CONTINUOUS METHODS FOR BILINEAR AND QUADRATIC ASSIGNMENT PROBLEMS

JOHN M. CONROY, STEVEN G. KRATZER
AND LOUIS J. PODRAZIK

Abstract. In this paper we propose and analyze continuous methods to solve the bilinear and quadratic assignment problems. We first consider the permutation constrained two-sided matrix Procrustes problem and show that it is equivalent to solving a bilinear assignment problem, which is NP hard. Three methods are investigated to solve a continuous relaxation of the problem. The first of these methods replaces the permutations with orthogonal matrices and uses the singular value decomposition to attempt a solution by comparing the SVD's of a matrix and a perturbation of it. A second approach formulates the problem as a bilinear programming problem with non-negativity and linear constraints. We investigate both an alternating direction method and the Frank-Wolfe algorithm to solve this bilinear programming problem. We point out the advantages and deficiencies of each of the methods and their associated problem approximations in solving the discrete problem. Based upon our experience in solving the bilinear assignment problem, we also present the results of using the Frank-Wolfe [4] method to solve standard test cases of the quadratic assignment problem, an NP-hard problem. On this test set solutions within 10% of optimal were achieved with $O(n^3)$ work.

The authors thank Dianne O'Leary of the Computer Science Department, Univ. of MD, College Park for discussions relating to the SVD approach presented here.

1. Introduction

Linear assignment (LAP), bilinear assignment (BAP), and quadratic assignment problems (QAP) arise frequently in facility location problems in operations research (OR). The algorithms given in this paper are motivated by a matrix formulation of assignment problems. These formulation, we will see later are equivalent to the common formulations found in OR literature [7]. Informally, a linear assignment problem is to find a permutation such that when the columns of A are permuted the resulting matrix is as near as possible to B, given two $m \times n$ matrices A and B. A bilinear assignment problem is to find both a row and column permutation so that the permuted A is as near as possible to B. The quadratic assignment problem only differs from the bilinear problem in that the given matrices are square and that the permutation to be found must be symmetric, i.e., the same permutation must be applied to both the rows and columns of A. We now define some notation, which will allow us to formally define the LAP, BAP, and QAP and expose some of the structure of these problems.

Definition 1.1 (Notation)**.**

(1) $\|A\|_F = \left(\sum_{i,j} A_{i,j}^2\right)^{\frac{1}{2}}$

(2) *Hadamard product of two matrices* A, B*:* $(A \circ B)_{ij} = a_{ij} b_{ij}$.

(3) $\Pi_n \subsetneq E^{n\times n}$ *space of permutation matrices, the set of zero one matrices with exactly one 1 in each row and column.*
(4) $\Pi_{n,m} \subsetneq E^{n\times m}$ *space of partial permutation matrices*
(5) $\mathcal{D}_n \subsetneq E^{n\times n}$ *space of doubly stochastic matrices*

We now give the definition of the assignment problem, which is a special type of linear program.

Definition 1.2 (Assignment Problem (AP))**.** *Given $C \in E^{n\times n}$*

$$\max_{X\in\mathcal{D}_n} \sum_{i,j=1}^{n} c_{ij}x_{ij} = \max_{X\in\mathcal{D}_n} \sum_{i,j=1}^{n} (C\circ X)_{ij}$$

We note that the AP as defined is to find a doubly stochastic matrix which maximizes the above sum. Since the objective function is linear we know that the optimum will occur on a vertex of constraints set. The set vertices of the doubly stochastic matrices, $\mathcal{D}_n$, is the set of permutation matrices, Π_n.

Conversely, the constraint set for the quadratic assignment problem (QAP) will be the $\Pi_{n,m}$. Later in the paper we will considered a relaxed version of the QAP which has $\mathcal{D}_{n,m}$ as the constraint set.

Definition 1.3 (Quadratic Assignment Problem (QAP))**.** *Given $A \in E^{n\times n}$, $B \in E^{m\times m}$*

$$\min_{X\subsetneq\Pi_{n,m}} \sum_{i,j=1}^{n} \sum_{k,l=1}^{m} a_{ij}b_{kl}x_{ik}x_{jl} = \min_{X\subsetneq\Pi_{n,m}} \sum_{i=1}^{n}\sum_{j=1}^{m}(AX\circ XB)_{ij}$$

In the next section we reformulate the AP and the QAP as matrix optimization problems and introduce the bilinear assignment problem (BAP).

