General and Nested Wiberg Minimization: L_2 and Maximum Likelihood

Dennis Strelow

Google,
Mountain View, CA
strelow@google.com

Abstract. Wiberg matrix factorization breaks a matrix Y into low-rank factors U and V by solving for V in closed form given U, linearizing $V(U)$ about U, and iteratively minimizing $||Y - UV(U)||_2$ with respect to U only. This approach factors the matrix while effectively removing V from the minimization. We generalize the Wiberg approach beyond factorization to minimize an arbitrary function that is nonlinear in each of two sets of variables. In this paper we focus on the case of L_2 minimization and maximum likelihood estimation (MLE), presenting an L_2 Wiberg bundle adjustment algorithm and a Wiberg MLE algorithm for Poisson matrix factorization. We also show that one Wiberg minimization can be nested inside another, effectively removing two of three sets of variables from a minimization. We demonstrate this idea with a nested Wiberg algorithm for L_2 projective bundle adjustment, solving for camera matrices, points, and projective depths.

1 Introduction

Matrix factorization breaks a matrix Y into low-rank factors U and V by minimizing $||Y - UV||_2$ with respect to U and V. If Y is complete, the singular value decomposition factors Y effectively. If Y has missing entries, U and V can be estimated by minimizing with respect to all of U and V's entries simultaneously, or by repeatedly solving for U holding V fixed, then V holding U fixed.

Wiberg[1] proposed a third approach for Y with missing entries, which effectively eliminates V from the problem: solve for V given U in closed form, linearize $V(U)$ about U, and iteratively minimize $||Y - UV(U)||_2$ with respect to U only. Okatani and Deguchi[2] and Okatani, Yoshida, and Deguchi[3] showed that Wiberg's approach converges better than simultaneous minimization with Levenberg-Marquardt and other algorithms. More recently, Eriksson and van den Hengel[4] extended Wiberg's approach to L_1 matrix factorization. They showed that their L_1-Wiberg factorization converged better than the previous state-of-the-art for L_1, alternating convex programming.

In this paper we generalize the Wiberg approach beyond factorization to minimize an arbitrary nonlinear function of two sets of variables, $f(U, V)$. Our general Wiberg minimization can be used for L_1 minimization, L_2 minimization, or maximum likelihood estimation (MLE), and Table 1 shows general Wiberg's place in

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part VII, LNCS 7578, pp. 195–207, 2012.

Table 1. Optimization problems in two sets of variables U, V and possible approaches for solving them. Our general Wiberg method greatly extends the applicability of the Wiberg approach, as shown by the red boxes. This paper describes general Wiberg for L_2 minimization and maximum likelihood estimation ("$\star$" in the table), while a recent companion paper[5] describes general Wiberg for L_1 minimization.

	linear in U or V		
	minimize L_2	minimize L_1	MLE
simultaneous	Gauss-Newton	successive LP	Newton-Raphson
alternating	alternating LS	alternating LP	EM
Wiberg	Wiberg 1976	Eriksson 2010	$\star$ general Wiberg

	nonlinear in both U and V		
	minimize L_2	minimize L_1	MLE
simultaneous	Gauss-Newton	successive LP	Newton-Raphson
alternating	alt. Gauss-Newton	alt. succ. LP	EM
Wiberg	$\star$ general Wiberg	general Wiberg	$\star$ general Wiberg

the space of optimization problems. In this paper we focus on L_2 minimization and maximum likelihood estimation, demonstrating these with an L_2-Wiberg bundle adjustment algorithm and Wiberg maximum likelihood estimation for Poisson matrix factorization, respectively.

Our general Wiberg minimization works by solving for V iteratively rather than in closed form. Since it is found iteratively, V can itself be split into two sets of variables found using Wiberg minimization. This results in a nested Wiberg minimization that can effectively minimize with respect to three sets of variables. We demonstrate this idea with an L_2-Wiberg algorithm for projective (uncalibrated) bundle adjustment, solving for camera matrices, points, and projective depths.

Our main contributions are general and nested Wiberg L_2 minimization; general and nested L_2-Wiberg algorithms for bundle adjustment and projective bundle adjustment, respectively; Wiberg maximum likelihood estimation; and a Wiberg maximum likelihood algorithm for Poisson matrix factorization.

