Globally Optimal Bilinear Programming for Computer Vision Applications

Manmohan Chandraker
mkchandraker@cs.ucsd.edu

David Kriegman
kriegman@cs.ucsd.edu

University of California, San Diego

Abstract

We present a practical algorithm that provably achieves the global optimum for a class of bilinear programs commonly arising in computer vision applications. Our approach relies on constructing tight convex relaxations of the objective function and minimizing it in a branch and bound framework. A key contribution of the paper is a novel, provably convergent branching strategy that allows us to solve large-scale problems by restricting the branching dimensions to just one set of variables constituting the bilinearity.

Experiments with synthetic and real data validate our claims of optimality, speed and convergence. We contrast the optimality of our solutions with those obtained by a traditional singular value decomposition approach. Among several potential applications, we discuss two: exemplar-based face reconstruction and non-rigid structure from motion. In both cases, we compute the best bilinear fit that represents a shape, observed in a single image from an arbitrary viewpoint, as a combination of the elements of a basis.

1. Introduction

Bilinearity is an oft-encountered phenomenon in computer vision, since observables in a vision system commonly arise due to an interaction between physical aspects well-approximated by linear models [27]. For instance, the coordinates of an image feature are determined by a camera matrix acting on a three-dimensional point [28]. Image intensity for a Lambertian object is an inner product between the surface normal and the light source direction. An image in non-rigid structure from motion arises due to a camera matrix observing a linear combination of the elements of a shape basis [29]. Thus, in its most general form, bilinear programming subsumes diverse sub-fields of computer vision such as 3D reconstruction, photometric stereo, non-rigid SFM and several others.

This paper proposes a practical algorithm that provably obtains the globally optimal solution to a class of bilinear programs widely prevalent in computer vision applications. The algorithm constructs tight convex relaxations of the objective function and minimizes it in a branch and bound framework. For an arbitrarily small ϵ, the algorithm terminates with a guarantee that the solution lies within ϵ of the global minimum, thus, providing a certificate of optimality.

One of the key contributions of the paper is to establish, with theoretical and empirical justification, that it is possible to attain convergence with a non-exhaustive branching strategy that branches on only a particular set of variables. (Note that this is different from bounds propagation schemes proposed in [1, 8], which are essentially exhaustive.) This has great practical significance for many computer vision problems where bilinearity arises due to the interaction of a small set of variables with a much larger, independent set.

For instance, one commonly represents an entity, say shape, as a linear combination of basis entities. The image formation process under this model can be understood as a bilinear interaction between a small number of camera parameters and a large number of coefficients for the basis shapes. As an illustration, we present two applications where the nature of this interaction is suitably exploited - reconstructing the 3D structure of a face from a single input image using exemplar models [7] and determining the cameras and shape coefficients in non-rigid structure from motion [29].

In this paper, we globally minimize bilinear programs under both the standard L_2-norm for Gaussian distributions and the robust L_1-norm corresponding to the heavier-tailed Laplacian distribution. The convex relaxations are linear programs (LP) for the L_1-norm case and second order cone programs (SOCP) for the L_2-norm, both of which are efficiently solvable by modern interior point methods [3].

A traditional counterpoint to our approach would employ linear regression followed by singular value decomposition (SVD), which is suboptimal for this rank-constrained problem in the noisy case. Our experiments clearly demonstrate the lower error rates compared to SVD attainable by our globally optimal algorithms.

The remainder of the paper is organized as follows: Section 2 describes related work and Section 3 briefly reviews branch and bound theory. Section 4 presents convex relax-

ations for the bilinear programs under consideration. Section 5 proposes a branching strategy to globally minimize our programs and proves convergence. Experimental results on real and synthetic data are presented in Section 6, while Section 7 concludes with a discussion and future directions.

2. Related Work

Bilinear problems arise in several guises in computer vision [17], a fact highlighted by the wisespread use of singular value decomposition in diverse vision algorithms. This is, in part, due to the preponderance of linear models in our understanding of several aspects of visual phenomena, such as structure from motion [28], illumination models [6, 11], color spectra [20] and 3D appearance [22].

The bilinear coupling between head pose and facial expression is recovered in [5] for actor-driven animation. Disparate applications like typography, face pose estimation and color constancy are tackled in [9] by learning and fitting bilinear models in an EM framework. A perceptual motivation for the abundance of bilinearity in vision is presented in [27].

From the perspective of optimization algorithms, bilinear programming is quite well-studied [21], particularly as a special case of biconvex programming [2]. A variety of approaches, such as cutting-plane algorithms [18] and reformulation linearization techniques [25] have been proposed to solve bilinear programs. Our approach differs in exploiting structure to achieve optimality in problems which would otherwise be considered too large for global optimization.

There has been significant recent activity in the community towards global optimization for vision problems, using a variety of tools, such as LMI relaxations [16], SOCP/SDP relaxations in a branch and bound framework [1, 8], Groebner basis for polynomial systems [26] and interval analysis methods [10]. Note that bilinear programming is a special case of polynomial optimization, however, the problem sizes that concern us in this paper are far greater than what modern polynomial solvers can handle [12].

Numerous computer vision applications involve norms besides L_2. Supergaussian distributions like Laplacian routinely arise in independent component analysis [14]. The L_1 norm has been used where robustness is a primary concern, such as [30] for range image integration. The underlying convexity or quasi-convexity of L_1 and L_∞ formulations of multiview geometry problems have been studied in [1, 15, 24].

3. Branch and Bound

Branch and bound algorithms are non-heuristic methods for global optimization of nonconvex problems [13]. For arbitrarily small ϵ, they terminate with a certificate guaranteeing an ϵ-suboptimal solution.