2. The Matrix Procrustes Problem & Preliminaries

We now need to define a bit more notation which we will need to help reformulate the AP and QAP.

Definition 2.1 (Notation)**.**

(1) *Given a matrix $A \in E^{n\times m}$, the stacking operator $s(.)$ creates an mn-vector $s(A)$ obtained by stacking the columns of A on top of each other.*
(2) *The m-vector of all ones is e_m.*
(3) *Given $x \in E^n$, the diagonal matrix whose diagonal elements are x is denoted $diag(x)$.*

We now consider the one-sided matrix Procrustes problem and observe the well-known result that it is equivalent to solving a linear assignment problem in the case of permutation matrix constraints.

Definition 2.2 (One-sided Matrix Procrustes Problem)**.** *Given $A, B \in E^{m\times n}$, find the permutation matrix $X \in E^{m\times m}$ to*

$$\min_{X\in\Pi_m} \|A - XB\|_F.$$

Lemma 2.1. *Problem 2.1 is equivalent to solving a linear assignment problem (LAP) with cost assignment matrix* AB^t*, that is,*

$$\min_{X\in\Pi_m} \|A - XB\|_F \equiv \max_{X\in\Pi_m} tr(AB^tX^t) = \max_{X\in\Pi_m} \sum_{i,j=1}^{m} \left(AB^t \circ X\right)_{ij}.$$

Proof: Since $X^tX = I$*, we have*

$$\begin{aligned}\|A - XB\|_F^2 &= tr\left((A - XB)^t(A - XB)\right)\\ &= \|A\|_F^2 + \|B\|_F^2 - 2tr(AB^tX^t).\end{aligned}$$

Furthermore,

$$\begin{aligned}tr(AB^tX^t) &= tr(diag(e_m)(AB^t)diag(e_m)X^t)\\ &= \langle e_m, (AB^t \circ X)e_m\rangle = \sum_{i,j=1}^{m} \left(AB^t \circ X\right)_{ij}. \quad \square\end{aligned}$$

We next consider one of the problems of interest, the permutation constrained matrix two-sided Procrustes problem and show that it is equivalent to solving a bilinear assignment problem.

Definition 2.3 (Two-sided Matrix Procrustes Problem)**.** *Given* $A, B \in E^{m\times n}$*,* $m \geq n$*, find the permutation matrices* $P \in E^{m\times m}$ *and* $Q \in E^{n\times n}$ *to*

$$\min_{(P,Q)\in(\Pi_m\times\Pi_n)} \|A - PBQ^t\|_F.$$

Lemma 2.2. *Problem 2.3 is equivalent to solving a bilinear assignment problem (BAP) with assignment matrices* A *and* B*, that is,*

$$\min_{(P,Q)\in(\Pi_m\times\Pi_n)} \|A - PBQ^t\|_F \equiv \max_{(P,Q)\in(\Pi_m\times\Pi_n)} tr(AQB^tP^t).$$

Proof: Since $\Pi \subset \mathcal{O}$*, the set of orthogonal matrices, the matrices* P *and* Q *are unitarily invariant and we have*

$$\begin{aligned}\|A - PBQ^t\|_F^2 &= tr\left((A - PBQ^t)^t(A - PBQ^t)\right)\\ &= \|A\|_F^2 + \|B\|_F^2 - 2tr(AQB^tP^t). \quad \square\end{aligned}$$

We now rewrite the bilinear cost in an algebraic form.