A companion paper[5] describes how L_1 general and nested Wiberg are derived from Eriksson and van den Hengel's L_1-Wiberg matrix factorization[4]. In contrast, this paper derives and explores the more practical L_2 and maximum likelihood Wiberg algorithms.

2 Related Work

Wiberg[1] presented an L_2 factorization algorithm for matrices with missing data, which solved for one set of variables V in terms of the other U, linearized V about U, and then minimized with respect to U only. Okatani and Deguchi[2] and Okatani, Yoshida, and Deguchi[3] showed that Wiberg factorization converged better than minimizing with respect to U and V simultaneously with Levenberg-Marquardt, and argued that Wiberg's method had been neglected by the computer vision community.

Recently, Eriksson and van den Hengel[4] extended this approach to L_1 matrix factorization using linear programming. Their method outperformed Ke and Kanade's alternating convex programming algorithms[6], establishing a new state-of-the-art for L_1 factorization. Eriksson and van den Hengel did not compare their method against a simultaneous minimization method, but in our own experiments Eriksson and van den Hengel's method performed better than minimizing with respect to all of the unknowns simultaneously using successive linear programming. In a recent companion paper[5], we extended Eriksson and van den Hengel's L_1-Wiberg factorization to minimize general nonlinear functions of two sets of variables, analogous to the L_2 and maximum likelihood general Wiberg we present here.

Wiberg's method was an application of Ruhe and Wedin's[7] more general work on separable nonlinear minimization that solved for a V in terms of U and then minimized with respect to U only. Ruhe and Wedin recognized that this approach would be advantageous when V breaks down into small independent problems given U, which happens in all the problems in this paper. But, their analysis and experiments focused on least squares objectives linear in V. In even earlier work, Richards[8] described a separable method for maximum likelihood estimation, but similarly demonstrated it only on a least squares problem linear in V. In contrast, we consider more general functions that can be nonlinear in both U and V.

The Wiberg approach contrasts with alternating least squares and similar methods, which alternate between solving for one set of unknowns while holding the other fixed. Alternating methods sometimes converge well, but they can also converge very slowly[2] or fail to converge "catastrophically"[9]. For this reason, we've bypassed alternating methods as baseline algorithms, instead choosing minimization with respect to all of the unknowns simultaneously.

3 General Wiberg Minimization: L_2

In this section we briefly review Wiberg matrix factorization and then generalize Wiberg factorization to minimize arbitrary nonlinear functions of two sets of variables. We call the resulting algorithm general Wiberg minimization. As an example of this idea, we implement bundle adjustment as a general Wiberg minimization.

Our algorithms use matrix calculus, which Fackler[10] summarizes well. But, most derivatives that we'll encounter – derivatives of a matrix or derivatives with respect to a matrix – can be handled by flattening the matrix by column and then using the normal rules for vector calculus.

3.1 Wiberg Matrix Factorization

Matrix factorization breaks a matrix Y into low-rank factors U and V by minimizing $||Y - UV||_2$ with respect to U and V. The Wiberg approach to factorization effectively eliminates V from the problem, by solving for V given U in closed

form, linearizing $V(U)$ about U, and iteratively minimizing $||Y - UV(U)||_2$ with respect to U only. In this section we briefly present each of these steps; see [2] and [3] for a more detailed discussion.

We've simplified the description slightly by assuming that Y is full. Changing the derivation to explicitly handle Y with missing entries introduces some non-essential notation, but is straightforward and in our bundle adjustment experiments below we include problems with missing observations.

Solve for V given U. Given an estimate of U, estimate each column v_j of V independently using the normal equations $Av_j = b$, where $A = U^{\mathrm{T}}U$ and $b = U^{\mathrm{T}}y_i$.

