace}\, X\text{us2Mat}\,(v)S^yS^y\text{us2Mat}\,(v)X + \\ 2\text{trace}\, S^y\text{us2Mat}\,(v)XS^y\text{us2Mat}\,(v)X.\end{aligned}$$

Therefore we can use the approximation

$$\begin{aligned}&(\mathcal{X}_u{}^* + \mathcal{S}^*)\,(\mathcal{X}_u + \mathcal{S})\,(v) \cong \\ &\quad \tfrac{1}{2}\text{us2vec}\,\left[(X^2 + (S^y)^2)\text{us2Mat}\,(v) + \text{us2Mat}\,(v)(X^2 + (S^y)^2)\right].\end{aligned} \tag{3.6}$$

In the previous Section 3.2.1, we showed that the bottom diagonal block is diagonal, so the exact Cholesky factorization can be found inexpensively. Furthermore, although the off-diagonals do not go to zero at convergence, it is reasonable to expect the incomplete Cholesky factorization for the top diagonal block and the exact Cholesky factorization (diagonal) for the bottom block to provide us a strong preconditioner for our problem. This strength was observed empirically, as shown by the computational results in Section 4.

We use the transformation between indices

$$c \cong (k,l), \quad c = \frac{(l-1)(l-2)}{2} + k, \quad k \le c, 1 \le k < l \le n.$$

The columns of the top left block follow.

$$\begin{aligned}(\mathcal{X}_u{}^* + \mathcal{S}^*)\,(\mathcal{X}_u + \mathcal{S})\,(e_c) &= \tfrac{1}{2\sqrt{2}}\text{us2vec}\,\left\{Z^2\left(e_ke_l^T + e_le_k^T\right) + \left(e_ke_l^T + e_le_k^T\right)Z^2\right\} \\ &= \tfrac{1}{2\sqrt{2}}\text{us2vec}\begin{pmatrix} \text{in row } k \quad (Z^2)_{l:} \\ \text{in row } l \quad (Z^2)_{k:} \\ \begin{pmatrix}\text{in col } k \\ (Z^2)_{:l}\end{pmatrix} \begin{pmatrix}\text{in col } l \\ (Z^2)_{:k}\end{pmatrix}\end{pmatrix}\end{aligned} \tag{3.7}$$

For $i \ne j$, we let $E_{ij} = \frac{1}{\sqrt{2}}\left(e_ie_j^T + e_je_i^T\right)$ denote the orthonormal basis for the symmetric matrix space. (When $i = j$, we use $e_ie_i^T$.) $\delta_{ij}$ denotes the *Kronecker delta*. Therefore, the element in row $r \cong (i,j)$ and column $c \cong (k,l)$ is

$$\begin{aligned}&\langle\text{us2vec}\,(E_{ij}), (\mathcal{X}_u{}^* + \mathcal{S}^*)\,(\mathcal{X}_u + \mathcal{S})\,(\text{us2vec}\,(E_{kl}))\rangle = \\ &\quad = \tfrac{1}{2}\text{us2vec}\,(E_{ij})^T\text{us2vec}\,\left\{(X^2 + (S^y)^2)E_{kl} + E_{kl}(X^2 + (S^y)^2)\right\} \\ &\qquad +\text{us2vec}\,(E_{ij})^T\text{us2vec}\,\left\{XE_{kl}S^y + S^yE_{kl}X\right\} \\ &\quad = \tfrac{1}{2}\text{trace}\,(E_{ij})\left\{Z^2E_{kl} + E_{kl}Z^2\right\} \\ &\quad = \text{trace}\, E_{ij}E_{kl}Z^2 \\ &\quad = \tfrac{1}{2}\text{trace}\,(e_ie_j^T + e_je_i^T)(e_ke_l^T + e_le_k^T)Z^2 \\ &\quad = \tfrac{1}{2}\text{trace}\,\left(e_ie_j^Te_ke_l^T + e_ie_j^Te_le_k^T + e_je_i^Te_ke_l^T + e_je_i^Te_le_k^T\right)Z^2 \\ &\quad = \tfrac{1}{2}\left\{\delta_{jk}(Z^2)_{li} + \delta_{jl}(Z^2)_{ki} + \delta_{ik}(Z^2)_{lj} + \delta_{il}(Z^2)_{kj}\right\}.\end{aligned} \tag{3.8}$$