Lemma 2.3. *Given* $A, B \in E^{m\times n}$*,* $P \in E^{m\times m}$ *and* $Q \in E^{n\times n}$*,* $m \geq n$*, the bilinear cost given by the right hand side of Eqn. (2.2) can be rewritten as*

$$f(P, Q) = tr\left(\ AQB^tP^t\ \right) = \sum_{i,k=1}^{m} \sum_{j,l=1}^{n} a_{ij}b_{kl}p_{ik}q_{jl}.$$

Proof: We first write

$$\begin{aligned}tr(\ AQB^tP^t\ ) &= tr(\ diag(e_m)(AQ)diag(e_n)(PB)^t\ )\\ &= \langle e_m, (AQ \circ PB)e_n\rangle.\end{aligned}$$

Observing that $AQ \in E^{m\times n}$ *and* $PB \in E^{m\times n}$*, the elements of the products are*

$$(AQ)_{il} = \sum_{j=1}^{n} a_{ij}q_{jl}, \qquad (PB)_{il} = \sum_{k=1}^{m} p_{ik}b_{kl}.$$

Therefore the il-th element of the Hadamard product $(AQ \circ PB) \in E^{m\times n}$ *is*

$$(AQ \circ PB)_{il} = \sum_{j=1}^{n}\sum_{k=1}^{m} a_{ij} b_{kl} p_{ik} q_{jl},$$

resulting in

$$\sum_{i=1}^{m}\sum_{l=1}^{n}(AQ \circ PB)_{il} = \sum_{i,k=1}^{m}\sum_{j,l=1}^{n} a_{ij} b_{kl} p_{ik} q_{jl}. \quad \square$$

In order to efficiently compute the gradient $\nabla f(P,Q)$, we further exploit the problem structure.

Lemma 2.4. *Given A, B, P, and Q, the BAP objective function can be written as*

$$\begin{aligned} f(P,Q) &= \sum_{i=1}^{m}\sum_{j=1}^{n} \left(A \circ PBQ^t\right)_{ij} \\ &= \sum_{i,j=1}^{m} \left(AQB^t \circ P\right)_{ij} = \sum_{i,j=1}^{n} \left(A^t PB \circ Q\right)_{ij} \\ &= \langle s(P), (B \otimes A)s(Q)\rangle. \end{aligned}$$

Consequently, the gradient $\nabla f = (\nabla_P f, \nabla_Q f) \in (E^{m\times m}, E^{n\times n})$ *can be computed as*

$$\begin{aligned} \nabla_P f(P,Q) &= AQB^t \\ \nabla_Q f(P,Q) &= A^t PB. \end{aligned}$$

Proof: From Lemma 2.6,

$$\begin{aligned} f(P,Q) &= tr\left(\ AQB^tP^t\ \right) \\ &= tr\left(diag(e_m)Adiag(e_n)(PBQ^t)^t\right) \\ &= \langle e_m, (A \circ PBQ^t)e_n\rangle = \langle s(PBQ^t), s(A)\rangle \\ &= \langle (Q \otimes P)s(B), s(A)\rangle = \langle (Q \otimes I_m)(I_n \otimes P)s(B), s(A)\rangle \\ &= \langle (I_n \otimes P)s(B), (Q^t \otimes I_m)s(A)\rangle = \langle s(PBI_n), s(I_mAQ)\rangle \\ &= \langle s(PB), s(AQ)\rangle \\ &= \langle s(I_mPB), s(AQI_n)\rangle = \langle (B^t \otimes I_m)s(P), (I_n \otimes A)s(Q)\rangle \\ &= \langle s(P), (B \otimes I_m)(I_n \otimes A)s(Q)\rangle \\ &= \langle s(P), (B \otimes A)s(Q)\rangle = \langle s(P), s(AQB^t)\rangle \\ &= \langle (B^t \otimes A^t)s(P), s(Q)\rangle = \langle s(A^tPB), s(Q)\rangle. \quad \square \end{aligned}$$

By relaxing the constraint set to be doubly stochastic, we obtain the following continuous problem.

Definition 2.4 (Relaxed Two-Sided Procrustes Problem)**.** *Given* $A, B \in E^{m\times n}$, $m \geq n$,

$$\max_{(P,Q)\in(\mathcal{D}_m\times\mathcal{D}_n)} \sum_{i=1}^{m}\sum_{j=1}^{n} \left(A \circ PBQ^t\right)_{ij}.$$

Due to its bilinear objective and the fact that the continuous relaxation of the constraint set is simply the convex hull of the permutation constraint set, the solution of Problem 2.8 is also the solution to the discrete Problem 2.4.