Linearize V(U) with respect to U. Then,

$$\frac{dv_j}{dA} = -v_j^{\mathrm{T}} \otimes A^{-1} \tag{1}$$

$$\frac{dv_j}{db} = A^{-1} \tag{2}$$

where $\otimes$ is the Kronecker product. Since each element of A is the dot product of a column of U and a row of U, and b is the dot product of a column of U and y_i, the derivatives dA/dU and db/dU are straightforward. Then,

$$\frac{dv_j}{dU} = \frac{dv_j}{dA}\frac{dA}{dU} + \frac{dv_j}{db}\frac{db}{dU} \tag{3}$$

Minimize with respect to U only. Given estimates U and v_j, our approximation or prediction $p_{i,j}$ of $Y_{i,j}$ is $u_i v_j$, where u_i is row i of U. Then,

$$\frac{dp_{i,j}}{dU} = \frac{\partial p_{i,j}}{\partial U} + \frac{\partial p_{i,j}}{\partial v_j}\frac{dv_j}{dU} \tag{4}$$

where

$$\frac{\partial p_{i,j}}{\partial u_i} = v_j^{\mathrm{T}} \qquad \frac{\partial p_{i,j}}{\partial v_j} = u_i \tag{5}$$

and the partial derivatives $\partial p_{i,j}$ with respect to the other components of U are zero.

Together, the errors $Y_{i,j} - p_{i,j}$ and the first derivatives (4) are used to construct the gradient and approximate Hessian for Levenberg-Marquardt, estimating U only. Press *et al.*[11] give a good overview of Levenberg-Marquardt.

3.2 General Wiberg Minimization

Wiberg factorization solves for V using the normal equations but solves for U using Levenberg-Marquardt. So, adapting the algorithm to minimize a nonlinear function of U is straightforward – possibly just by changing a few lines of code – as long as the function is linear in V. But many functions are nonlinear in two sets of variables. In bundle adjustment, for instance, the objective function is

a sum of reprojection errors, which are nonlinear in both the three-dimensional point positions and the six-degree-of-freedom camera positions.

To handle objectives like these, we use Levenberg-Marquardt for V as well as U. With this approach, we have an outer loop that minimizes with respect to U, and within each U iteration, we have an inner loop that minimizes with respect to V. This method is best suited for problems like bundle adjustment (and factorization) where given U, V breaks down into independent subproblems v_c. In this case the time for iteratively solving for the v_c is small because each v_c is much smaller than U.

But in the Wiberg approach, the v_c's vary implicitly with U via dv_c/dU. How do we find dv_c/dU if we found v_c iteratively? In short, each Levenberg-Marquardt step is found in closed form, and the derivative of v_c is the derivative of the final Levenberg-Marquardt step δv_c:

$$\frac{dv_c}{dU} = \frac{d\delta_{v_c}}{dU} \tag{6}$$

$$= \frac{d\delta_{v_c}}{d\text{Hessian}} \frac{d\text{Hessian}}{dU} \tag{7}$$

$$+ \frac{d\delta_{v_c}}{d\text{gradient}} \frac{d\text{gradient}}{dU} \tag{8}$$

The derivatives of δv_c with respect to the Hessian and gradient are found similarly to the derivative of v_j with respect to A and b in Wiberg factorization. The Hessian and gradient each include derivatives of the predictions with respect to v_c as factors, so the derivatives of the Hessian and gradient with respect to U include second derivatives with respect to v_c and U as factors.

Once we have the v_c's and dv_c/dU's, the derivatives of the predictions with respect to U are found similarly to (4):

$$\frac{dp_i}{dU} = \frac{\partial p_i}{\partial U} + \frac{\partial p_i}{\partial V}\frac{dV}{dU} \tag{9}$$

We can then minimize with respect to U by using this derivative in the Levenberg-Marquardt minimization for U, just as we did in the Wiberg factorization, using (9) to construct the Hessian and gradient.

3.3 Bundle Adjustment

Bundle adjustment is the go-to algorithm for structure-from-motion. Given two-dimensional observations $x_{i,j}$ of three-dimensional points in an image collection, bundle adjustment estimates the three-dimensional position of the each point X_j and the six-degree-of-freedom position (rotation ρ_i and translation t_i) of the camera for each image, by minimizing:

$$\sum_{i,j} (x_{i,j} - \pi(R(\rho_i)X_j + t_i))^2 \tag{10}$$

where π is the perspective projection and $R(\rho_i)$ is the rotation matrix for Euler angles ρ_i. In Section 6.1 below, we'll investigate minimizing (10) with general

Wiberg, with the camera variables in the outer loop and the point variables in the inner loop.

