Robust Bundle Adjustment Revisited

Christopher Zach

Toshiba Research Europe, Cambridge, UK*

Abstract. In this work we address robust estimation in the bundle adjustment procedure. Typically, bundle adjustment is not solved via a generic optimization algorithm, but usually cast as a nonlinear least-squares problem instance. In order to handle gross outliers in bundle adjustment the least-squares formulation must be robustified. We investigate several approaches to make least-squares objectives robust while retaining the least-squares nature to use existing efficient solvers. In particular, we highlight a method based on *lifting* a robust cost function into a higher dimensional representation, and show how the lifted formulation is efficiently implemented in a Gauss-Newton framework. In our experiments the proposed lifting-based approach almost always yields the best (i.e. lowest) objectives.

Keywords: Bundle adjustment, nonlinear least-squares optimization, robust cost function.

1 Introduction

Large scale nonlinear least-squares optimization occurs frequently in geometric computer vision and robotics to refine a set of continuous unknowns given all the observations extracted from acquired (image) data. Least-squares estimation in general has an underlying Gaussian noise (or residual) assumption if viewed as inference in a probabilistic model. This Gaussian assumption is typically violated whenever large residuals are observed, and consequently least-squares formulations are robustified to cope with such large residuals. Nevertheless, least-squares methods are popular even in the robustified setting because of their simplicity, efficiency, and the general availability of respective implementations. Thus, a robustified problem has to be cast as a nonlinear least-squares instance in order to make use of existing software and algorithms. Recent improvements for large-scale nonlinear estimation in geometric computer vision focus on the aspect of efficiently solving least-squares optimization, but usually do not explicitly address robustness of the formulation. In this work we focus on different options to cast a robustified objective into a nonlinear least-squares one. Thus, our contributions include:

- we review a number of approaches to make nonlinear least squares robust (Section 3)

* Much of this work was done while the author was with MSR Cambridge.

D. Fleet et al. (Eds.): ECCV 2014, Part V, LNCS 8693, pp. 772–787, 2014.

- we provide an in-depth discussion of lifting schemes (Section 3.4),
- we show how to make the lifting approach computationally attractive (Section 4),
- and experimentally compare the discussed approaches on large scale bundle adjustment instances (Section 5).

In our experiments the lifting method shows generally a very promising performance by reaching a much better local minimum in most test cases.

The main application of nonlinear least-squares in 3D computer vision is in the bundle adjustment routine, which refines the camera parameters (such as their pose, focal lengths and distortion coefficients) and the 3D point positions. We refer to [11] for a general treatment of geometric computer vision and to [22] for an in-depth discussion of bundle adjustment. Since the number of cameras of a typical dataset is in the hundreds or thousands (with about 10 scalar unknowns each) and the number of 3D points is in the millions, a modern bundle adjustment implementation must be able to cope with such large-scale problem instances.

The current work horse for bundle adjustment is the Levenberg-Marquardt algorithm [16,19], which at its core solves a sequence of linear systems of *normal* equations with a (strictly) positive definite and generally very sparse system matrix. The standard tool to solve such linear systems are sparse Cholesky factorization [8] in combination with a column reordering scheme to reduce the fill-in. More recently, iterative conjugate-gradient based solvers to address the normal equations have been explored to speed up bundle adjustment (e.g. [4,6,15,14]). To our knowledge none of these recent works on bundle adjustment provide guidelines on how to incorporate a robust cost function into a least-squares objective such that existing solvers can be reused. The standard method to utilize robust costs into a nonlinear least-squares framework is iteratively reweighted least squares (IRLS), i.e. by reweighting the terms in the objective according to the current residual values. Two notable exceptions are [22] and [9], which are discussed in more detail in Section 3. Although the Levenberg-Marquardt method is by far the most popular to address nonlinear least-squares problems, related trust-region methods may be more efficient for bundle adjustment tasks [18]. It is also possible to reduce the problem size in bundle adjustment significantly by algebraically eliminating some unknowns (often the 3D points) and optimizing essentially over multi-view relations directly [23,20,12]. We stay with the classical projection-based formulation (optimizing over camera parameters and 3D points) for bundle adjustment, since we feel that a robustified noise model is most appropriate in that setting.

2 Background

In this section we introduce some notation and terminology, and provide a brief description of nonlinear least-squares objectives playing a central role in this work.

2.1 Notations

We address minimization problems of the form

$$\Theta^* = \arg\min_{\Theta \in \mathbb{R}^n} \sum_{k=1}^{M} \psi\big(r_k(\Theta)\big), \tag{1}$$

where $r_k(\Theta) : \mathbb{R}^n \to \mathbb{R}^d$ is the k-th residual function dependent on the unknown vector Θ. Therefore d is the dimension of the residuals, which is usually 2 in a standard bundle adjustment instance. All of the residual functions are assumed to be differentiable.

Without loss of generality all the (differentiable) penalizing functions ψ are the same for the residual terms. Typical choices for ψ have the properties that $\psi(r) \geq 0$, $\psi(0) = 0$, and ψ is monotonically increasing with respect to the norm of its argument (the residual vector). Sensible choices for ψ feature a sub-linear growth of its function value in order to make the objective robust to outlier residuals. Therefore we call ψ a robust kernel in this work. Nevertheless, we assume that small residuals are penalized in the least-squares sense, i.e. the Hessian of ψ is the identity matrix, which corresponds to a Gaussian noise assumption in the underlying probabilistic model. This explains in particular the $1/2$ scaling factor in many of the objectives stated below, and also the choice of scaling for robust kernels ψ in Section 3.4.