Lemma 2.5. *Every KKT point of Problem 2.8 is an extreme point of* $(\mathcal{D}_m \times \mathcal{D}_n)$ *and in particular, the solution* $(P^*, Q^*) \in (\Pi_m \times \Pi_n)$.

Proof: -ADD- □

3. Three Methods to solve the Continuous Relaxation

In this section, we consider three methods to solve Problem 2.3. The first method is a direct method that exploits the particular structure of the matrix B and is based on the singular value decomposition. The other two methods we consider are iterative that exploit the bilinear structure of the cost function and the fact that the solution of the relaxed problem lies at a vertex of the constraint set.

3.1. The SVD Approach.

The SVD can be used to find an instance of LAP that approximates a given instance of QAP. This method is designed for "low residual" problems, where A is close to a permuted version of B.

Theorem 3.1 (SVD [5].). *Given* $A \in E^{m\times n}$, *there exist orthogonal matrices* $U \in E^{m\times m}$ *and* $V \in E^{n\times n}$, *and a diagonal matrix* $\Sigma \in E^{m\times n}$, *such that* $A = U\Sigma V'$, *with* $\Sigma =$diag$(\sigma_1, ..., \sigma_k)$ and $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_k \geq 0$, where $k = min(m, n)$.

Let $A = U_A\Sigma_A V_A^t$ and $B = U_B\Sigma_B V_B^t$. The model for deriving the SVD method is

$$PAP^t = B + E,$$

where $P \in \Pi_m$ $A \in E^{m\times n}, B \in E^{m\times n}$, $m \geq n$. In a typical application, $E \in E^{m\times n}$ is a matrix of measurement errors.

For a low-residual problem ($||E|| << ||A||$), we expect that the singular value spectrum of A is close to that of B. If, in addition, neither A nor B has repeated singular values, then the singular vectors of PAP^t and B are very close. Since the singular vectors of A are the eigenvectors of AA^t and A^tA, we can use the perturbation theory of the symmetric eigenvalue problem to examine the behavior of the SVD. From [8] simplifying for the symmetric case, we have the following. Let $MX = X\Lambda$ and $\hat{M}\hat{X} = \hat{X}\hat{\Lambda}$ be the eigenvalue decompositions of symmetric matrices M and $\hat{M}$ respectively, where $\hat{M} = M + F$. Suppose that the eigenvalues $\{\lambda_j\}$ are distinct and that $\epsilon/(\delta_j - \epsilon) < 1/2$, where $\epsilon = ||F||_2$, and $\delta_j = 1/[sqrt\sum 1/(\lambda_i - \lambda_j)^2]$. Then $||\bar{x_j} - x_j|| < \epsilon/\delta_j$.

Letting $M = BB^t$ and $\hat{M} = PAA^tP^t$, we can apply the above result to bound the perturbations to the singular vectors of B due to the noise matrix E. For a low-residual problem, $Pu_{A,j} \approx d_j u_{B,j}$, where $d_j = \pm 1$. If we can determine the signs d_j, then P can be approximated as $\sum \alpha_j d_j u_{B,j} u_{A,j}^t$, where the weights α_j reflect the relative error in the singular vectors.

If the elements of E are independent random variables with identical, Gaussian distributions, then it is reasonable to use a Gaussian model for the perturbations to the singular vectors as well.

Note that δ_j depends primarily on the closest neighboring singular vector $\sigma_{A,i}$ to $\sigma_{A,j}$. Therefore, we approximate δ_j by

$$\mathrm{weighted}(\mathrm{V_A}, \mathrm{V_B}, \Sigma_\mathrm{A}, \Sigma_\mathrm{B}) = \mathrm{diag}(\min(|\sigma_\mathrm{A,j} - \sigma_\mathrm{A,i}|, |\sigma_\mathrm{B,j} - \sigma_\mathrm{B,i}|)).$$

Based on the bound given by the perturbation theory, the variance of each element of $\bar{u}_j$ is approximately proportional to $1/\delta_{A,j}\delta_{B,j}$. The above function *weighted*() will be used to compute a weights which will be used to determine an approximate permutation of the singular vectors of A to match those of B.