### 3.3 least Squares using Regularization

Rather than use PCG, we now apply a Lanczos approach to a related eigenvalue problem. We will find the regularized solution of the parametrized trust region subproblem, denoted (TRS),

$$\text{(TRS)} \quad r^*(s) := \begin{array}{cc} \min & \|F'_\mu(\Delta v) + F_\mu(s,y)\| \\ \text{subject to} & \|\Delta v\| \leq s \end{array} \tag{3.9}$$

We apply the technique from [5]. After squaring the objective function and both sides of the constraint, the algorithm finds the minimum eigenvalue of the parametrized symmetric matrix

$$\begin{pmatrix} t & \mathcal{A}^*\left(F_\mu(s,y)\right)^T \\ \mathcal{A}^*\left(F_\mu(s,y)\right) & \mathcal{A}^*\mathcal{A} \end{pmatrix}, \tag{3.10}$$

where we denote $\mathcal{A} = \mathcal{F}'_\mu$ for convenience. Therefore, given the vector $\begin{pmatrix} y_0 \\ \Delta v \end{pmatrix}$, at each iteration of the Lanczos interior part we need to provide the vector

$$\begin{pmatrix} ty_0 + \langle \mathcal{A}^* F_\mu(s,y), \Delta v\rangle^T \\ y_0 \mathcal{A}^*\left(F_\mu(s,y)\right) + \mathcal{A}^*(\mathcal{A}\left(\Delta v\right)) \end{pmatrix}. \tag{3.11}$$

After normalizing so that $y_0 > 0$, the solution to TRS is the vector $\Delta v \leftarrow \frac{1}{y_0}\Delta v$.

# 4 Computational Results

## 4.1 Small Problems

We begin by presenting computational results for small problems with particular properties. We solve these problems using two approaches: (i) our specialized SDP algorithm and (ii) the mixed-cone (1.2) SDP formulation with the SDP code SeDuMi authored by Sturm [11]. The tests were done using MATLAB 6.5 on a Pentium IV PC with 256MB of RAM.

First we applied both approaches to small hard dense problems with $n$ ranging from 20 to 60. The construction of these problems is described in [7], and typical results are presented in Table 4.1. These problem model the situation where a correlation matrix is corrupted by relatively large errors. The problems are highly degenerate, i.e. strict complementarity fails and the

| Size of $A$ | CPU time for our algorithm with $\epsilon_1 = 10^{-8}$ | CPU time for our algorithm with $\epsilon_1 = 10^{-12}$ | Sedumi SOC |
|---|---|---|---|
| 20 | 31.4 | 46.3 | 7.7 |
| 30 | 182.4 | 260.9 | 48.1 |
| 40 | 758.6 | 1041.4 | 269.0 |
| 50 | 2220.5 | 3197.6 | 1042.9 |
| 60 | 5139.7 | 7279.6 | 3205.9 |

Table 4.1: Computational results for small hard $A$

sum of the optimal matrices $X + S^y$ has many zero eigenvalues. Although our specialized algorithm is less efficient in the absence of sparsity, we point out that it is able to achieve a very high level of precision without any numerical difficulties. This contrasts with the usual interior-point methods for SDP for which loss of strict complementarity is usually an issue.

We also compared the two approaches on randomly generated sparse matrices A of dimension up to $n = 70$. The required accuracy (for the optimality conditions) was set to $10^{-8}$ for both algorithms. The results are summarized in Figure 4.1.

As is typical with interior-point methods, the number of iterations required by Sedumi remains essentially constant (around 12 to 15 iterations) independently of the dimension of the problem; it is the computation time per iteration and the memory space required that quickly become prohibitively high. On the other hand, our algorithm is able to exploit the sparsity and the cost per iteration is much lower, so larger problems can be solved.