While $\psi : \mathbb{R}^d \to \mathbb{R}_0^+$ maps residual vectors to non-negative costs, in Section 3.2 we will use functions $\rho : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ mapping squared norms of residual vectors to costs. A choice of ρ induces ψ via $\psi(r) = \rho(\|r\|_2^2)$. Note that ρ penalizes residuals isotropically ignoring the direction of r_k. In the following we drop the explicit subscript to denote the Euclidean norm, and every occurrence of a norm $\|\cdot\|$ should be always read as $\|\cdot\|_2$.

Further, we stack the residual vector r_k and the respective Jacobians $J_k \overset{\text{def}}{=} \nabla_\Theta r_k$ into a large vector and matrix, respectively,

$$\mathbf{r}(\Theta) \overset{\text{def}}{=} \begin{pmatrix} r_1(\Theta) \\ \vdots \\ r_M(\Theta) \end{pmatrix} \qquad \mathbf{J}(\Theta) \overset{\text{def}}{=} \begin{pmatrix} J_1(\Theta) \\ \vdots \\ J_M(\Theta) \end{pmatrix}. \tag{2}$$

In general, bold-face letters indicate quantities spanning the whole objective not only individual residual terms.

Finally, for a positive semi-definite (p.s.d.) matrix A we denote the respective square root matrix either by $\sqrt{A}$ or $A^{1/2}$, $A = (\sqrt{A})^2 = (A^{1/2})^2$.

2.2 Nonlinear Least Squares Optimization

Nonlinear least squares optimization aims to find a minimizer Θ^* of a least-squares objective,

$$\Theta^* = \arg\min_{\Theta} \frac{1}{2} \sum_k \|r_k(\Theta)\|^2 = \arg\min_{\Theta} \frac{1}{2} \|\mathbf{r}(\Theta)\|^2. \tag{3}$$

It is beneficial to exploit the outer least-squares structure instead of solely relying on general minimization methods such as gradient descent, conjugate gradients, or quasi-Newton methods. Most popular methods to solve Eq. 3 are based on first order expansion of the residual r_k,

$$r_k(\Theta + \Delta\Theta) \approx r_k(\Theta) + J_k(\Theta)\Delta\Theta, \tag{4}$$

where $J_k(\Theta)$ is the Jacobian of r_k with respect to the parameters Θ (and evaluated at the current linearization point Θ). In the following we implicitly assume the dependence of r_k and J_k on Θ and consequently drop the respective argument Θ. Plugging the first order expansion into Eq. 3 and rearranging terms yields the normal equations of the Gauss-Newton method,

$$\mathbf{J}^T\mathbf{J}\Delta\Theta = -\mathbf{J}^T\mathbf{r}. \tag{5}$$

While it may be numerically advisible to solve the overconstraint linear equation $\mathbf{J}\Delta\Theta = -\mathbf{r}$ directly e.g. via a QR-decomposition [22], to our knowledge all methods to solve large scale instances of Eq. 3 are based on the Gauss-Newton method and the normal equations. In particular, the Levenberg-Marquardt method using augmented normal equations,

$$\left(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}\right)\Delta\Theta = -\mathbf{J}^T\mathbf{r}, \tag{6}$$

with $\mu > 0$, became very popular over the last years. The key requirement for Gauss-Newton-type methods to be competitive solvers is that $\mathbf{J}^T\mathbf{J}$ is a sparse matrix, which can be exploited to solve the (augmented) normal equations efficiently. In terms of the original problem formulation (Eq. 3) the sparsity of $\mathbf{J}^T\mathbf{J}$ means, that each residual r_k depends only on a small subset of parameters in Θ.

Most available implementations for large scale nonlinear least squares problems have an interface, that allows the user to specify the residual vector $\mathbf{r}$ and the Jacobian $\mathbf{J}$ (as sparse matrix). In some implementations a weight or even a covariance matrix may be provided with each residual. Two of the methods described in Section 3 require a minimal extension to such an interface: in the computation of $\mathbf{J}^T\mathbf{J}$ and $\mathbf{J}^T\mathbf{r}$ we allow block diagonal matrices to be inserted, i.e. we facilitate the efficient computation of

$$\mathbf{J}^T\mathbf{D}\mathbf{J} \qquad \text{and} \qquad \mathbf{J}^T\bar{\mathbf{D}}\mathbf{r} \tag{7}$$

for block diagonal matrices $\mathbf{D}$ and $\bar{\mathbf{D}}$ with $d \times d$ non-zeros blocks along the diagonal. Note that $\mathbf{J}^T\mathbf{J}$ and $\mathbf{J}^T\mathbf{D}\mathbf{J}$ will have the same sparsity pattern.

3 Robustified Nonlinear Least-Squares

The objective given in Eq. 1 has, at a first glance, little in common with nonlinear least-squares instances in Eq. 3 (other than that the objective is composed of individual terms). Often $\psi(\cdot)$ in Eq. 1 behaves like a quadratic function for small arguments, i.e. for residual vectors r close to 0 we have that $\psi(r) \approx \|r\|^2/2$. This

usually corresponds to a Gaussian assumption on the noise model for inliers, which are characterized by having non-heavy tail residuals. In most estimation tasks the majority of observations are inliers, hence the majority of terms in Eq. 1 are (close to) quadratic functions near the minimizer Θ^*. Consequently, it appears beneficial to cast the problem Eq. 1 as nonlinear least-squares instance (Eq. 3) in order to leverage available efficient solvers for the latter problem class. Below we discuss several options to introduce robustness into least-squares estimation.