Let $\bar{B} = U_{\bar{B}}\Sigma_{\bar{B}}V_{\bar{B}}^t$. Note that the right singular vectors of PBQ^T are independent of P. Therefore, $V_{\bar{B}} = V_{PBQ^T}D_V$, where D_V is a diagonal matrix with diagonal entries ± 1.

According to the above-mentioned perturbation theory, for a low-residual problem where the singular values of A are well separated, $V_{\bar{B}} \approx V_A D$ where D is a ± 1 diagonal matrix. Therefore $V_A^t \bar{V_B} \approx D$. The left singular vectors of BQ^T are independent of Q (except for a ± 1 diagonal matrix), so the LAP matrix becomes $\bar{B}A^T = U_{\bar{B}}\Sigma_{\bar{B}}V_{\bar{B}}^t V_A \Sigma_A U_A^T \approx U_B D \Sigma_B \Sigma_A U_A^T$. Similarly, we can solve for Q without knowing P by solving $LAP(A^T B)$, with $A^T B$ approximated by $V_A D \Sigma_A \Sigma_B V_B^T$. The details of this procedure are given in the below algorithm.

The remaining difficulty is the determination of the sign matrix D. An initial guess is $d_j = sgn(\hat{u}_{A,j}^T \hat{u}_{B,j})$ where $\hat{u}_{A,j}$ is a sorted version of the jth left singular vector of A, and similarly for $\hat{u}_{B,j}$. After the SVD has been used to get estimates of P and Q, improved value for D is then obtained by the iteration given in Step 4 below.

Algorithm 3.1. *function [p,q]=weightedsvd(A,B)*
/ Step 1: Compute the SVD's of A and B. */*

$$U_A \Sigma_A V_A = A;$$
$$U_B \Sigma_B V_B = B$$

/ Step 2: Assign consistent signs to the columns of the U and V matrices by forcing the max magnitude element in each column of U_A and U_B to be positive. */*

$$t = (\mathrm{sign}((\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{U_A(i,j)})(\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{U_B(i,j)})))$$
$$U_A = U_A \mathrm{diag(t)}$$
$$V_A = V_A \mathrm{diag(t)}$$

/ Step 3: Determine the permutation matrix P (specified by the index vector p) that minimizes $||PU_A U_B' - I||_F$ */*

$$U = weighted(U_A, U_B, \Sigma_A, \Sigma_A);$$
$$pvec = assign(U)';$$

/ Iterate to Improve Signs*/*
pold=0; iter=0;
while (p =pold)&(iter< 2n))
 $U_{aa} = U_A(p,:)$;
 t=sign(sum($U_{aa}.*U_B$));
 U=weighted($U_A\mathrm{diag(t)}, \mathrm{V_A diag(t)}, \Sigma_A, \Sigma_B$);
 pold=p;
 p=assign(U)';
 iter=iter+1;
end;

/ Step 4. Find the best permutation matrix Q given P*/*

$$V = weighted(V_A, V_B, \Sigma_A, \Sigma_B);$$

$$q = assign(V)';$$

In view of the fact that Eqn. (3.1) is a particular case of a one-sided permutation constrained matrix Procrustes problem, we first observe the following problem equivalence.

Lemma 3.1. *The following one-sided permutation constrained matrix Procrustes problem is equivalent to a maximum linear assignment problem.*

$$\min_{P\in\Pi_m} \|I - P(U_A S_A)U_B^t\|_F \equiv \max_{P\in\mathcal{D}_m} \sum_{i,j=1}^{m} \left(U_B (U_A S_A)^t \circ P\right)_{ij}$$

$$\min_{Q\in\Pi_n} \|I - Q(V_A S_A)V_B^t\|_F \equiv \max_{Q\in\mathcal{D}_n} \sum_{i,j=1}^{m} \left(V_B (V_A S_A)^t \circ Q\right)_{ij}$$