Let $\Phi(x)$ be a multivariate scalar function that we wish to globally minimize over a rectangular domain Q_0. *Branching* refers to splitting a rectangle into two or more sub-rectangles. Let $\mathcal{P}_k$ be a partition of Q_0 after k branching operations. The *bounding* operation computes provable lower and upper bounds within each $Q_i \in \mathcal{P}_k$, denoted by $\Phi_{\text{lb}}(Q_i)$ and $\Phi_{\text{ub}}(Q_i)$. The algorithm also maintains global lower and upper bounds for the function Φ after k branching operations:

$$\psi_k^l = \min_{Q_i \in \mathcal{P}_k} \Phi_{\text{lb}}(Q_i), \quad \psi_k^u = \min_{Q_i \in \mathcal{P}_k} \Phi_{\text{ub}}(Q_i). \quad (1)$$

The algorithm is convergent if $\psi_k^u - \psi_k^l \to 0$ as $k \to \infty$.

The success of a branch and bound scheme greatly depends on the quality of the bounds. They must be good approximations to the original function, as well as efficiently computable and minimizable. The bounding function must also satisfy a technical condition: namely, it must approximate the function increasingly well as the "size" of the rectangle shrinks, for some notion of "size".

Choosing a rectangle for branching is essentially heuristic and does not affect convergence. We choose the rectangle with the lowest value of $\Phi_{\text{lb}}(Q)$ and subdivide it along a particular dimension. The choice of branching dimension and the actual subdivision do influence convergence. While the worst-case complexity of a branch and bound algorithm can be exponential, we significantly expedite the solution with a branching strategy that exploits problem structure. Section 5 describes our approach and proves it to be convergent.

4. Formulation

To motivate the derivations and establish a consistent terminology, we will refer to a concrete example, namely reconstructing a 3D shape from a single image, given basis shapes. Modulo suitable variable reorderings, the following derivations hold for other bilinear programs too.

For simplicity, we will write expressions for one coordinate of the 2D image, extension to two coordinates is straightforward. Let $\mathbf{u} = \{u_j\}_{j=1}^N$ be the observed image, $\mathbf{a} \in \mathbb{R}^n$ be a row of the camera matrix ($n = 4$ for 3D shapes) and $\boldsymbol{\alpha} = \{\alpha^i\}_{i=1}^m$ be the shape coefficients corresponding to the m basis shapes $\mathcal{B}^i = \{\mathbf{X}_j^i \in \mathbb{R}^n\}$. Then, the shape is represented as $\sum_i \alpha^i \mathcal{B}^i$ and the affine imaging equation is

$$u_j = \mathbf{a}^\top \textstyle\sum_{i=1}^m \alpha^i \mathbf{X}_j^i, \quad j = 1, \cdots, N \quad (2)$$

4.1. LP relaxation for the L_1-norm case

The L_1-norm bilinear program to find the globally optimal camera and shape coefficients is:

$$\min_{\mathbf{a},\boldsymbol{\alpha}} \sum_{j=1}^N \left| u_j - \mathbf{a}^\top \sum_{i=1}^m \alpha^i \mathbf{X}_j^i \right| \quad (3)$$
$$\text{subject to} \quad \mathbf{a} \in \mathcal{Q}_a\,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha\,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0$$

where $\mathbf{X}_j^i \in \mathbb{R}^n$ and $\mathcal{Q}_a$, $\mathcal{Q}_\alpha$ specify rectangular domains for $\mathbf{a}$ and $\boldsymbol{\alpha}$. $\mathcal{G}(\mathbf{a}, \boldsymbol{\alpha})$ represents a set of linear constraints

(or constraints which can be relaxed into linear ones) on $\mathbf{a}$ and/or $\boldsymbol{\alpha}$ to fix the scale of the variables.

Introducing scalar variables t_j, $j = 1, \cdots, N$, an equivalent constrained optimization problem is

$$\min_{\mathbf{a},\boldsymbol{\alpha},\mathbf{t}} \sum_{j=1}^{N} t_j \,, \quad \text{s.t. } t_j \geq \left| u_j - \sum_{k=1}^{n}\sum_{i=1}^{m} a_k \alpha^i X^i_{j,k} \right| \tag{4}$$
$$\mathbf{a} \in \mathcal{Q}_a \,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha \,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0$$

where $\mathbf{a} = (a_1, \cdots, a_n)^\top$ and $\mathbf{X}^i_j = (X^i_{j,1}, \cdots, X^i_{j,n})^\top$. Note that the non-convexity in the problem is now contained in the bilinear terms $a_k\alpha^i$ in the constraints. The convex and concave relaxations for a bilinear term xy in the domain $[x_l, x_u] \times [y_l, y_u]$ are given by, respectively, the two pairs of linear inequalities [2, 21]:

$$z \geq \max\left\{x_l y + y_l x - x_l y_l,\ x_u y + y_u x - x_u y_u\right\} \tag{5}$$
$$z \leq \min\left\{x_u y + y_l x - x_u y_l,\ x_l y + y_u x - x_l y_u\right\} \tag{6}$$

We will collectively refer to the four inequalities above as $\operatorname{conv}(xy) \leq z \leq \operatorname{conc}(xy)$.

Each constraint of the form $t_j \geq |\cdot|$ in (4) can be equivalently replaced by the pair of constraints $t_j \geq (\cdot)$ and $t_j \geq -(\cdot)$. Substituting $\gamma^i_k = a_k\alpha^i$, we can construct a convex relaxation of (4):

$$\min_{\mathbf{a},\boldsymbol{\alpha},\boldsymbol{\gamma},\mathbf{t}} \sum_{j=1}^{N} t_j \,, \quad \text{s.t. } t_j \geq \left(\sum_{k=1}^{n}\sum_{i=1}^{m} X^i_{j,k}\gamma^i_k - u_j \right) \tag{7}$$
$$t_j \geq -\left(\sum_{k=1}^{n}\sum_{i=1}^{m} X^i_{j,k}\gamma^i_k - u_j \right)$$
$$\operatorname{conv}\left(a_k\alpha^i\right) \leq \gamma^i_k \leq \operatorname{conc}\left(a_k\alpha^i\right)$$
$$\mathbf{a} \in \mathcal{Q}_a \,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha \,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0.$$