3.1 Iteratively Reweighted Least Squares

We assume that ψ is isotropic, i.e. $\psi(r) = \phi(\|r\|)$ for a function $\phi : \mathbb{R}_0^+ \to \mathbb{R}_0^+$. Then the first-optimality condition for Θ^* in Eq. 1 are

$$0 = \sum_k \phi'\big(\big\|r_k(\Theta^*)\big\|\big) \frac{r_k^T(\Theta^*)}{\big\|r_k(\Theta^*)\big\|} \frac{\partial r_k}{\partial \Theta}(\Theta^*) = \sum_k \underbrace{\frac{\phi'\big(\big\|r_k(\Theta^*)\big\|\big)}{\big\|r_k^T(\Theta^*)\big\|}}_{\stackrel{\text{def}}{=} \omega\left(r_k(\Theta^*)\right)} r_k^T(\Theta^*) J_k(\Theta^*)$$

$$= \sum_k \omega\big(r_k(\Theta^*)\big) r_k^T(\Theta^*) J_k(\Theta^*).$$

If the (scalar) values $\omega_k^* \stackrel{\text{def}}{=} \omega\big(r_k(\Theta^*)\big)$ are known for a minimizer Θ^* (and therefore constant), the optimality conditions above are the ones for a standard nonlinear least-squares problem,

$$\Theta^* = \arg\min_{\Theta} \frac{1}{2} \sum_k \omega_k^* \big\|r_k(\Theta)\big\|^2. \tag{8}$$

Since the weights ω_k^* are usually not known, one can employ an estimate based on the current solution maintained by the optimization algorithm. This also implies that the objective to minimize varies with each update of the current solution. Further, algorithms such as the Levenberg-Marquardt method have a built-in "back-tracking" step, which discards updates leading to an inferior objective value. Depending on the programming interface this back-tracking stage may only see the surrogate objective in Eq. 8 or the true cost in Eq. 1.

3.2 Triggs Correction

For completeness we review the approach outlined in [22] (and termed "Triggs correction" in [3]) to introduce robustness into a (nonlinear) least-squares method. Let $\rho : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ be a robust kernel mapping the squared residual norm to a real number i.e. the objective is

$$\min_{\theta} \frac{1}{2} \sum_k \rho\left(\|r_k(\theta)\|^2\right). \tag{9}$$

A first-order expansion of $r_k(\Theta + \Delta\Theta) \approx r_k + J_k\Delta\Theta$ (dropping the explicit dependence of r_k and J_k on the current linearization point Θ) together with a second-order expansion of

$$h_k(\Delta\Theta) \stackrel{\text{def}}{=} \frac{1}{2}\rho\left(\|r_k + J_k\Delta\Theta\|^2\right) \tag{10}$$

yields

$$h_k(\Delta\Theta) \approx \rho' r_k^T J_k \Delta\Theta + \frac{1}{2}(\Delta\Theta)^T J_k^T \underbrace{\left(\rho' I + 2\rho'' r_k r_k^T\right)}_{\stackrel{\text{def}}{=} H_k} J\Delta\Theta + const, \tag{11}$$

where ρ' and ρ'' are evaluated at r_k. This expression is rewritten in [22] as a nonlinear least-squares instance by replacing J with $\tilde{J} = \sqrt{H_k}J$ (given that H_k is strictly p.d.[1]). In theory, any existing algorithm for nonlinear least squares minimization can be used, provided that the implementation allows sufficient powerful reweighting of the residuals and the Jacobian.[2]

Observe that one eigenvector of H_k is r_k with corresponding eigenvalue $\rho' + 2\rho''\|r_k\|^2$, and all other eigenvalues are ρ' (which is non-negative since a sensible function ρ is monotonically increasing) with eigenvectors $v \perp r_k$. Consequently, H_k is p.s.d. iff $\rho' + 2\rho''\|r_k\|^2$ is non-negative. If H_k is indefinite, one approach is to drop the ρ'' term entirely (i.e. set ρ'' to 0, as e.g. done in the Ceres Solver [3]). This amounts to representing the mapping $\Delta\Theta \mapsto \frac{1}{2}\rho(\|r(\Theta + \Delta\Theta)\|^2)$ locally at Θ via the (strictly) convex surrogate function

$$\Delta\Theta \mapsto \rho' r_k J_k \Delta\Theta + \frac{\rho'}{2}(\Delta\Theta)^T J_k^T J_k \Delta\Theta + const. \tag{12}$$

One could replace an indefinite H_k by its closest p.s.d. approximation $\tilde{H}_k$, which (using the Frobenius norm as distance between matrices) is determined by

$$\tilde{H}_k = \begin{cases} H_k & \text{if } H_k \succeq 0 \\ \rho'\left(I - \frac{r_k r_k^T}{\|r_k\|^2}\right) & \text{otherwise.} \end{cases} \tag{13}$$

Consequently, the respective normal equations to solve in each iteration read as

$$\sum_k J_k^T \tilde{H}_k J_k \Delta\Theta = -\sum_k \rho' J_k^T r_k. \tag{14}$$

In practice this approach converged to higher objectives, and therefore we use the same strategy as Ceres for indefinite H_k.

[1] One way to model $\sqrt{H_k}$ is using the ansatz $\sqrt{H_k} = \sqrt{\rho'}(I + \alpha r_k^0 (r_k^0)^T)$ for an α [22].
[2] The Jacobian and the residuals need to be reweighted differently.