Proof: Simply let $A = I$ and $B=(U_A S_A)U_B^t$ in Lemma 2.1 □

Algorithm 3.2 (SVD Based Algorithm)**.** *Given $A, B \in E^{m\times n}$.*
Step 1: Compute

$$A = U_A \Sigma_A V_A^t$$
$$B = U_B \Sigma_B V_B^t.$$

Step 2: Make SVD of A, B unique by finding S_A s.t. components of largest magnitude in each column of $(U_A S_A)$ and (U_B) have the same sign.
Step 3: Solve the linear assignment problems:

$$P = \arg\ max_P \{ \sum_{i,j=1}^{m} \left(U_B (U_A S_A)^t \circ P\right)_{ij} \mid P \in \mathcal{D}_m \}$$

$$Q = \arg\ \max_Q \{ \sum_{i,j=1}^{m} \left(V_B (V_A S_A)^t \circ Q\right)_{ij} \mid Q \in \mathcal{D}_m \}$$

Remark 3.1.
The quality of the solution is highly dependent upon the choice of S_A.

3.2. **Alternating Assignment Problem Approach.** In view of the fact the cost is bilinear in Problem 2.8, we now consider an alternating direction approach [Vaish] that requires two linear assignment problems be solved at every iteration.

Algorithm 3.3 (Alternating Assignment Algorithm (AAA))**.** *Given $(P^1, Q^1) \in (\mathcal{D}_m \times \mathcal{D}_n)$.*
Step 0: Let $k = 1$.
Step 1: Solve the LAP:

$$P = \arg\ \max_P \{ \sum_{i,j=1}^{m} \left(\nabla_P f(P^k, Q^k) \circ P\right)_{ij} \mid P \in \mathcal{D}_m \}$$

Step 2: If $f(P, Q^k) < f(P^k, Q^k)$ then $P^{k+1} = P$, else $P^{k+1} = P^k$.

Step 3: Solve the LAP:

$$Q = \arg\ \max_{Q}\{\sum_{i,j=1}^{n} \left(\nabla_Q f(P^k, Q^k) \circ Q\right)_{ij} \mid Q \in \mathcal{D}_n\}$$

Step 4: If $f(P^k, Q) < f(P^k, Q^k)$ *then* $Q^{k+1} = Q$, *else* $Q^{k+1} = Q^k$.
Step 5: If $(P^{k+1}, Q^{k+1}) = (P^k, Q^k)$ *then stop; else go to step 6.*
Step 6: Let k = k + 1 and go to step 1.

Remark 3.2.
At each iteration, (P^k, Q^k) *is an extreme point of the constraint set. Consequently, the algorithm is a strictly boundary-following method.*
The algorithm linearly converges to a KKT point.
Many of the KKT points discovered by AAP algorithm will not correspond to the optimal permutations. To effectively use AAP method multiple starting points will have to be run.

3.2.1. *Frank-Wolfe Approach.* In view of the fact the solution to Problem 2.8 is a vertex, we now consider the Frank-Wolfe algorithm [xx].

Algorithm 3.4 (Frank-Wolfe Algorithm (FW))**.** *Given* $(P^1, Q^1) \in (\mathcal{D}_m \times \mathcal{D}_n)$.
Step 0: Let k = 1.
Step 1: If $\|P_{(\mathcal{D}_m \times \mathcal{D}_n)}[\nabla f(P^k, Q^k)]\| \leq \epsilon$ *then stop; else go to step 2.*
Step 2: Compute $W^k = (W_P, W_Q)$ *by solving the linear assignment problems:*

$$W_P = \arg\ \max_{W_P}\{\sum_{i,j=1}^{m} \left(\nabla_P f(P^k, Q^k) \circ W_P\right)_{ij} \mid W_P \in \mathcal{D}_m\}$$

$$W_Q = \arg\ \max_{W_Q}\{\sum_{i,j=1}^{m} \left(\nabla_Q f(P^k, Q^k) \circ W_Q\right)_{ij} \mid W_Q \in \mathcal{D}_n\}.$$

Step 3: Let $d^k = (d_P^k, d_Q^k)$, *where*

$$d_P^k = W_P - P^k, \qquad d_Q^k = W_Q - Q^k.$$

Step 4: Find $\alpha^k \in [0, 1]$ *such that*

$$f((P^k, Q^k) + \alpha^k d^k) < f(P^k, Q^k).$$

Step 5: Let

$$P^{k+1} = P^k + \alpha^k d_P^k, \qquad Q^{k+1} = Q^k + \alpha^k d_Q^k.$$

Step 6: Let k = k + 1 and go to step 1.