Introducing new variables $\mu_j = t_j - \sum_{i,k} \mathbf{X}^i_{j,k}\gamma^i_k + u_j$ and eliminating t_j from the resulting system of equations, the program (7) can be equivalently rewritten as

$$\min_{\mathbf{a},\boldsymbol{\alpha},\boldsymbol{\gamma},\boldsymbol{\mu}} \quad \sum_{j=1}^{N} \left(\sum_{k=1}^{n}\sum_{i=1}^{m} \mathbf{X}^i_{j,k}\gamma^i_k - u_j + \mu_j \right) \tag{8}$$
$$\text{subject to} \quad -2\left(\sum_{k=1}^{n}\sum_{i=1}^{m} \mathbf{X}^i_{j,k}\gamma^i_k - u_j \right) - \mu_j \leq 0$$
$$\mu_j \geq 0$$
$$\operatorname{conv}\left(a_k\alpha^i\right) \leq \gamma^i_k \leq \operatorname{conc}\left(a_k\alpha^i\right)$$
$$\mathbf{a} \in \mathcal{Q}_a \,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha \,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0.$$

While both (7) and (8) are linear programs, (7) has two "general" linear inequalities in the constraint set for each $j = 1, \cdots, N$. In (8), one of them has been replaced by a comparison of the scalar variable μ_j to 0, which can be handled in a more efficient manner in interior point solvers [3].

In computer vision applications where $N \gg n$, the importance of this transformation is immense - it improves timings by up to an order of magnitude in some of our experiments.

4.2. SOCP relaxation for the L_2-norm case

The L_2-norm bilinear problem is:

$$\min_{\mathbf{a},\boldsymbol{\alpha}} \sqrt{\sum_{j=1}^{N} \left(u_j - \mathbf{a}^\top \sum_{i=1}^{m} \alpha^i \mathbf{X}^i_j \right)^2} \tag{9}$$
$$\text{subject to} \quad \mathbf{a} \in \mathcal{Q}_a \,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha \,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0.$$

A convex relaxation for (9) can be easily constructed in the form of a second order cone program using the same principles as for the L_1 case:

$$\min_{\mathbf{a},\boldsymbol{\alpha},\boldsymbol{\gamma}} \left\| u_1 - \sum_{i,k} \mathbf{X}^i_{1,k}\gamma^i_k \,, \cdots, u_N - \sum_{i,k} \mathbf{X}^i_{N,k}\gamma^i_k \right\|$$
$$\operatorname{conv}\left(a_k\alpha^i\right) \leq \gamma^i_k \leq \operatorname{conc}\left(a_k\alpha^i\right)$$
$$\mathbf{a} \in \mathcal{Q}_a \,,\ \boldsymbol{\alpha} \in \mathcal{Q}_\alpha \,,\ \mathcal{G}(\mathbf{a}, \boldsymbol{\alpha}) \geq 0. \tag{10}$$

The convex relaxations in (8) and (10) are tight and efficiently computable, so they are ideal for use in a branch and bound algorithm (Section 5).

4.3. Additional notes for the L_2 case

A traditional approach to solving such bilinear problems as (9) might use SVD. In particluar, one may substitute $\beta^i_k = \alpha^i a_k$, $i = 1, \cdots, m$ and $k = 1, \cdots, n$ and solve for β^i_k by minimizing the linear least squares

$$\min \ \textstyle\sum_j (u_j - \sum_{\{i,k\}} \mathbf{X}^i_{j,k}\beta^i_k)^2. \tag{11}$$

Subsequently, arrange β^i_k in a $n \times m$ matrix $\boldsymbol{\beta}$, perform SVD and retain the first singular vector. However, this is not optimal in the noisy case as the rank 1 structure of $\boldsymbol{\beta}$ is ignored by the linear regression in (11). Some of the experiments in Section 6 will underline the global optimality of our L_2-norm solution by comparison against linear regression followed by SVD.

A second observation is that, for a differentiable problem like (9), an additional optimization is possible. Once the lower bounding obtains a feasible point, we can refine the objective function value using a gradient descent method within the rectangle under consideration. Note that this does not compromise optimality, it simply allows the best estimate in a rectangle to push closer to the lower bound. Since our experiments are designed to validate the effectiveness of the branch and bound strategy, all of them are conducted *without* this additional optimization.

Finally, for an orthographic camera, orthogonality constraints of the form $\mathbf{a}^\top\mathbf{b} = 0$ are bilinear equalities and

can be handled using (5) and (6). Unit norm constraints such as $\|\mathbf{a}\| = 1$, however, can only be imposed indirectly. One approach would be to consider only those rectangles, Q, feasible for branching that satisfy the condition $\min_{\mathbf{a}\in Q} \mathbf{a}^\top\mathbf{a} \leq 1 \leq \max_{\mathbf{a}\in\mathcal{V}(Q)} \mathbf{a}^\top\mathbf{a}$, where $\mathcal{V}(Q)$ is the set of vertices of the rectangle Q. The left inequality is a small convex quadratic program while the right inequality is a small enumeration. Thus, at a negligible cost, rectangles which are guaranteed not to contain the $\|\mathbf{a}\| = 1$ solution can be pruned away.

5. Branching strategy

Suppose we wish to globally minimize a bilinear function $f(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m$, which might contain bilinear terms involving $x_i y_j$ and linear terms involving x_i. Given a procedure for constructing the convex relaxations as in Section 4, we propose a branching strategy to minimize the bilinear objective. Let ψ_k^l and ψ_k^u be the global lower and upper bounds maintained by the branch and bound algorithm at the end of k iterations, as defined in equation (1). The branching strategy of the algorithm is given by:

- $k = 0, \mathcal{P}_0 = \{Q_0\}$.
- while $\psi_k^u - \psi_k^l > \epsilon$
  - choose $Q \in \mathcal{P}_k$ such that $\Phi_{\text{lb}}(Q) = \psi_k^l$
  - bisect the longest edge of Q that corresponds *only* to some x_i to form two rectangles Q_1 and Q_2
  - $\mathcal{P}_{k+1} = \{\mathcal{P}_k \backslash Q\} \bigcup \{Q_1, Q_2\}$
  - $k = k + 1$.