3.3 Square-Rooting the Kernel

Another option proposed in [9] is to rewrite $\psi(r)$ as $\left(\sqrt{\psi(r)})\right)^2$, which implies that $\sqrt{\psi(r)}$ is now employed as the residual in the outer least-squares formulation. In some way this is a complete opposite approach to the "Triggs correction" above, since the robust kernel is pulled into the inner scope of the minimization problem. Note that by naive "square rooting" the robust kernel one effectively replaces a d-dimensional residual r with a one-dimensional one, $\sqrt{\psi(r)}$, which loses structural information. By noting that for any d-dimensional vector u with unit norm we have

$$\psi(r) = \left\langle \sqrt{\psi(r)}u,\, \sqrt{\psi(r)}u \right\rangle = \left\| \sqrt{\psi(r)}u \right\|^2, \tag{15}$$

we can use the specific choice of $u = r/\|r\|$ to obtain $\psi(r) = \left\| \sqrt{\psi(r)}/\|r\| \cdot r \right\|^2$, which means that the original residual r is weighted by a factor of $\sqrt{\psi(r)}/\|r\|$, which is a different one than used in IRLS (Section 3.1), since—first of all—this weight is explicitly dependent on Θ. Further, for convenience we state the derivative of this modified residual with respect to the unknowns Θ,

$$\frac{d}{d\Theta}\left(\frac{\sqrt{\psi(r)}}{\|r\|}r\right) = \left(\frac{1}{2\|r\|\sqrt{\psi(r)}}r\frac{d\psi(r)}{dr} + \frac{\sqrt{\psi(r)}}{\|r\|^3}\left(\|r\|^2 I - rr^T\right)\right)\frac{dr}{d\Theta}. \tag{16}$$

3.4 Lifting the Kernel

Another way to represent a robust cost function within a (nonlinear) least-squares framework is by introducing extra variables playing the role of "confidence weights" for each residual. Converting a minimization problem to a higher-dimensional one by augmenting the set of unknowns is often termed "lifting" in the optimization literature. Sometimes lifting allows a global optimal solution for an otherwise difficult minimization problem (e.g. [13,7]). Our motivation for enriching the set of unknowns is different: we hope to circumnavigate the flat regions in robust kernels by indirectly representing the robustness and therefore have better chances to reach a good local minimum. Initial evidence for such an improved behavior of lifted costs in a synthetic line-fitting experiment is given in [24]. Rewriting non-convex robust costs via lifting has a long history: "half-quadratic" optimization introduces additional variables to allow efficient minimization by using a block-coordinate method in the context of low-level vision tasks [10,5]). More recently, "switching constraints" (analogous to confidence weights) are employed to make pose graph optimization robust [21,1] (where the robust kernel is identified as the Geman-McClure one). These methods use an IRLS approach to incorporate robustness into a non-linear least-squares solver.

In our setting, one instance of a lifted kernel for robust objectives is

$$\min_{\Theta} \sum_k \frac{1}{2}\psi\big(\,\|r_k(\Theta)\|\,\big) = \min_{\Theta,\mathbf{w}} \sum_k \underbrace{\frac{1}{2}\left(w_k^2\,\|r_k(\Theta)\|^2 + \kappa^2(w_k^2)\right)}_{\stackrel{\text{def}}{=}\hat{\psi}(r_k(\Theta),w_k)}, \tag{17}$$

where w_k is the confidence weight for the k-th residual. We denote the lifted kernel of $\psi(\cdot)$ as $\hat{\psi}(\cdot,\cdot)$. Depending on the choice of κ one arrives at different kernels ψ, and in the following we briefly discuss a few sensible choices for κ.

L^1-cost: The choice of $\hat{\psi}(r,w) = \frac{1}{2}\left(w^2 r^2 + \frac{1}{w^2}\right)$, i.e. $\kappa^2(w^2) = 1/w^2$, results in $\psi(r) = \min_w \hat{\psi}(r,w) = \|r\|/2$.

Tukey's biweight: One can lift the robust biweight kernel

$$\psi(r) = \begin{cases} \frac{\tau^2}{6}\left(1 - \left(1 - \frac{\|r\|^2}{\tau^2}\right)^3\right) \\ \frac{\tau^2}{6} \end{cases} = \begin{cases} \frac{\|r\|^2}{2}\left(1 - \frac{\|r\|^2}{\tau^2} + \frac{\|r\|^4}{3\tau^4}\right) & \text{if } \|r\|^2 \leq \tau^2 \\ \frac{\tau^2}{6} & \text{otherwise} \end{cases} \tag{18}$$

by setting

$$\hat{\psi}(r,w) = \frac{1}{2}w^2\|r\|^2 + \frac{1}{6}(|w|-1)^2(2|w|+1). \tag{19}$$

Note that the regularizer of w, $(|w|-1)^2(2|w|+1)/6$, is a double-well potential with minima at $w = \pm 1$ (inlier case), finite cost at $w = 0$ (outlier), and unbounded cost as $|w| \to \infty$ (see also Fig. 2). Formally we have in this setting

$$\kappa^2(w^2) = \frac{1}{6}\left(\sqrt{w^2} - 1\right)^2\left(2\sqrt{w^2} + 1\right).$$

Smooth truncated quadratic: The kernel of our main interest was proposed in [17] (but without establishing the strong connection between the introduced weights and robust estimation) and is given by

$$\kappa^2(w^2) = \frac{\tau^2}{2}\left(w^2 - 1\right)^2 \qquad \text{or} \qquad \kappa(w^2) = \frac{\tau}{\sqrt{2}}\left(w^2 - 1\right). \tag{20}$$