Normally (10) is minimized with Levenberg-Marquardt. Each iteration of Levenberg-Marquardt solves for a step $[\delta_C; \delta_S]$ in the camera components C and structure (point) components S:

$$\begin{bmatrix} H_{CC} & H_{CS} \\ H_{SC} & H_{SS} \end{bmatrix} \begin{bmatrix} \delta_C \\ \delta_S \end{bmatrix} = - \begin{bmatrix} g_C \\ g_S \end{bmatrix} \tag{11}$$

One way to solve (11) is to first solve a reduced camera system[12] for δ_C:

$$\overline{H}_{CC} \delta_C = -\overline{g}_{CC} \tag{12}$$

$$\overline{H}_{CC} = H_{CC} - H_{CS} H_{SS}^{-1} H_{SC} \tag{13}$$

$$\overline{g}_{CC} = g_C - H_{CS} H_{SS}^{-1} g_S \tag{14}$$

and then find δ_S by back-substitution. This produces the same step as solving (11) directly.

General Wiberg and the reduced camera system both remove the points from the problem, and the sparse structures of the general Wiberg and reduced camera system Hessians are the same. Further, the actual values in the two systems are close if the errors in the observations and estimates are small. But in general, the two systems differ. The algorithms also differ in that given the most recent camera variable estimates, Wiberg finds optimal point estimates using the inner iteration rather than accepting the δ_S from back-substitution.

In contrast, Okatani *et al.*[3] noted that for the specific problem of matrix factorization, applying the reduced camera system idea produces the same system and step for U as Wiberg. So, Wiberg's advantage over Levenberg-Marquardt for matrix factorization appears to be Wiberg's optimal resolve for V given the new U on each iteration.

4 General Wiberg Minimization: Maximum Likelihood Estimation

The general Wiberg minimization in Section 3.2 minimizes least squares error, using Levenberg-Marquardt in both the inner and outer iteration. A common, more general problem is minimizing a negative log likelihood (NLL) for maximum likelihood estimation (MLE). In this section, we present a general Wiberg algorithm for minimizing negative log likelihoods. This MLE algorithm is similar to the least squares algorithm, but replaces Levenberg-Marquardt with Newton-Raphson and requires d^2V/dU^2, which does not appear in the previous algorithms.

Wiberg MLE has the same overall structure as least squares general Wiberg – an inner iteration solves for V while an outer iteration solves for U. Each of these iterations repeatedly solves a linear system $H\delta = -g$ for a step δ in the

estimate, where g and H are the first and second derivatives of the error (for least squares) or NLL (for MLE).

Levenberg-Marquardt is specific to least squares and approximates the second derivative matrix H as the product of two first derivatives[11]. In contrast, Newton-Raphson requires full second derivatives. The second derivative of NLL with respect to V for the inner iteration is straightforward, while the second derivative with respect to U is challenging. The first derivative with respect to U is similar in form to the first derivative of the predictions in our least squares problems (4), (9):

$$\frac{d\text{NLL}}{dU} = \frac{\partial \text{NLL}}{\partial U} + \frac{\partial \text{NLL}}{\partial V}\frac{dV}{dU} \tag{15}$$

Taking the derivative of (15) with respect to U again gives us the second derivative:

$$\frac{d^2\text{NLL}}{dU^2} = \frac{d(\partial \text{NLL}/\partial U)}{dU} + \frac{d(\partial \text{NLL}/\partial V * dV/dU)}{dU} \tag{16}$$

Evaluating (16) introduces d^2V/dU^2, which is not present in Wiberg matrix factorization or general Wiberg least squares.

To find dV/dU and d^2V/dU^2, we'll take the derivatives of V to be the derivatives of the final Newton-Raphson step for V, just as we did in the least squares algorithm. Here, the step is given by $H_V \delta V = -g_V$, where H_V is the full second derivative matrix of NLL with respect to V, and g_V is the first derivative with respect to V. Then, dV/dH_V and dV/dg_V are similar to (1) and (2), respectively:

$$\frac{dV}{dH_V} = -V^{\mathrm{T}} \otimes H_V^{-1} \tag{17}$$

$$\frac{dV}{dg_V} = H_V^{-1} \tag{18}$$

Working from these, we can then find dV/dU and d^2V/dU^2 by considering H_v and g_v as functions of U.