The algorithm differs from traditional approaches in the choice of branching dimensions - we restrict the branching dimensions to those corresponding to only one set of variables that make up the bilinear forms. To borrow the terminology of [27], we branch over either the style variables or the content ones.

We will prove below that this branching strategy results in a convergent algorithm. The practical advantage in computer vision applications is that, typically, one set of variables in a bilinear model has far fewer elements than the other. For instance, Section 6 exploits this fact to branch over just the camera parameters while optimizing for both the camera and the linear coefficients of a shape basis consisting of tens to hundreds of exemplars. The convergence is also empirically demonstrated in the experiments in Section 6.1.

Proposition 1. *The above branch and bound algorithm is convergent.*

We will prove the claim for bilinear functions of the form $f(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n x_i y_i$. It can be easily extended to deal with more general bilinear forms by replacing y_i with linear functions of $\mathbf{y}$ and appending f with linear functions of $\mathbf{x}$. The program we solve in, say (9), contains the bilinearity in the constraint set, rather than the objective. With continuity arguments, the following analysis holds for that case too.

Let the initial region be specified as $Q_0 = [l_1, u_1] \times \cdots \times [l_n, u_n] \times [L_1, U_1] \times \cdots \times [L_n, U_n]$, where $x_i \in [l_i, u_i]$ and $y_i \in [L_i, U_i]$. Define $\chi(Q)$ to be the length of the longest edge of rectangle Q corresponding *only* to some x_i, that is, $\chi(Q) = \max_i (u_i - l_i)$. We show that the branch and bound algorithm is convergent if the sub-division rule at each branching step is to bisect an x_i-edge of length $\chi(Q)$.

We first prove the following main result:

Lemma 1. *As* $\chi(Q) \to 0$ *,* $\Phi_{ub}(Q) - \Phi_{lb}(Q) \to 0$.

Proof. Let us begin with a single bilinear term $x_i y_i$. For ease of notation, we drop the subscript and denote x_i by x and y_i by y. The corresponding rectangle is denoted by $[l, u] \times [L, U]$. We define

$$\lambda = \frac{L-U}{u-l}x + \frac{uU - lL}{u-l}\ ,\ \lambda' = \frac{U-L}{u-l}x + \frac{uL - lU}{u-l}. \tag{12}$$

Noting the form of the convex and concave relaxations of xy in (5) and (6), respectively, the following four cases arise:

$$\begin{aligned}
y \geq \lambda,\ y \leq \lambda' \ &\Rightarrow\ \Phi_{\text{ub}} - \Phi_{\text{lb}} \ = \ (u - x)(U - L) \\
y \geq \lambda,\ y \geq \lambda' \ &\Rightarrow\ \Phi_{\text{ub}} - \Phi_{\text{lb}} \ = \ (u - l)(U - y) \\
y \leq \lambda,\ y \leq \lambda' \ &\Rightarrow\ \Phi_{\text{ub}} - \Phi_{\text{lb}} \ = \ (u - l)(y - L) \\
y \leq \lambda,\ y \geq \lambda' \ &\Rightarrow\ \Phi_{\text{ub}} - \Phi_{\text{lb}} \ = \ (x - l)(U - L)
\end{aligned} \tag{13}$$

The lemma is proved for a single bilinear term, if $\forall \epsilon > 0, \exists \delta > 0$ such that

$$u - l < \delta \Rightarrow \Phi_{\text{ub}} - \Phi_{\text{lb}} < \epsilon,\ \forall x \in [l, u], \forall y \in [L, U]. \tag{14}$$

And indeed, given $\epsilon > 0$, choosing $\delta < \dfrac{\epsilon}{U - L}$, satisfies the condition (14) for each of the above four cases.

For the case where f is a sum of bilinear terms, we appeal to a result from [13]: let Φ_{lb}^i and Φ_{ub}^i be the convex and concave envelopes of $x_i y_i$ in the domain $[l_i, u_i] \times [L_i, U_i]$. Then the convex and concave envelope of $f(\mathbf{x}, \mathbf{y}) = \sum_i x_i y_i$ in the domain $Q = \prod_i [l_i, u_i] \prod_i [L_i, U_i]$ are given by $\Phi_{\text{lb}}(Q) = \sum_i \Phi_{lb}^i$ and $\Phi_{\text{ub}}(Q) = \sum_i \Phi_{ub}^i$.

Consequently, Lemma 1 stands proved if $\forall \epsilon > 0, \exists \delta > 0$ such that, $\forall x_i \in [l_i, u_i],\ \forall y_i \in [L_i, U_i]$

$$\chi(Q) < \delta \Rightarrow \textstyle\sum_i \Phi_{ub}^i - \sum_i \Phi_{lb}^i < \epsilon. \tag{15}$$

And indeed, since $\chi(Q) \geq \max_i (u_i - l_i)$, choosing δ such that $n\delta \max_i (U_i - L_i) < \epsilon$ always satisfies (15). □

With Lemma 1 proved, the convergence of our algorithm can be easily derived by standard arguments [4, 13]. Our choice of branching strategy induces some minor technical differences, so the appendix includes a short proof for completion.

6. Experiments

We conduct experiments for performance evaluation with synthetic data, with and without outliers, for the optimal L_2 and the optimal L_1 methods as well as the (suboptimal) linear regression followed by SVD method. Experiments are also reported for two applications - face reconstruction from exemplar 3D models (with real data) and bilinear fitting for non-rigid structure from motion (with synthetic data).

Both the LP and SOCP relaxations are implemented using the MOSEK solver [3], in an unoptimized MATLAB environment[1]. The branch and bound terminates when the difference between the global optimum and the convex underestimator is less than a stringent tolerance of 0.001. Regardless of the problem norm, any reprojection (image) errors we report are computed as Root Mean Squares (RMS) errors.