Again the regularizer $\kappa^2(w^2)$ is a double-well potential with zero cost at $w = \pm 1$, finite cost at $w = 0$ (outlier case), and unbounded cost for $|w| \to \infty$ (see Fig. 2). It can be easily shown that

$$\psi(r) = \frac{1}{2}\min_w \left\{ w^2\|r\|^2 + \frac{\tau^2}{2}\left(w^2 - 1\right)^2 \right\} = \begin{cases} \frac{1}{2}\|r\|^2\left(1 - \frac{\|r\|^2}{2\tau^2}\right) & \text{if } \|r\|^2 \leq \tau^2 \\ \tau^2/4 & \text{otherwise,} \end{cases} \tag{21}$$

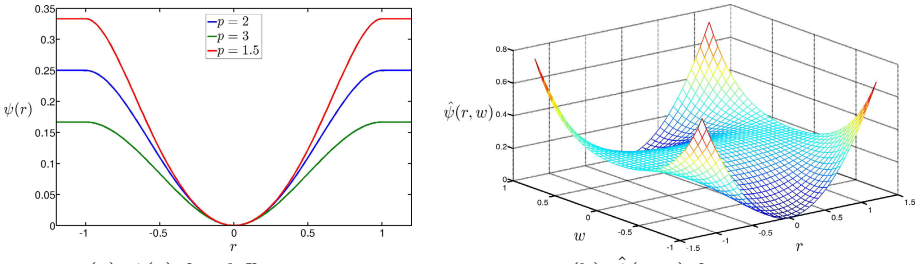
which is a particular smooth approximation to a truncated quadratic kernel. We can create a family of such smooth approximations by making κ dependent on a parameter $p > 1$, e.g.

$$\kappa^2(w^2) = \frac{\tau^2}{p}\left(w^2 - 1\right)^p, \tag{22}$$

which corresponds to the robust kernel

$$\psi(r) = \begin{cases} \frac{1}{2}\|r\|^2 \left(1 - \frac{p-1}{p}\left(\frac{\|r\|^2}{\tau^2}\right)^{\frac{1}{p-1}}\right) & \text{if } \|r\|^2 \leq \tau^2 \\ \frac{\tau^2}{2p} & \text{otherwise,} \end{cases} \tag{23}$$

In the limit $p \to 1$ this kernel ψ approaches the truncated quadratic cost (see also Fig. 1(a)). Fig. 1(b) illustrates the lifted kernel $\hat{\psi}$, which has almost nowhere a zero gradient (in contrast to the one-dimensional function ψ, which has a zero gradient whenever the argument is larger than τ). Our hypothesis therefore is, that it is exactly this feature that enables lifted representations to have a better chance of escaping poor local minima than direct robust kernels.



(a) $\psi(r)$ for different p (b) $\hat{\psi}(r, w)$ for $p = 2$

Fig. 1. Different robust kernels ψ for different values of p in Eq. 23 (left), and the visualization of the lifted kernel $\hat{\psi}$ for $p = 2$. Note that the lifted kernel appears "relatively convex" for large residuals.

From weight functions to lifted representations: As in Section 3.1 we now assume $\psi(r) = \phi(\|r\|)$ for a function $\phi : \mathbb{R}_0^+ \to \mathbb{R}_0^+$. We will use $e = \|r\|$ in the following. The *weight function* ω for the kernel ϕ is given by $\omega(e) \stackrel{\text{def}}{=} \phi'(e)/e$. This weight function plays a crucial role in iteratively reweighted least-squares. In order to lift ϕ (and consequently ψ), we have to find a function κ^2, such that

$$\phi(e) = \min_w \hat{\phi}(e, w) = \min_w \frac{1}{2}\left(w^2 e^2 + \kappa^2(w^2)\right).$$

First order optimality conditions imply $w = 0$ or $e^2 + (\kappa^2)'(w^2) = 0$. Since in a reweighted approach, $\omega(e)$ plays the role of w^2 in the lifted formulation, we use the ansatz $w^2 = \omega(e)$, leading to $e = \omega^{-1}(w^2)$ or $e^2 - \left(\omega^{-1}(w^2)\right)^2 = 0$. Comparing this with the first-order optimality condition we read that

$$(\kappa^2)'(w^2) = -\left(\omega^{-1}(w^2)\right)^2,$$

where ω^{-1} is the inverse function of ω (if it exists). For instance, the Cauchy kernel, $\phi(e) = \frac{\tau^2}{2}\log\left(1 + e^2/\tau^2\right)$, has a weight function $\omega(e) = 1/(1 + e^2/\tau^2)$ and a lifted representation

$$\hat{\phi}(e, w) = \frac{1}{2} w^2 e^2 + \frac{\tau^2}{4} \left(w^4 - 2 \log(w^2) - 1 \right). \tag{24}$$

Similarly, lifted formulations for all robust kernels with strictly monotonically decreasing weight function ω can be derived. This construction is related but not equivalent to the one in [5].

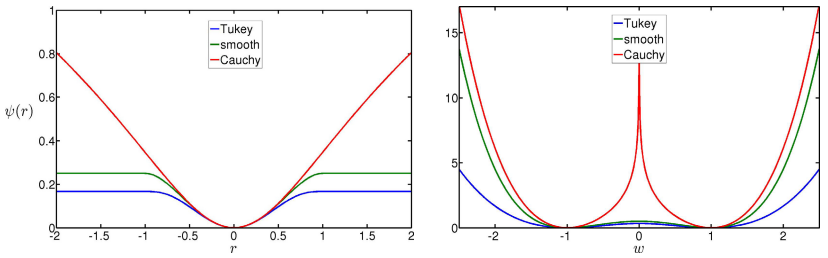


Fig. 2. Tukey's biweight, the smooth truncated quadratic (Eq. 21), and the Cauchy kernels (left). The associated regularization $w \mapsto \kappa^2(w^2)$ in the lifted representation (right).