4.1 Poisson Matrix Factorization

Often we want to factor a Poisson-distributed matrix A into low-rank, nonnegative factors U, V[13]. If we enforce nonnegativity (or strictly speaking, positivity) by optimizing with respect to the elementwise logarithms $\log(U)$ and $\log(V)$, then maximum likelihood estimates can be found by minimizing the negative log likelihood:

$$\text{NLL} = \sum_{i=1}^{m}\sum_{j=1}^{n}(x_{ij} - a_{ij}\log x_{ij}) \tag{19}$$

where $X = \exp(\log(U))\exp(\log(V))$ and $\log()$ and $\exp()$ are elementwise. (19) can be minimized with the general Wiberg MLE approach in this section, and in Section 6.3 below, we'll compare Wiberg and Newton-Raphson for this problem.

5 L_2 Nested Wiberg Minimization

Our general Wiberg minimization in Section 3.2 works by solving for V iteratively rather than in closed form. Since it is found iteratively, V can itself be split into two sets of variables found using the Wiberg approach. This results in a nested Wiberg minimization that can effectively minimize with respect to three sets of variables. In this section, we demonstrate this idea on projective structure-from-motion, where the three sets of unknowns are camera matrices, three-dimensional point positions, and projective depths.

So suppose we have three sets of variables U, V, and D; that given U and V we minimize with respect to D in closed form; that given U we minimize with respect to V and D in an inner iteration; and that we minimize with respect to U in an outer iteration. Then to minimize with respect to U using Levenberg-Marquardt, we'll need the total derivative of our predictions p with respect to U:

$$\frac{dp}{dU} = \frac{\partial p}{\partial U} + \left(\frac{\partial p}{\partial V} + \frac{\partial p}{\partial D}\frac{dD}{dV} \right) \frac{dV}{dU} + \frac{\partial p}{\partial D}\frac{dD}{dU} \tag{20}$$

Equation (9) is a total derivative of p with respect to U, with V a function of U. Equation (20) generalizes this to the total derivative of p with respect to U, with V a function of U and D a function of both U and V. The factor in parentheses is the total derivative of p with respect to V, with D a function of V, and reflects the nesting.

Deriving (20) requires us to expand a complex tree of derivatives. Because of limited space, we can't completely explore that tree here. But to suggest the full procedure, we summarize one path from the root of this tree as follows:

1. (20) includes the factor dV/dU.
2. dV/dU is similar to equation (6-8), and includes $d(dp(v_c)/dv_c)/dU$.
3. $dp(v_c)/dv_c$ is a total derivative and includes the factor dD/d_{v_c}.
4. D and the derivatives of D with respect to the system that produces it are found in closed form.
5. That coefficient matrix and right-hand side of the system for D are functions of v_c, so we can use the chain rule to get the derivative of D with respect to v_c.

The derivatives at the other leaves are found similarly and combined using the rules for derivatives of matrix expressions[10].

5.1 Projective Bundle Adjustment

The bundle adjustment in Section 3.3 can be used when the camera calibration (e.g., focal length) is known. In contrast, projective bundle adjustment recovers structure and motion from uncalibrated image sequences. A projective reconstruction includes 3×4 camera projection matrices C_i and 4-dimensional projective points X_j that are consistent with the image observations and are known up to a common 4×4 transformation. This transformation can be identified, and

the projective reconstruction upgraded to a metric reconstruction, given some knowledge of the scene geometry or camera intrinsics.

Our objective is:

$$\sum_{i,j} ||[u_{i,j}\ v_{i,j}\ 1]^{\mathrm{T}} - d_{i,j}C_iX_j||_1 \tag{21}$$

where $(u_{i,j}, v_{i,j})$ and $d_{i,j}$ are the two-dimensional image location and inverse projective depth of point j in image i. We can minimize (21) with respect to C_i, X_j, and $d_{i,j}$, using either nested Wiberg minimization or simultaneous minimization.