6.1. Synthetic data

The purpose of these experiments is to understand the behavior of the algorithm across varying noise levels, number of exemplars and number of point correspondences. For each experiment, the exemplar models are generated randomly in the cube $[-1, 1]^3$, while α^i are random numbers in $[0, 1]$ such that $\sum_i \alpha^i = 1$. The camera, which is $\mathbf{a} \in \mathbb{R}^4$ in equations (3) and (9), is generated randomly with each $a_i \in [-1, 1]$. Zero mean, Gaussian noise of varying standard deviation is added to the image coordinates. We define:

$$\begin{aligned}
\text{Reprojection error}: &\ \sqrt{\textstyle\sum_{j=1}^{N}(u_j - \widehat{u_j})^2/N} \\
\text{Camera error}: &\ \sqrt{\textstyle\sum_{k=1}^{n}(a_k - \widehat{a_k})^2/(n\|\widehat{a_k}\|)} \\
\text{Shape coefficient error}: &\ \sqrt{\textstyle\sum_{i=1}^{m}(\alpha^i - \widehat{\alpha^i})^2/(m\sum_i \widehat{\alpha^i})} \\
\text{3D error}: &\ \sqrt{\textstyle\sum_{j=1}^{N}(\mathbf{X}_j - \widehat{\mathbf{X}_j})^2/N}
\end{aligned}$$

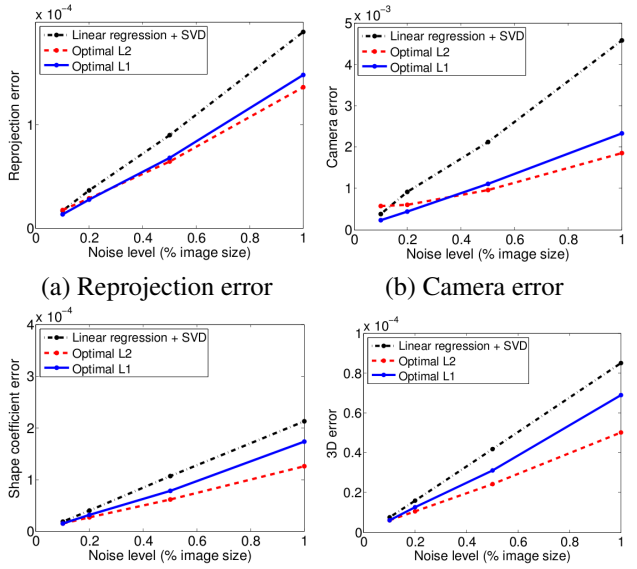
The hat denotes ground truth entities. All the quantitative errors that we report are averaged over 50 trials for each combination of experimental parameters.

We also compare the errors obtained by the optimal L_1 and L_2 algorithms to those obtained by linear regression followed by SVD. Note that camera translation cannot be recovered by SVD as the data needs to be centered. There is also a sign and scale ambiguity in the SVD solution, which is corrected by demanding that all the α^i are non-negative and imposing, post-optimization, the condition that $\sum_i \alpha^i = 1$.

In the first set of experiments, we compute the above errors with $m = 20$ exemplars and $N = 100$ points, for various noise levels from 0.1% to 1% of the image size. The observations are plotted in Figure 1, whereby the L_1 and L_2 algorithms clearly outperform the SVD method, especially for higher noise levels. Note that the L_1 algorithm does not minimize the L_2-reprojection error, so it is not meaningful to compare its image error to that of the other two methods.

The next set of experiments are performed with outliers in the data. For these experiments, $m = 20$, $N = 100$ and

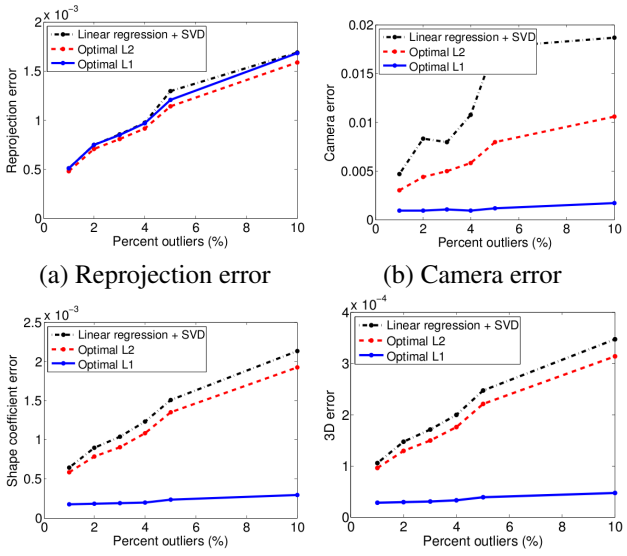
[1] Prototype code at http://vision.ucsd.edu/ bilinear



(a) Reprojection error (b) Camera error

(c) Shape coefficient error (d) 3D error

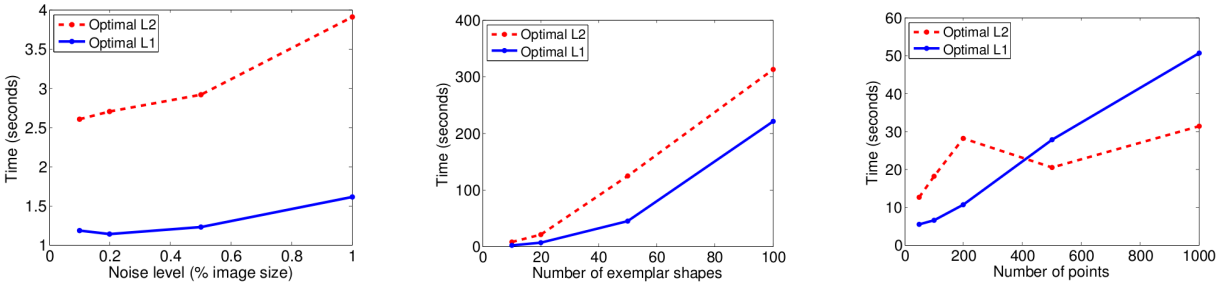
Figure 1. Errors with varying percentage of noise levels for linear regression + SVD (black, dotted curve), the L_2-norm bilinear algorithm (red, dashed curve) and the L_1-norm bilinear algorithm (blue, solid curve). (a) Reprojection error (b) Error in camera entries (c) Error in shape coefficients (d) Error in 3D coordinates of reconstructed shape. All quantities plotted are averages of 50 trials.