Remark 3.3.
This implementation is based upon a stop rule that uses the norm of the gradient projected onto the doubly stochastic constraint set; consequently, any solution produced by the above algorithm satisfies the first-order necessary optimality conditions. Other stop rules can be used that require less work than the projection and will be considered when the general QAP is solved later in the paper.
At each iteration, (W_P, W_Q) *is an extreme point of the constraint set. However, in general, the iterates* (P^k, Q^k) *remain in the interior of the constraint set.*
It is well-known that the algorithm is quadratically convergent when in the neighborhood of a KKT point.

4. Comparison and Results

We generated the test cases for Problem 2.8 as follows.

Definition 4.1 (Two-sided Matrix Procrustes Test Problem Generation)**.** *Given randomly generated $P \in \Pi_m$, $Q \in \Pi_n$, and given $A \in E^{m\times n}$, $m \geq n$, where*

$$a_{ij} \sim N(0,1),$$

we compute $B \in E^{m\times n}$ as

$$B = PAQ^t + E,$$

where $E \in E^{m\times n}$, $e_{ij} \sim N(0,\sigma)$.

Remark 4.1. *Clearly, $f(P^*, Q^*) = \|E\|_F$.*

σ	SVD		AAP		FW	
	#Cor Assign	%Cor Place	#Cor Assign	%Cor Place	#Cor Assign	%Cor Place
0.05	10	100.0	5	57.0	10	100.0
0.10	9	99.7	2	27.2	10	100.0
0.20	1	30.9	3	41.7	10	100.0
0.30	0	9.5	3	37.3	9	91.6
0.40	0	9.4	2	25.9	8	82.2
0.50	0	7.2	2	29.8	8	82.7

Table 1. Permuted Data Problems: 10 Problems, $m = n = 32$.

$n = m$	SVD		AAP		FW	
	#Cor Assign	%Cor Place	#Cor Assign	%Cor Place	#Cor Assign	%Cor Place
4	1	50.0	10	100.0	9	95.0
8	0	44.4	6	81.9	6	71.9
16	0	14.7	3	73.8	7	84.1
32	0	7.2	2	29.8	8	82.7
64	0	2.6	0	2.7	2	29.1

Table 2. Permuted Data Problem: 10 Problems, $\sigma = 0.5$.

5. General QAP

We now consider solving the general form of the quadratic assignment problem.

Definition 5.1 (One-sided Matrix Procrustes Problem)**.** *Given $A, C \in E^{n\times n}$ and $B \in E^{m\times m}$,*

$$\min_{X\in\Pi_n} \langle s(X), (B \otimes A)s(X)\rangle + \langle C, X\rangle \equiv \min_{X\in\Pi_n} \langle AXB^t + C, X\rangle.$$

Lemma 5.1. *Given $Y \in E^{n \times n}$, the projection of Y onto Π_n can be obtained by solving the LAP*

$$W = \arg \min_W \{\|W - Y\|_F \mid W \in \Pi_n\} \equiv \arg \max_W \{tr(YW^t) \mid W \in \mathcal{D}_n\}.$$

Proof: Since $W \in \Pi_n \subset \mathcal{O}$, we have

$$\begin{aligned} \|W - Y\|_F^2 &= tr((W-Y)^t(W-Y)) \\ &= (\|Y\|_F^2 + \|I\|_F^2) - 2tr(YW^t). \quad \square \end{aligned}$$

5.1. **Approach.**

(1) Relax constraint set $X \in \mathcal{D}_n \supseteq \Pi_n$
(2) Find interior optimum $Y \in \mathcal{D}_n$ using FW method

$$AP : \min_{d \in \mathcal{D}_n} \langle s(d), (B \otimes A)s(X) + s(C) \rangle$$