We present results for both Wiberg and simultaneous minimization in Section 6.2 below. For Wiberg, we've chosen to find each inverse depth independently given point and projection matrix estimates, in closed form; to find each projection matrix given point estimates using an inner Wiberg minimization, letting the inverse depths vary implicitly; and to solve for the points in an outer Wiberg minimization, letting the projection matrices and inverse depths vary implicitly.

Given point and projection matrix estimates, it's also possible to "read off"[14] the projective depths as the last element of CX rather than estimating them. This results in the projective bundle adjustment algorithm given by Hartley and Zisserman[15]. Here, we explicitly estimate the inverse depths as an example of a nested Wiberg minimization. The objective using this approach is similar to that in factorization methods for projective structure-from-motion, which do require that the depths be explicitly estimated. Oliensis and Hartley[16] also perform an L_2 projective bundle adjustment by minimizing with respect to the projection matrices, points, and depths.

6 Results

6.1 Bundle Adjustment

In this section we compare bundle adjustment using Wiberg and Levenberg-Marquardt with synthetic experiments, and recover structure and motion from a real image sequence using the Wiberg algorithm.

We conducted two sets of synthetic experiments. In the first, Bundle Adjustment 1, we used points uniformly distributed on a sphere of radius 10, and cameras moving in a ring of radius 20 around the sphere's center, looking inward. We varied the number of images (2, 3, 5, 7, 11, 18), the number of points (10, 23, 54, 128, 300), and the error in the initial estimates of the camera rotation Euler angles, camera translation, and three-dimensional point positions. We drew the errors in the initial estimates from $[-\epsilon, \epsilon]$, for 10 ϵ's varying exponentially between 10^{-3} and 10^{1}. The larger ϵ's near 10^{1} are larger than we would expect from a reasonable initial estimate, e.g., from a structure-from-motion extended Kalman filter[17]. In each trial, we ran both Wiberg and Levenberg-Marquardt for 50 iterations, using the same noisy initial estimates for both methods. In this experiment, Wiberg and Levenberg-Marquardt both converge to the correct estimate in a few iterations in every trial.

However, Wiberg and Levenberg-Marquardt begin to differ in two more extreme versions instances of this problem. First, for $\epsilon > 1$, Wiberg's inner point

solves produce some points at "infinity," preventing convergence. In contrast, Levenberg-Marquardt converges reliably up to about $\epsilon = 10$, because the single damping factor in joint step for the cameras and points prevents the point estimates from going wild. Second, if we require a very strict residual threshold for convergence, we find that Wiberg sometimes converges in many fewer iterations than Levenberg-Marquardt. Of the 300 trials, both methods converged to this strict threshold in 166 trials; Levenberg-Marquardt converged to the threshold faster than Wiberg in one trial; and Wiberg converged to the threshold faster than Levenberg-Marquardt in 149 trials. Often Wiberg finished in many fewer iterations, as shown in Figure 1.

In the second suite of synthetic experiments, Bundle Adjustment 2, we investigated the effects of occlusion by varying the percentage of visible projections. We generated a realistic occlusion pattern by having a camera with a finite field of view flying over a terrain-like set of points. We again ran 300 trials, using 20 images in each; varying the number of points (100, 131, 173, 227, 300); the fraction of visible projections (.5, .75, 1.0); and the random errors in the initial estimates. For the random errors, we used the same distribution of ϵ's as in Bundle Adjustment 1 above.

The results are similar to Bundle Adjustment 1. Both methods converge in each trial in a few iterations. For a version of the experiment with larger errors in the initial estimates, Levenberg-Marquardt converges for slightly higher initial errors than Wiberg. For the most interesting case of 0.5 visible projections, Wiberg converged for errors up to $\epsilon = 1.94$, whereas Levenberg-Marquardt typically converged for errors up to $\epsilon = 5.15$. For a version of the experiment with a very strict convergence threshold, Wiberg converges faster. Again for 0.5 visible projections, both methods converged to the strict threshold in 201 cases; Levenberg-Marquardt converged faster in one case; Wiberg converged faster in 200 cases. The difference in the number of iterations is again shown in Figure 1.

Figure 2 shows an example image from a real sequence, "perspective rover," with tracked points shown as black dots. The camera looks out from the back of a rover while the rover executes three large loops. The sequence includes about 700 images and about 10,000 three-dimensional points.