(a) Reprojection error (b) Camera error

(c) Shape coefficient error (d) 3D error

Figure 2. Errors with varying percentage of outliers in the data for linear regression + SVD (black, dotted curve), optimal L_2-norm algorithm (red, dashed curve) and optimal L_1-norm algorithm (blue, solid curve). All quantities plotted are averages of 50 trials.



(a) (b) (c)

Figure 3. Time taken to converge to a pre-specified optimality gap by the optimal L_2-norm algorithm (red, dashed curve) and the optimal L_1-norm algorithm (blue, solid curve). (a) 10 exemplars, 100 points and varying noise levels (b) 0.5% noise, 200 points and varying number of exemplars (c) 0.5% noise, 25 exemplars and varying number of points. All quantities plotted are averages of 50 trials.

0.5% Gaussian noise was added to the image coordinates. To simulate an outlier, 10% of image size is added to the image point. The various errors are recorded as percentage of data points corrupted by outliers varies and plotted in Figure 2. It can be seen that the optimal L_2-norm algorithm achieves a lower reprojection error than the SVD method. However, as expected, the robust L_1-norm algorithm significantly outperforms the other two in terms of the geometrically meaningful camera, shape coefficient and 3D errors.

In the next three sets of experiments, we vary the noise levels, number of exemplars and number of points and for each case, record the time for the algorithms to converge to a pre-specified optimality gap (0.001) The graphs in Figure 3 show that the empirical performances of our algorithms scale quite gracefully with increase in problem size or difficulty. This illustrates the utility of our branching strategy which greatly alleviates the worst case complexity of branch and bound by restricting the number of branching dimensions to a small, fixed number irrespective of problem size.

6.2. Applications

The experiments in this section are merely suggestive of a couple of scenarios - namely, face reconstruction using exemplar models and non-rigid structure from motion - where an accurate bilinear fitting is desirable. Both these problems are mature areas of computer vision in their own right. So rather than attempts to beat the state-of-the-art in these specific fields, these experiments must be viewed as demonstrations of the practicality of a fast and robust globally optimal solution in real-world applications.

6.2.1 Face reconstruction using 3D exemplars

We perform real data experiments on the Yale Face Database. 3D exemplar models are constructed by photometric stereo using 9 frontal images of each subject, followed by surface normal integration. The faces were roughly aligned to a common grid prior to photometric stereo using the coordinates of the eye centers and the nose tip.

One of the 35 subjects is chosen for testing, while 3D models of the other 34 subjects are considered exemplars for the bilinear fitting problem. Correspondences are established between a non-frontal image of the test subject and the 3D exemplar models. (Since this is just a demonstration, we circumvent the feature matching stage and simply select 50 correspondences between the test image and a frontal image of the same subject, which gives us 2D-3D correspondences as the exemplars are on the same grid as the frontal image.)

We assume the 3D representation of the input face is a convex combination of the exemplar faces, which turns out to be a good assumption in practice. It is, however, not a requirement of the algorithm and the input shape can as well be assumed to be a linear combination of the elements of a PCA basis derived from the exemplars. The high quality reconstructions obtained by globally minimizing the L_1-norm objective are displayed in Figure 4.

6.2.2 Non-rigid structure from motion

A popular approach to non-rigid structure from motion is to explain the observed structure as a linear combination of elements of a rigid shape basis. Given a shape basis, we can solve the bilinear fitting problem to determine the optimal camera projection and coefficients of the linear combination.

For experimental validation, we use the synthetic shark dataset from [29], which consists of 240 images in various camera orientations of a non-rigid shape consisting of 91 points. A shape basis consisting of 2 elements is obtained using [29]. The algorithm of this paper is applied to minimize reprojection error in the L_2-norm. Sample scatter plots to demonstrate the goodness of the fit are displayed in Figure 5. The average 3D shape error for our algorithm over the whole length of the sequence is 0.9%.

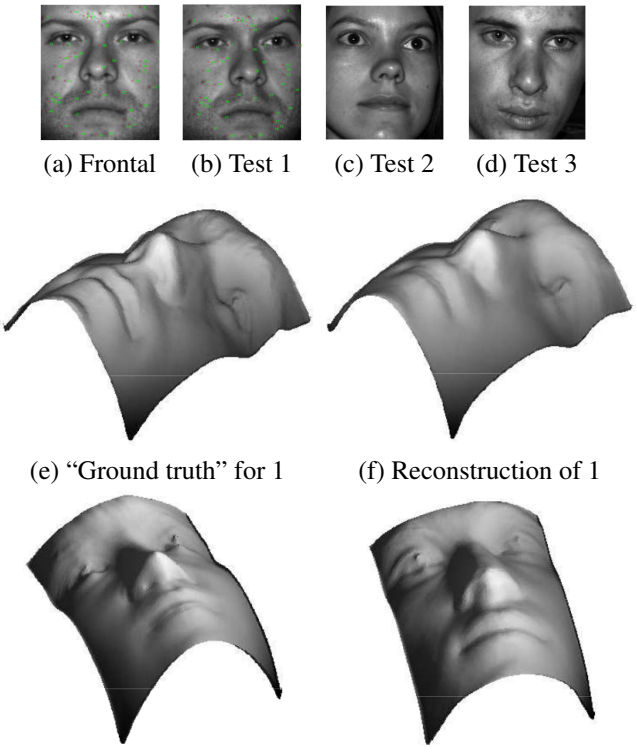
7. Discussions

In this paper, we have proposed a globally optimal algorithm for solving bilinear programs in computer vision, under the standard L_2-norm as well as the robust L_1-norm.