## 4.2 Large Sparse Problems

We solved three sets of 26 problems with dimensions $n = 200, 300, 350$ and densities for matrix $A$ given as .0005 to .003 in steps of .001. In all cases, we found the optimum to high accuracy, at least ten decimals. The results appear in Figures 4.2 and 4.3. We can see the correlation between the cputime and the number of nonzeros in the optimum $X$.

### 4.3 Robustness

If $A$ is not extremely sparse, then the matrix $X$ is not sparse and the operators $\mathcal{X}_u, \mathcal{S}, \mathcal{X}_d$ are not sparse. To avoid memory problems in the MATLAB codes, we initially zero out all values in $A$ below a certain tolerance, e.g. all values $\text{abs}\,(A_{ij}) < tol_0$, and initially $tol_0 = 0.9$. We solve the problem with this tolerance until we reach 3 decimals accuracy in the duality gap. We then decrease by 0.1 so $tol_0 = 0.8$. We continue until we reach $tol_0 = 0$. The algorithm is extremely robust and these perturbations do not slow it down appreciably showing that *warm starts* are possible using this approach.

## 5 Conclusion

We have presented a p-d-i-e-p algorithm for finding the closest correlation matrix to a given matrix. The preliminary numerical tests show promise. However, the cost for finding the search direction using a least squares approach is still too high. New tests are in progress that use a regularization approach to find this search direction.

## References


[1] O. AXELSSON. *Iterative solution methods.* Cambridge University Press, Cambridge, 1994.

[2] J.E. DENNIS Jr. and H. WOLKOWICZ. Sizing and least-change secant methods. *SIAM J. Numer. Anal.*, 30(5):1291–1314, 1993.

[3] M.M. DEZA and M. LAURENT. *Geometry of cuts and metrics.* Springer-Verlag, Berlin, 1997.

[4] M.C. FERRIS, M.P. MESNIER, and J.J. MORÉ. NEOS and Condor: Solving optimization problems over the Internet. *ACM Transactions on Mathematical Software*, 26(1):1–18, 2000.

[5] C. FORTIN and H. WOLKOWICZ. The trust region subproblem and semidefinite programming. *Optimization Methods and Software*, 2003. To appear in the special issue dedicated to Jochem Zowes 60th birthday, Taylor and Francis Publisher.

[6] A. GREENBAUM. *Iterative methods for solving linear systems.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[7] N.J. HIGHAM. Computing the nearest correlation matrix—A problem from finance. *IMA J. of Numerical Analysis*, 22(3):329–343, 2002.

[8] S. KRUK, M. MURAMATSU, F. RENDL, R.J. VANDERBEI, and H. WOLKOWICZ. The Gauss-Newton direction in linear and semidefinite programming. *Optimization Methods and Software*, 15(1):1–27, 2001.

[9] J. MALICK. A dual approach for conic least-squares problems. Technical report, INRIA, France, 2003.

[10] R.D.C. MONTEIRO and M.J. TODD. Path-following methods. In *Handbook of Semidefinite Programming*, pages 267–306. Kluwer Acad. Publ., Boston, MA, 2000.

[11] J.F. STURM. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.*, 11/12(1-4):625–653, 1999.

[12] H. WOLKOWICZ. Simple efficient solutions for semidefinite programming. Technical Report CORR 2001-49, University of Waterloo, Waterloo, Ontario, 2001.

[13] S. WRIGHT. *Primal-Dual Interior-Point Methods.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pa, 1996.

[14] T. YANG. *Iterative Methods for Least Squares and Total Least Squares Problems.* PhD thesis, Linköping University, Sweden, 1996.

[15] T. YANG. New IMGS-based preconditioners for least squares problems. In *Numerical analysis and its applications (Rousse, 1996)*, volume 1196 of *Lecture Notes in Comput. Sci.*, pages 581–588. Springer, Berlin, 1997.