The Wiberg bundle adjustment algorithm correctly recovers the structure and motion, and the estimated motion is shown in Figure 2. The result is locally correct and consistent with the global motion estimates from GPS and odometry. We picked this example because it cannot be solved using factorization and affine structure-from-motion – perspective effects are extremely strong because of the large difference in distance to the near and far points.

6.2 Projective Bundle Adjustment

We conducted two experiments comparing the nested Wiberg algorithm for projective bundle adjustment in Section 5.1 with minimizing with respect to all of the unknowns simultaneously with Levenberg-Marquardt. The first, Projective Bundle Adjustment 1, uses 300 trials with the same points and camera positions as Bundle Adjustment 1 above. We drew the errors in the initial point and

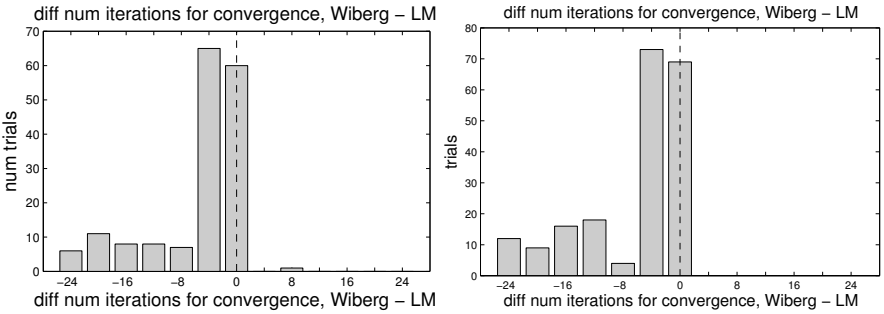

Fig. 1. Histograms of the differences in the number of iterations required for convergence to a strict residual threshold, Wiberg - Levenberg-Marquardt, for bundle adjustment. The left plot shows the differences for Bundle Adjustment 1 (points on a sphere) and the right plot shows the differences Bundle Adjustment 2 (occlusion) with fraction of data visible = 0.5 in on the right. For this strict threshold, Wiberg often convergences in many fewer iterations than Levenberg-Marquardt.

camera estimates from $[-\epsilon, \epsilon]$, for 10 ϵ's varying exponentially between 10^{-2} and 10^{1}. Our initial estimates for the inverse projective depths was 1.0, which is the estimate normally used by projective factorization algorithms. Both methods converged for all values of ϵ except $\epsilon = 10$, where Wiberg failed for four trials and Levenberg-Marquardt failed for one trial.

The second experiment, Projective Bundle Adjustment 2, is identical to Projective Bundle Adjustment 1 except that the camera ring radius is shorted to 10.1 – just 0.1 units from the point sphere. In this case, the default estimate of 1.0 for the inverse depths is poor, and Levenberg-Marquardt often fails for all ϵ. In contrast, Wiberg doesn't begin to fail until $\epsilon = 2.15$. The Wiberg approach - finding optimal inverse depths and camera positions given the point estimates, and allowing them to vary implicitly with the point estimates - is more robust than finding a joint step for the inverse depths, cameras, and points.

6.3 Poisson Matrix Factorization

We compared the Wiberg Poisson matrix factorization algorithm with damped Newton-Raphson. We performed 100 trials of factoring a 25×25 matrix into rank 3 matrices. Our ground truth factors had random elements in the range $[0, 1]$, and the noise in our initial estimates varied across the 100 trials, varying exponentially between 10^{-2} and 10^{1}. Wiberg started to fail occasionally at $\epsilon = 0.081$ and routinely at $\epsilon = 3.05$, which are extremely large initial errors relative to the actual range of $[0, 1]$. However, in the cases where Wiberg does converge, it converges to zero. In contrast, Newton-Raphson fails more gracefully, but produced smallish residuals for all ϵ's rather than converging to zero.