Note that for all the choices of $\hat{\psi}$ above we can rewrite $\hat{\psi}(r, w)$ as

$$\hat{\psi}(r, w) = \frac{1}{2} \left(w^2 \|r\|^2 + \kappa^2(w^2) \right) = \frac{1}{2} \left\| \begin{matrix} wr \\ \kappa(w^2) \end{matrix} \right\|^2, \tag{25}$$

i.e. $\hat{\psi}$ can be written as a nonlinear least-squares term over a lifted residual $\hat{r} \stackrel{\text{def}}{=} \begin{pmatrix} wr \\ \kappa(w^2) \end{pmatrix}$, and therefore lifted kernels can be immediately incorporated into a standard nonlinear least squares solver. This is probably not advisible, since having a potentially huge number of extra unknowns puts extra burden on the column reordering step and on the employed matrix factorization method. Consequently, we describe a more efficient implementation of lifted approaches in the next section.

4 Efficient Implementation of Lifted Kernels

Lifting the robust kernel requires maintaining one extra variable per residual. While we believe that the cost of storing an extra confidence weight is negligible, the computational burden in particular in Gauss-Newton-type solvers seems to increase significantly using a lifted representation. In this section we demonstrate that the computational complexity is essentially the same for lifted and non-lifted representations. Since a confidence weight w_k is linked to exactly one residual

r_k, the sparsity structure of the lifted Jacobian stays the same. If we consider the lifted residual as in Eq. 25, then we have for the lifted full Jacobian $\hat{\mathbf{J}}$,

$$\hat{\mathbf{J}} = \begin{pmatrix} \mathbf{WJ} & \mathbf{R} \\ \mathbf{0} & \mathbf{K}' \end{pmatrix} \qquad \hat{\mathbf{r}} = \begin{pmatrix} \mathbf{Wr} \\ \mathbf{k} \end{pmatrix} \tag{26}$$

where

- $\mathbf{W} \stackrel{\text{def}}{=} \operatorname{diag}(w_1, \dots, w_M) \otimes I_{d\times d}$,
- $\mathbf{k} \stackrel{\text{def}}{=} \big(\kappa(w_1), \dots, \kappa(w_M)\big)^T$,
- $\mathbf{K}' \stackrel{\text{def}}{=} 2 \operatorname{diag}\big(w_1\kappa'(w_1^2), \dots, w_M\kappa'(w_M^2)\big)$, and
- $\mathbf{R}$ is a rectangular block matrix $\mathbf{R} \stackrel{\text{def}}{=} \operatorname{diag}(r_1, \dots, r_M)$.

Thus, $\hat{\mathbf{J}}^T\hat{\mathbf{J}}$ and $\hat{\mathbf{J}}^T\mathbf{r}$ are given by

$$\hat{\mathbf{J}}^T\hat{\mathbf{J}} = \begin{pmatrix} \mathbf{J}^T\mathbf{W}^2\mathbf{J} & \mathbf{J}^T\mathbf{WR} \\ \mathbf{R}^T\mathbf{WJ} & \mathbf{R}^T\mathbf{R} + \mathbf{K}'^2 \end{pmatrix} \qquad \hat{\mathbf{J}}^T\hat{\mathbf{r}} = \begin{pmatrix} \mathbf{J}^T\mathbf{W}^2\mathbf{r} \\ \mathbf{R}^T\mathbf{Wr} + \mathbf{K}'\mathbf{k} \end{pmatrix}. \tag{27}$$

Note that $\mathbf{R}^T\mathbf{R} = \operatorname{diag}\big(\|r_1\|^2, \dots, \|r_M\|^2\big)$, and we have the vector

$$\mathbf{R}^T\mathbf{Wr} + \mathbf{K}'\mathbf{k} = \big(w_k \left(\|r_k\|^2 + 2\kappa'(w_k^2)\kappa(w_k^2)\right)\big)_{k=1,\dots,M}.$$

Since the confidence weights are only appearing in exactly one residual term, one can eliminate all w_k in parallel from the (augmented) normal equations

$$\left(\hat{\mathbf{J}}^T\hat{\mathbf{J}} + \mu\mathbf{I}\right)\begin{pmatrix} \Delta\Theta \\ \Delta\mathbf{w} \end{pmatrix} = -\hat{\mathbf{J}}^T\hat{\mathbf{r}}, \tag{28}$$

via the Schur complement, which leads to the reduced system

$$\left(\mathbf{J}^T\mathbf{W}(\mathbf{I} - \mathbf{DRR}^T)\mathbf{WJ} + \mu\mathbf{I}\right)\Delta\Theta = \text{r.h.s.} \tag{29}$$

where $\mathbf{D} = (\mathbf{R}^T\mathbf{R} + \mathbf{K}'^2 + \mu\mathbf{I})^{-1}$ is a diagonal matrix with

$$\mathbf{D}_{kk} = \frac{1}{\|r_k\|^2 + (2w_k\kappa'(w_k^2))^2 + \mu}, \tag{30}$$