[16] T. YANG and H.X. LIN. Solving sparse least squares problems with preconditioned CGLS method on parallel distributed memory computers. *Parallel Algorithms Appl.*, 13(4):289–305, 1999.

# A Adjoints

We evaluate the three adjoint operators. Let $v = \text{us2vec}\,(V)$, $V \in \mathcal{S}^n$, $z \in \mathbb{R}^n$, and $W \in \mathcal{M}^n$.

$$\begin{aligned}
\langle W, \mathcal{X}_u(v)\rangle &= \text{trace}\, W^T X \text{us2Mat}\,(v) \\
&= \text{trace}\, \text{us2Mat}\,(v) X W \\
&= \left\langle \text{us2Mat}\,(v), \frac{1}{2}(XW + W^T X) \right\rangle \\
&= \left\langle v, \frac{1}{2}\text{us2vec}\,(XW + W^T X) \right\rangle \\
&= \langle v, \mathcal{X}_u{}^*(W)\rangle\,;
\end{aligned}$$

$$\begin{aligned}
\langle W, \mathcal{X}_d(z)\rangle &= \text{trace}\, W^T X \text{Diag}\,(z) \\
&= \text{trace}\, \text{Diag}\,(z) W^T X \\
&= z^T \text{diag}\,(W^T X) \\
&= (e^T (W \circ X)) z \\
&= \langle \mathcal{X}_d{}^*(W), z\rangle\,;
\end{aligned}$$

$$\begin{aligned}
\langle W, \mathcal{S}(v)\rangle &= \text{trace}\, W^T \text{us2Mat}\,(v) S^y \\
&= \left\langle \text{us2Mat}\,(v), \frac{1}{2}(W S^y + S^y W^T) \right\rangle \\
&= \left\langle v, \frac{1}{2}\text{us2vec}\,(W S^y + S^y W^T) \right\rangle \\
&= \langle v, \mathcal{S}^*(W)\rangle\,.
\end{aligned}$$

In addition, we need the composition of operators:

$$\begin{aligned}
\mathcal{X}_d{}^* \mathcal{X}_d(z) &= \mathcal{X}_d{}^*(X \text{Diag}\,(z)) \\
&= \text{diag}\,(\text{Diag}\,(z) X^2);
\end{aligned}$$

$$\begin{aligned}
\mathcal{X}_d{}^* \mathcal{X}_u(v) &= \mathcal{X}_d{}^*\,(X \text{us2Mat}\,(v)) \\
&= \text{diag}\,\left(\text{us2Mat}\,(v) X^2\right);
\end{aligned}$$

$$\begin{aligned}
\mathcal{X}_d{}^* \mathcal{S}(v) &= \mathcal{X}_d{}^*\,(\text{us2Mat}\,(v) S^y) \\
&= \text{diag}\,\left(S^y \text{us2Mat}\,(v) X\right);
\end{aligned}$$

$$
\begin{aligned}
\mathcal{X}_u{}^*\mathcal{X}_d(z) &= \mathcal{X}_u{}^*\left(X\mathrm{Diag}\left(z\right)\right) \\
&= \frac{1}{2}\mathrm{us2vec}\left(X^2\mathrm{Diag}\left(z\right)+\mathrm{Diag}\left(z\right)X^2\right);
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{X}_u{}^*\mathcal{X}_u(v) &= \mathcal{X}_u{}^*\left(X\mathrm{us2Mat}\left(v\right)\right) \\
&= \frac{1}{2}\mathrm{us2vec}\left(X^2\mathrm{us2Mat}\left(v\right)+\mathrm{us2Mat}\left(v\right)X^2\right);
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{X}_u{}^*\mathcal{S}(v) &= \mathcal{X}_u{}^*(\mathrm{us2Mat}\left(v\right)S^y) \\
&= \frac{1}{2}\mathrm{us2vec}\left(X\mathrm{us2Mat}\left(v\right)S^y+S^y\mathrm{us2Mat}\left(v\right)X\right) \\
&= \mathcal{S}^*\mathcal{X}_u(v) \\
\mathcal{S}^*\mathcal{X}_u(v) &= \mathcal{S}^*(X\mathrm{us2Mat}\left(v\right)) \\
&= \frac{1}{2}\mathrm{us2vec}\left(X\mathrm{us2Mat}\left(v\right)S^y+S^y\mathrm{us2Mat}\left(v\right)X\right);
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{S}^*\mathcal{X}_d(z) &= \mathcal{S}^*(X\mathrm{Diag}\left(z\right)) \\
&= \frac{1}{2}\mathrm{us2vec}\left(X\mathrm{Diag}\left(z\right)S^y+S^y\mathrm{Diag}\left(z\right)X\right);
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{S}^*\mathcal{S}(v) &= \mathcal{S}^*(\mathrm{vec}\left(\mathrm{us2Mat}\left(v\right)S^y\right)) \\
&= \frac{1}{2}\mathrm{us2vec}\left(\mathrm{us2Mat}\left(v\right)(S^y)^2+(S^y)^2\mathrm{us2Mat}\left(v\right)\right).
\end{aligned}
$$