The driving forces of the algorithm are efficient convex relaxations and a branch and bound framework that exploits the problem structure to restrict the branching dimensions to just one set of variables in the bilinear program.

Our branching strategy is provably convergent and particularly advantageous in computer vision problems where the cardinality of one set of variables is much smaller than the other. Note that the uniform convergence requirement proved in Lemma 1 is non-trivial - several popular lower bounding schemes such as LMI relaxations [19] or sum of squares decompositions [23] do not possess this property and thus, cannot be used in a branch and bound framework.

One avenue for future research is extension to multilinearity. While multilinear expressions can always be broken down into a series of bilinear ones, a more sophisticated approach to address trilinearity has a few applications in computer vision. For instance, it will allow texture (or reflectance) to be incorporated into the exemplar-based face reconstruction problem.

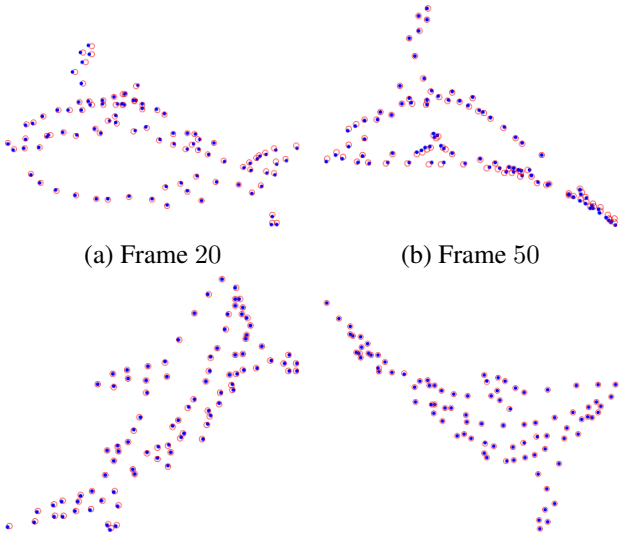


(a) Frontal (b) Test 1 (c) Test 2 (d) Test 3

(e) "Ground truth" for 1 (f) Reconstruction of 1

(e) Reconstruction of 2 (f) Reconstruction of 3

Figure 4. Reconstruction of human faces using 3D exemplars. (a) One of 9 frontal images used for obtaining "ground truth" from photometric stereo. (b) Sample input image for our algorithm, with correspondences marked to the 3D model. (c) Test image 2. (d) Test image 3. (e) "Ground truth" photometric stereo reconstruction for subject 1. (f) Reconstruction for subject 1 obtained using the L_1-norm algorithm in this paper. (g) Reconstruction for subject 2. (h) Reconstruction for subject 3.



(a) Frame 20 (b) Frame 50

(c) Frame 115 (d) Frame 148

Figure 5. 3D scatter plots demonstrating the bilinear fit obtained by the optimal L_2 algorithm for representing a deforming shark as a linear combination of rigid basis shapes. Red circles indicate ground truth and blue dots indicate the reconstruction.

Appendix

Given Lemma 1, it can be shown, similar to [4, 13], that the branch and bound algorithm proposed in Section 5 converges.

Define the condition number of a rectangle as $\mathcal{C}(Q) = \max_i(u_i - l_i)/\min_i(u_i - l_i)$, where l_i and u_i are bounds on the x_i terms only. Then, the following is true for our choice of sub-division rule:

Lemma 2. $\mathcal{C}(Q) \leq \max\{\mathcal{C}(Q_0), 2\}$

Proof. Let some $Q \in \mathcal{P}$ be split into Q_1 and Q_2 by bisection along the longest x_i-edge. Let $\chi'(Q) = \min_i(u_i - l_i)$ and as before, $\chi(Q) = \max_i(u_i - l_i)$. Then, $\chi(Q_1) \leq \chi(Q)/2$ and $\chi(Q_2) \leq \chi(Q)/2$. Moreover, $\chi'(Q_1) = \chi'(Q_2) = \min\{\chi(Q)/2,\ \chi'(Q)\}$. It follows that $\mathcal{C}(Q_1) \leq \max\{\mathcal{C}(Q), 2\}$ and $\mathcal{C}(Q_2) \leq \max\{\mathcal{C}(Q), 2\}$ and as a consequence, the lemma is proved. □

Lemma 3. *After a "large enough" number of branching operations, there exists at least one "small enough" rectangle. More precisely, there exists a rectangle for which the maximum length along dimensions corresponding to x_i, $i = 1, \cdots, n$, can be bounded in terms of the number of branching operations.*

Proof. The volume of a rectangle is given by

$$\begin{aligned}\text{vol}(Q) &= \textstyle\prod_i(u_i - l_i)\prod_i(U_i - L_i) = V_0^n \prod_i(u_i - l_i) \\ &\geq V_0^n \chi(Q) \cdot \left[\chi'(Q)\right]^{n-1} \ \geq\ V_0^n \frac{[\chi(Q)]^n}{[\mathcal{C}(Q)]^{n-1}} \\ &\geq \left(\frac{V_0 \cdot \chi(Q)}{\mathcal{C}(Q)}\right)^n, \quad [\text{since } \mathcal{C}(Q) \geq 1]\end{aligned}$$

where $V_0^n = \prod_i (U_i - L_i)$ remains constant throughout the branching process. Thus, $\chi(Q) \leq \frac{1}{V_0}\,\mathcal{C}\,(Q)\,\mathrm{vol}\,(Q)^{1/n}$.

Since a new rectangle is created by bisection at every branching stage, after k branching steps, there exists some $Q' \in \mathcal{P}_k$ such that $\mathrm{vol}\,(Q') \leq \frac{\mathrm{vol}\,(Q_0)}{k}$. Using Lemma 2, it follows that there exists some $Q' \in \mathcal{P}_k$ such that

$$\chi(Q') \leq \frac{1}{V_0} \max\{\mathcal{C}\,(Q_0), 2\}[\mathrm{vol}\,(Q_0)/k]^{1/n} \qquad (16)$$

and the lemma stands proved. □

We are now in a position to prove Proposition 1. Specifically, let $\psi^l_{k'}$ and $\psi^u_{k'}$ in equation (1) be the global lower and upper bounds maintained by the branch and bound algorithm at the end of k' iterations. Then, we show that given $\epsilon > 0$, $\exists k$ such that for some $k' < k$, it was the case that $\psi^u_{k'} - \psi^l_{k'} \leq \epsilon$.