(3) Find $X^* \in \Pi_n$

$$AP : \max \Pi_n \langle s(Y), s(X) \rangle$$

5.2. **General QAP Test Problems.**

(1) QAPLIB - Burkard, Karisch, Rendl [1994] [2]
(2) n: 5 - 128

6. Results on Selected QAP Test Problems

Table 3. QAP Test Problems from C.E. Nugent, T.E. Vollmann and J.Ruml

Name	n	best solution	bound	FW solution	# starts
nug05	5	50	*	50	4†
nug06	6	86	*	86‡	2
nug07	7	148	*	148‡	1
nug08	8	214	*	214‡	3
nug12	12	578	*	578‡	23
nug15	15	1150	*	1150‡	2
nug20	20	2570	*	2570‡	10
nug30	30	6124	5772(IEB)	6124	39

* optimal solution
† A diag scaled
‡ same cost, different π than given

Table 4

Name	n	best solution	bound	FW solution	# starts
tho30	30	149936	136447(IEB)	149936	271
tho40	40	240516	214218(IEB)	241190	215

TABLE 5

Name	n	best solution	bound	FW solution	# starts
lipa10a	10	473	*	473	20
lipa10b	10	2008	*	2008	2
lipa20a	20	3683	*	3683	70
lipa50a	20	62093	*	62666	372

TABLE 6

Name	n	best solution	bound	FW solution	# starts
esc08a	8	2	*	2‡	1
esc08b	8	8	*	8‡	1
esc08c	8	32	*	32‡	2
esc08d	8	6	*	6‡	2
esc08e	8	2	*	2‡	1
esc08f	8	18	*	18‡	1
esc016a	16	68	47(ELI)	68	34
esc016b	16	292	250(ELI)	292	2
esc016c	16	160	95(ELI)	160	6
esc016d	16	16	3(GLB)	16	1
esc016e	16	28	12(GLB)	28	2
esc016f	16	0	0(GLB)	0	1
esc016g	16	26	12(GLB)	26	1
esc016h	16	996	708(ELI)	996	1
esc032a	32	130	35(GLB)	132	186
esc032b	32	168	96(GLB)	168	26
esc032c	32	642	464(ELI)	642	2
esc032d	32	200	106(GLB)	200	7
esc032e	32	2	0(GLB)	2	1
esc032f	32	2	0(GLB)	2	1
esc064a	64	116	47(GLB)	116	2

TABLE 7

Name	n	best solution	bound	FW solution	# starts
sko42	42	15812	13830(IEB)	15818	168
sko64	64	48498	43668(IEB)	48508	9
wil50	50	48816	47098(ILB)	48816	328

7. SUMMARY

(1) Continuous relaxation
(2) SVD - direct approach with sign matrix refinements
(3) AAP removed line search
(4) FW - best results

(1) Need efficient AP solver

(2) FW seen to zig-zag to interior solutions
(3) GP fast in iterations, slow projection
(4) solved many test problems quickly [1],[2]
(5) cheaper than existing methods [7],[6],[3]

References

[1] R.E. Burkard, S.E. Karisch, and R. Rendl. Qaplib – a quadratic assignment problem library. *European Journal of Operational Research*, 55:14–119, 1991.

[2] R.E. Burkard, S.E. Karisch, and R. Rendl. Qaplib – a quadratic assignment problem library. Technical report, University, 1994.

[3] G. Finke, R. E. Burkard, and F. Rendl. Quadratic assignment problems. *Annals of Discrete Mathematics*, 31:61–82, 1987.

[4] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logist. Quarterly*, 3:95–110, 1956.

[5] G. Golub and C. Van Loan. *Matrix Computations*, page 71. Johns Hopkins University Press, New York, New York, 1996??

[6] P.M. Pardalos, M.G.C. Resende, and Y. Li. Algorithm 754: Fortran subroutines for approximate solution of dense qap using grasp. *ACM Transactions on Mathematical Software*, 22(1):104–118, March 1996.

[7] P.M. Pardalos and H. Wolkowicz. *DIMACS Volume 16: Quadratic Assignment and Related Problem*. American Math. Society, Princeton, New Jersey, 1994.

[8] G.W. Stewart. *Introduction to Matrix Computations*, pages 293–295. Academic Press, New York, New York, 1982.