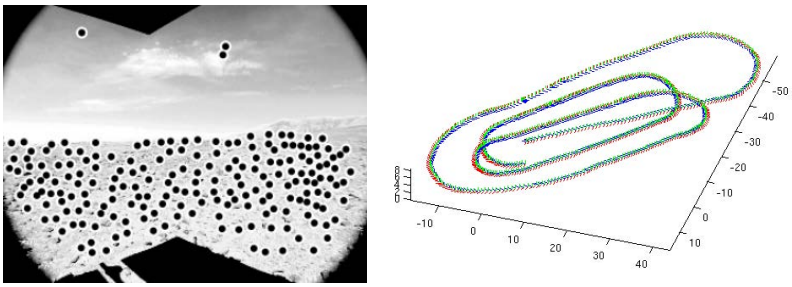

Fig. 2. Wiberg bundle adjustment reconstructs the correct structure and motion from the "rover" sequence, which includes about 700 images and 10,000 points. Left: an example image from the sequence, with tracked points shown as black dots. Right: an oblique view of the recovered camera positions at the time of each image.

7 Conclusion

We've introduced general and nested Wiberg minimization, which extend Wiberg's approach to matrix factorization to general functions that are nonlinear in two or three sets of variables. More specifically, in this paper we've explored Wiberg algorithms for L_2 minimization and maximum likelihood estimation, and presented Wiberg algorithms for L_2 bundle adjustment, L_2 projective bundle adjustment, and maximum likelihood Poisson matrix factorization.

A separate paper describes L_1 general Wiberg, which is robust to outliers in the observations. However, L_1 Wiberg solves a linear program to find each step while L_2 Wiberg solves a simple linear system, so L_2 is much faster. Minimizing the Huber norm, which is continuous but approximates the L_1 norm, might produce L_1's robustness to outliers and L_2's speed. So, our next task is to determine whether the Huber norm can be minimized using our Wiberg approach.

References

1. Wiberg, T.: Computation of principal components when data are missing. In: Second Symposium of Computation Statistics, Berlin, pp. 229–326 (1976)
2. Okatani, T., Deguchi, K.: On the Wiberg algorithm for matrix factorization in the presence of missing components. International Journal of Computer Vision 72(3) (May 2007)
3. Okatani, T., Yoshida, T., Deguchi, K.: Efficient algorithm for low-rank matrix factorization with missing components and performance comparison of latest algorithms. In: International Conference on Computer Vision, Barcelona, Spain (November 2011)
4. Eriksson, A., van den Hengel, A.: Efficient computation of robust low-rank matrix approximations in the presence of missing data using the L_1 norm. In: Computer Vision and Pattern Recognition, San Francisco, CA (June 2010)
5. Strelow, D.: General and nested wiberg minimization. In: Computer Vision and Pattern Recognition, Providence, RI (June 2012)

6. Ke, Q., Kanade, T.: Robust L_1 norm factorization in the presence of outliers and missing data by alternative convex programming. In: Computer Vision and Pattern Recognition, San Diego, CA (June 2005)
7. Ruhe, A., Wedin, P.: Algorithms for separable nonlinear least squares problems. SIAM Review 22(3), 318–337 (1980)
8. Richards, F.: A method of maximum-likelihood estimation. Journal of the Royal Statistical Society, Series B (Methodological) 23(2), 469–475 (1961)
9. Poelman, C.: The paraperspective and projective factorization methods for recovering shape and motion. PhD thesis, Carnegie Mellon University (1995)
10. Fackler, P.L.: Notes on matrix calculus (September 2005), `http://www4.ncsu.edu/~pfackler/MatCalc.pdf` (accessed August 29, 2011)
11. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes: The Art of Scientific Computing, 3rd edn. Cambridge University Press, New York (2007)
12. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle Adjustment – A Modern Synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) Vision Algorithms 1999. LNCS, vol. 1883, pp. 298–372. Springer, Heidelberg (2000)
13. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791 (1999)
14. Forsyth, D.A., Ponce, J.: Computer vision: A modern approach. Prentice-Hall, Upper Saddle River (2003)
15. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge (2003)
16. Oliensis, J., Hartley, R.: Iterative extensions of the Sturm/Triggs algorithm: convergence and nonconvergence. IEEE PAMI 29(12), 2217–2233 (2007)
17. Civera, J., Davison, A.J., Montiel, J.M.M.: Struture from motion using the extended Kalman filter. Springer, Berlin (2012)