$$
\begin{aligned}
\left(\mathcal{X}_u{}^*+\mathcal{S}^*\right)\left(\mathcal{X}_u+\mathcal{S}\right)(v) &= \frac{1}{2}\mathrm{us2vec}\left[\mathrm{us2Mat}\left(v\right)\left(X^2+(S^y)^2\right)+\left(X^2+(S^y)^2\right)\mathrm{us2Mat}\left(v\right)\right] \\
&+ \mathrm{us2vec}\left[X\mathrm{us2Mat}\left(v\right)S^y+S^y\mathrm{us2Mat}\left(v\right)X\right]
\end{aligned}
$$

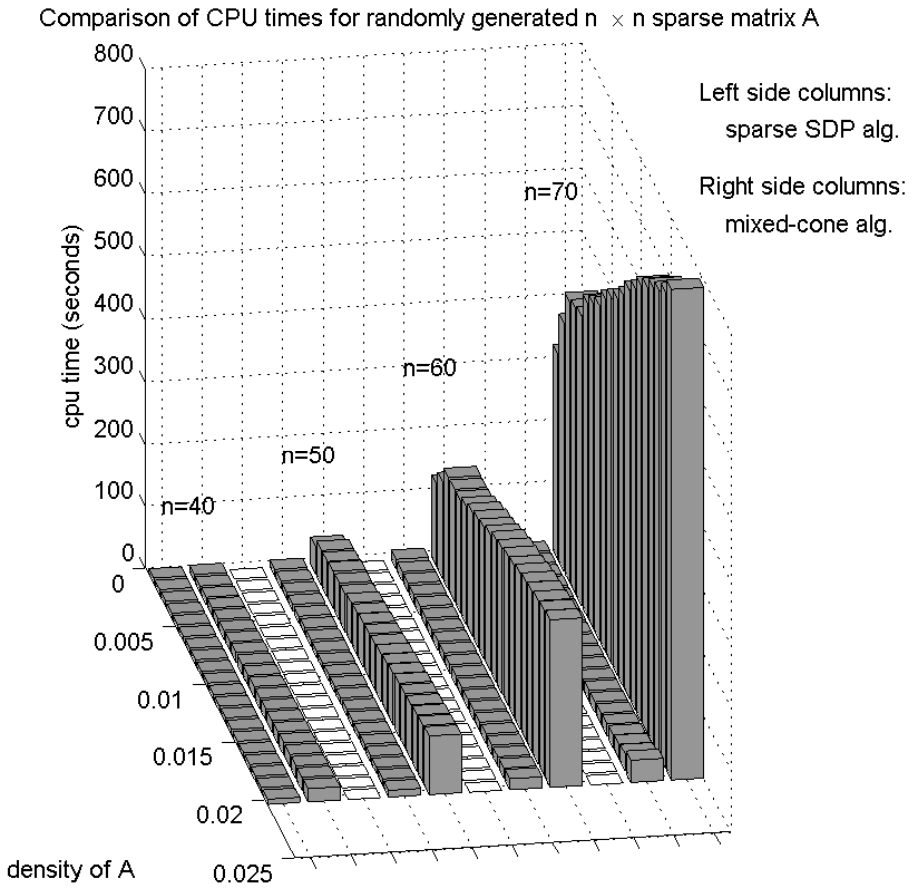


Figure 4.1: CPU times (averaged over 10 runs for each density)