Proof. Lemma 1 guarantees the existence of $\delta > 0$ such that $\chi(Q) \leq \delta \Rightarrow \Phi_{\mathrm{ub}}\,(Q) - \Phi_{\mathrm{lb}}\,(Q) \leq \epsilon$. From Lemma 3, for a large enough k such that

$$\frac{1}{V_0} \max\{\mathcal{C}\,(Q_0), 2\}[\mathrm{vol}\,(Q_0)/k]^{1/n} \leq \delta/2\,, \qquad (17)$$

there exists a rectangle $Q' \in \mathcal{P}_k$ such that $\chi(Q') \leq \delta/2$. Let Q' be formed by sub-division of some rectangle Q''. Then, $\chi(Q'') \leq \delta$, thus, from Lemma 1, $\Phi_{\mathrm{ub}}\,(Q'') - \Phi_{\mathrm{lb}}\,(Q'') \leq \epsilon$. Since the algorithm chose Q'' for branching at some iteration k', it must have satisfied $\Phi_{\mathrm{lb}}\,(Q'') = \psi^l_{k'}$. Thus,

$$\psi^u_{k'} - \psi^l_{k'} \leq \Phi_{\mathrm{ub}}\,(Q'') - \psi^l_{k'} \leq \Phi_{\mathrm{ub}}\,(Q'') - \Phi_{\mathrm{lb}}\,(Q'') \leq \epsilon \quad (18)$$

and the convergence proof stands completed. □

References

[1] S. Agarwal, M. Chandraker, F. Kahl, S. Belongie, and D. Kriegman. Practical global optimization for multiview geometry. In *ECCV*, pages 592–605, 2006.

[2] F. Al-Khayyal and J. Falk. Jointly constrained biconvex programming. *Math. Oper. Res.*, 8:273–286, 1983.

[3] E. Andersen, C. Roos, and T. Terlaki. On implementing a primal-dual interior-point method for conic quadratic optimization. *Math. Prog.*, 95(2):249–277, 2003.

[4] V. Balakrishnan, S. Boyd, and S. Balemi. Branch and bound algorithm for computing minimum stability degree of parameter-dependent linear systems. *Int. J. Rob. & Non-lin. Cont.*, 1(4):295–317, 1991.

[5] B. Bascle and A. Blake. Separability of pose and expression in facial tracking and animation. In *ICCV*, pages 323–328, 1998.

[6] P. Belhumeur and D. Kriegman. What is the set of images of an object under all possible lighting conditions? In *CVPR*, pages 270–277, 1996.

[7] V. Blanz and T. Vetter. A morphabe model for the synthesis of 3D faces. In *SIGGRAPH*, pages 187–194, 1999.

[8] M. Chandraker, S. Agarwal, D. Kriegman, and S. Belongie. Globally optimal affine and metric upgrades in stratified autocalibration. In *ICCV*, 2007.

[9] W. Freeman and J. Tenenbaum. Learning bilinear models for two-factor problems in vision. In *CVPR*, pages 554–560, 1997.

[10] A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti. Globally convergent autocalibration using interval analysis. *PAMI*, 26(12):1633–1638, 2004.

[11] P. Hallinan. A low-dimensional representation of human faces for arbitrary lighting conditions. In *CVPR*, pages 995–999, 1994.

[12] D. Henrion and J. B. Lasserre. GloptiPoly: Global optimization over polynomials with Matlab and SeDuMi. *ACM Trans. Math. Soft.*, 29(2):165–194, 2003.

[13] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer Verlag, 2006.

[14] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, Inc., 2001.

[15] F. Kahl. Multiple view geometry and the L_∞-norm. In *ICCV*, pages 1002–1009, Beijing, China, 2005.

[16] F. Kahl and D. Henrion. Globally optimal estimates for geometric reconstruction problems. In *ICCV*, pages 978–985, 2005.

[17] J. Koenderink and A. van Doorn. The generic bilinear calibration-estimation problem. *IJCV*, 23(3):217–234, 1997.

[18] H. Konno. A cutting plane algorithm for solving bilinear programs. *Math. Prog.*, 11:14–27, 1976.

[19] J. Lasserre. Global optimization with polynomials and problem of moments. *SIAM J. Opt.*, 11(3):796–817, 2001.

[20] D. Marimont and B. Wandell. Linear models of surface and illuminant spectra. *JOSA*, 9(11):1905–1913, 1992.

[21] G. McCormick. Computability of global solutions to factorable nonconvex programs. *Math. Programming*, 10:147–175, 1976.

[22] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *IJCV*, 14(1), 1995.

[23] P. Parrilo. *Sum of squares decompositions for structured polynomials via semidefinite programming*. PhD thesis, TU München, 2003.

[24] Y. Seo and R. Hartley. A fast method to minimize L-infinity error norm for geometric vision problems. In *ICCV*, 2007.

[25] H. Sherali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems. *J. Glob. Opt.*, 2(4):379–410, 1992.

[26] H. Stewénius, F. Schaffalitzky, and D. Nistér. How hard is three-view triangulation really? In *ICCV*, pages 686–693, 2005.

[27] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. *Neu. Comp.*, 12(6):1247–1283, 2000.

[28] C. Tomasi and T. Kanade. Shape & motion from image streams under orthography: a factorization method. *IJCV*, 9(2):137–154, 1992.

[29] L. Torresani, A. Hertzmann, , and C. Bregler. Learning non-rigid 3d shape from 2d motion. In *NIPS*, pages 1555–1562, 2003.

[30] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L1 range image integration. In *ICCV*, 2007.