and $\mathbf{RR}^T$ is a block diagonal matrix with $d \times d$ blocks $(\mathbf{RR}^T)_{kk} = r_k r_k^T$. Note that $\mathbf{I} - \mathbf{DRR}^T \succ 0$ for $\mu > 0$. Further, the (block) non-zero structure of $\mathbf{J}^T\mathbf{W}(\mathbf{I}-\mathbf{DRR}^T)\mathbf{WJ}$ is the same as for $\mathbf{J}^T\mathbf{J}$, since $\mathbf{W}(\mathbf{I}-\mathbf{DRR}^T)\mathbf{W}$ is a block diagonal matrix with $d \times d$ blocks along the diagonal. One can easily calculate a square root of $\mathbf{I} - \mathbf{DRR}^T$ via the ansatz $(\mathbf{I} - \mathbf{DRR}^T)^{1/2} = \mathbf{I} - \tilde{\mathbf{D}}\mathbf{RR}^T$ with $\tilde{\mathbf{D}}$ being a diagonal matrix,

$$\tilde{\mathbf{D}}_{kk} = \frac{1}{\|r_k\|^2}\left(1 - \sqrt{\frac{(2w_k\kappa'(w_k^2))^2 + \mu}{\|r_k\|^2 + (2w_k\kappa'(w_k^2))^2 + \mu}}\right). \tag{31}$$

Consequently, the modified Jacobian can be easily computed by left-multiplication with a block-diagonal matrix, i.e.

$$\mathbf{J} \rightsquigarrow \left(\mathbf{I} - \tilde{\mathbf{D}}\mathbf{R}\mathbf{R}^T\right)\mathbf{W}\mathbf{J}, \tag{32}$$

which is provided to the solver. The right hand side in Eq. 29 is given by

$$\text{r.h.s.} = \mathbf{J}^T\mathbf{W}\left(\mathbf{R}\mathbf{D}\left(\mathbf{R}^T\mathbf{W}\mathbf{r} + \mathbf{K}'\mathbf{k}\right) - \mathbf{W}\mathbf{r}\right) = -\mathbf{J}^T\mathbf{W}\tilde{\mathbf{r}}, \tag{33}$$

which corresponds to using a reweighted and adjusted residual vector,

$$\tilde{r}_k \stackrel{\text{def}}{=} w_k\left(1 - \frac{\|r_k\|^2 + 2\kappa'(w_k^2)\kappa(w_k^2)}{\|r_k\|^2 + (2w_k\kappa'(w_k^2))^2 + \mu}\right)r_k \tag{34}$$

together with a reweighted Jacobian $\mathbf{W}\mathbf{J}$. Backsubstitution to determine Δw_k leads to

$$\Delta w_k = -w_k\frac{r_k^T\left(r_k + J_k\Delta\Theta\right) + 2\kappa'(w_k^2)\kappa(w_k^2)}{\|r_k\|^2 + (2w_k\kappa'(w_k))^2 + \mu}. \tag{35}$$

With the minimal extension to a standard Levenberg-Marquardt code as outlined in Section 2.2, it is straightforward to implement this Schur-complement approach by solving

$$\left(\mathbf{J}^T\mathbf{B}\mathbf{J} + \mu\mathbf{I}\right)\Delta\Theta = -\mathbf{J}^T\bar{\mathbf{B}}\mathbf{r}$$

in each iteration, where $\mathbf{B}$ and $\bar{\mathbf{B}}$ are appropriate block diagonal matrices.

5 Numerical Results

We use a freely available sparse Levenberg-Marquardt C++ implementation[3] and extended its interface to allow insertion of block-diagonal matrices as indicated in Section 2.2. The test problems are the ones used in [4] based on structure-from-motion datasets generated from community photo collections [2]. We had to leave out the larger problem instances due to their memory consumption. The utilized objective in our bundle adjustment is

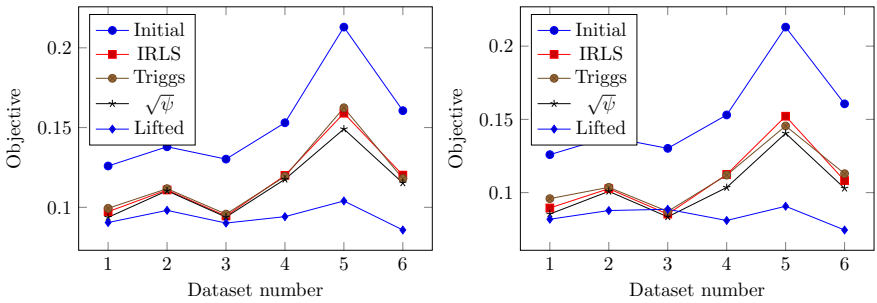
$$\sum \psi\big(f_i\eta_i\left(\pi(R_iX_j + t_i)\right) - \hat{p}_{ij}\big), \tag{36}$$

where $\hat{p}_{ij} \in \mathbb{R}^2$ is the observed image observation of the j-th 3D point in the i-image, $X_j \in \mathbb{R}^3$ is the j-th 3D point, $R_i \in SO(3)$ and $t_i \in \mathbb{R}^3$ are the orientation parameters of the i-th camera, $\pi : \mathbb{R}^3 \to \mathbb{R}^2$, $\pi(X) = X/X_3$ is the projection, and f_i is the respective focal length. η_i is the lens distortion function with $\eta_i(p) = (1 + k_{i,1}\|p\|^2 + k_{i,2}\|p\|^4)p$, and ψ is the robust kernel from Eq. 21 with $\tau = 1$ (i.e.

[3] We selected the "simple sparse bundle adjustment" software, `http://www.inf.ethz.ch/personal/chzach/oss/SSBA-3.0.zip`, mostly because of the simplicitly of the underlying API.