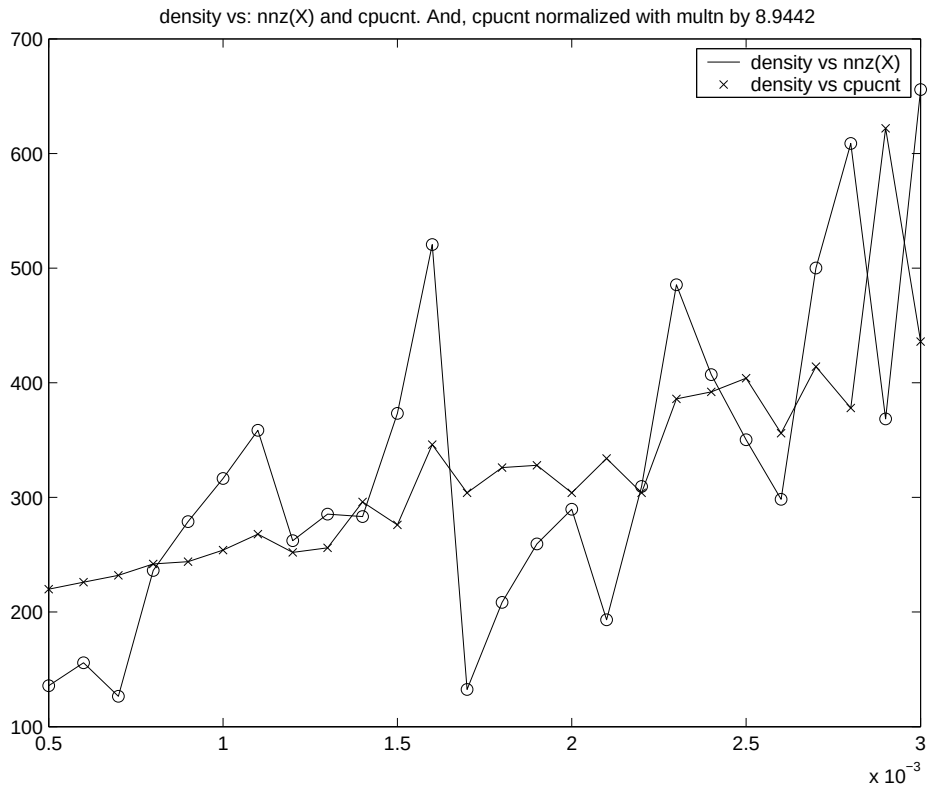


Figure 4.2: 30 problems; dimension $n = 200$

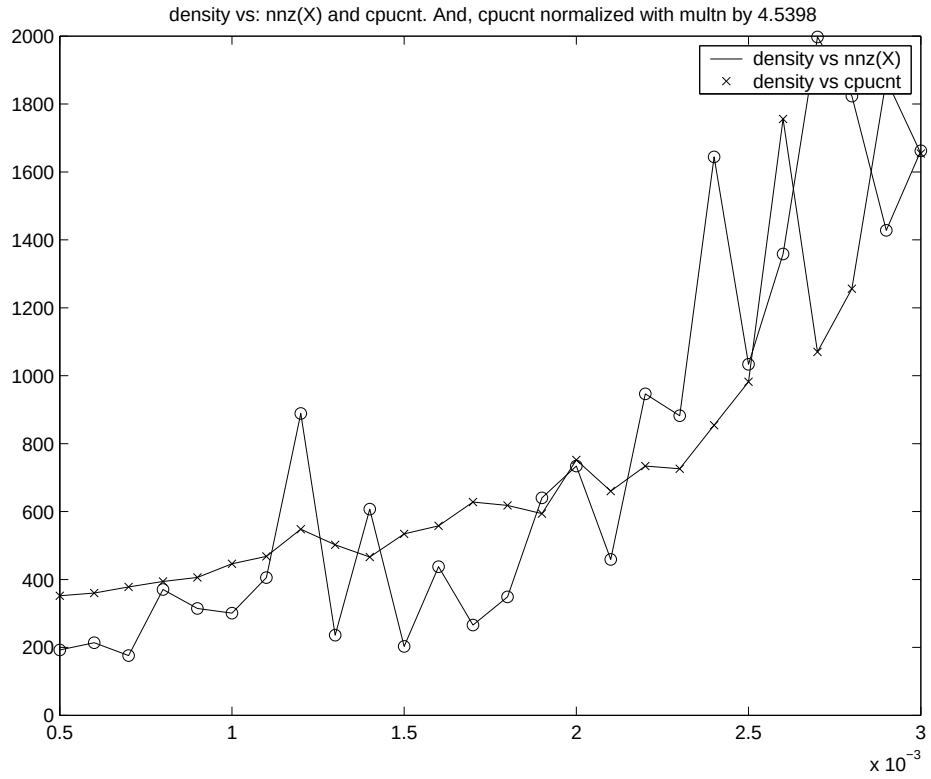


Figure 4.3: 30 problems; dimension $n = 300$

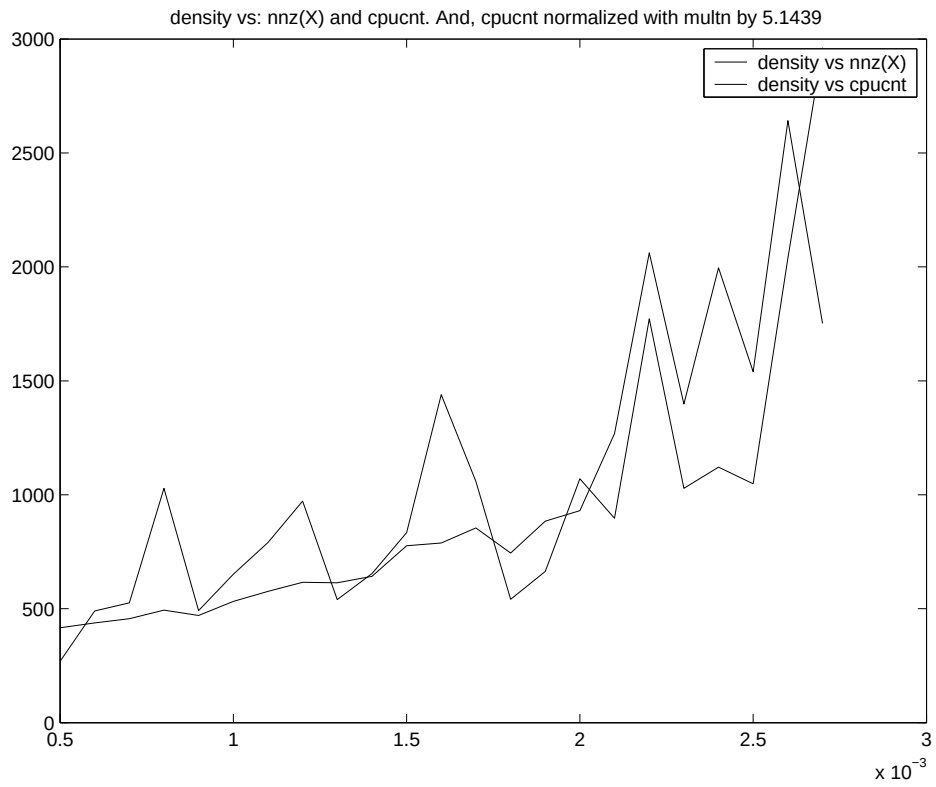


Figure 4.4: 28 problems; dimension $n = 350$