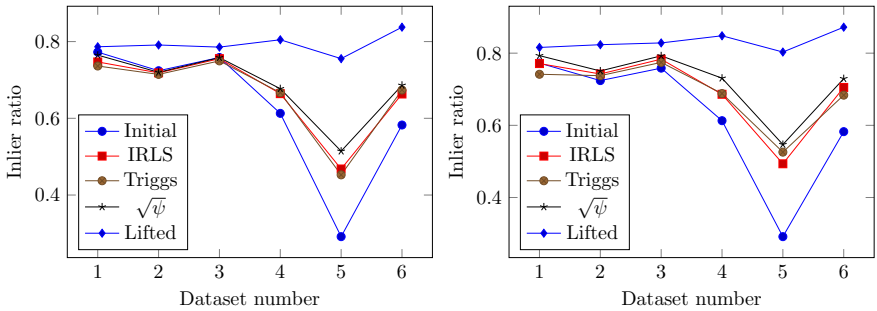
Table 1. The main characteristics of the used datasets

Number	Dataset	# cameras	# 3D points	# observations
1	Trafalgar	257	65132	225911
2	Dubrovnik	356	226730	1255268
3	Ladybug	598	69218	304170
4	Venice	427	310384	1699145
5	Final-93	93	61203	287451
6	Final-394	394	100368	534408



(a) Metric (b) Full

Fig. 3. Initial and final objectives (normalized with the observation count) reached by the different methods



(a) Metric (b) Full

Fig. 4. Initial and reached final ratios obtained by the different methods. The inlier ratio is an indicator of how many terms in the objective are in the flat outlier region.

inliers may have up to 1 pixel reprojection error). We run bundle adjustment in two modes: the "metric" mode optimizes only over X_j, R_i and t_i and keeps the focal lengths and lens distortion parameters fixed, and the "full" mode optimizes over all unknowns. We report both settings in order to determine whether the additional nonlinearity induced by the internal camera parameters has an impact on the results.

The runtime varies between the datasets and ranges from about 1 minute (Final-93 using metric refinement only) and 30 minutes (Final-394 with full adjustment) on a standard laptop computer. The computational complexity of all approaches is similar. The median slowdown (with respect to IRLS) of one LM iteration is 1.0 (IRLS), 1.0337 (Triggs), 1.6643 ($\sqrt{\psi}$), and 1.5814 (lifted). Please refer to the supplementary material for detailed runtimes.

The maximum number of iterations in the Levenberg-Marquardt algorithm is 100, and the iterations stop when the relative updates are less than 10^{-12} in magnitude. Table 1 summarizes the relevant figures for the tested datasets. The initial and obtained final objective values for the different approach to robustification (called "IRLS", "Triggs", $\sqrt{\psi}$, and "lifted" with the weights w_k initialized to one) are illustrated in Fig. 3. With the exception of the "Ladybug" dataset in full refinement mode, the lifted representation achieves the lowest objectives. The other methods cluster around similar (usually higher) final objective values with the square-root approach slightly ahead of the others.

Another interesting statistics is the inlier ratio (i.e. the fraction of residuals with at most $\tau = 1$ pixels reprojection error), which is depicted in Fig. 4. This number indicates how many residuals are "active", i.e. contribute to the overall gradient for our choice of robust kernel. Interestingly, the inlier ratio is always the highest for the lifted approach, which may be an advantage in bundle adjustment instances with large initial drift (such as often induced by loop closure). More experimental results can be found in the supplementary material.

6 Conclusions

In this work we addressed how robust cost functions can be incorporated into a nonlinear least-squares framework. We discussed several options how to combine robust costs with a Gauss-Newton-type solver, including iteratively reweighted least-squares, the Triggs correction, square rooting the robust kernel, and finally lifting the kernel. In terms of achieved objective the lifted approach outperforms the other ones by far in most tested problem instances. Consequently, we believe that lifting a kernel function is a promising method to incorporate robustness into an otherwise non-robust objective, especially since lifting comes with negligible extra computational cost,

One direction of future work is to make the square-root and the lifted approach accessible in the recently very popular Ceres solver (which currently implements robust cost functions via the Triggs correction). Source code based on a direct sparse Levenberg-Marquardt implementation for non-linear least squares is available at http://github.com/chzach/SSBA.

Ackowledgements. I am very grateful to Andrew Fitzgibbon for extensive discussions and to Andrea Cohen for contributing Figures 1 and 2. Further, I would like to thank Sameer Agarwal and the area chairs for pointing out additional literature.

References

1. Agarwal, P., Tipaldi, G.D., Spinello, L., Stachniss, C., Burgard, W.: Robust map optimization using dynamic covariance scaling. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 62–69. IEEE (2013)
2. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building rome in a day. Communications of the ACM 54(10), 105–112 (2011)
3. Agarwal, S., Mierle, K.: Others: Ceres solver, `https://code.google.com/p/ceres-solver/`
4. Agarwal, S., Snavely, N., Seitz, S.M., Szeliski, R.: Bundle adjustment in the large. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 29–42. Springer, Heidelberg (2010)
5. Black, M.J., Rangarajan, A.: On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. IJCV 19(1), 57–91 (1996)
6. Byröd, M., Åström, K.: Conjugate gradient bundle adjustment. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 114–127. Springer, Heidelberg (2010)
7. Chan, T.F., Esedoglu, S., Nikolova, M.: Algorithms for finding global minimizers of image segmentation and denoising models. SIAM Journal on Applied Mathematics 66(5), 1632–1648 (2006)
8. Chen, Y., Davis, T., Hager, W., Rajamanickam, S.: Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. ACM Transactions on Mathematical Software 35(3), 1–14 (2008)
9. Engels, C., Stewénius, H., Nistér, D.: Bundle adjustment rules. In: Photogrammetric Computer Vision, PCV (2006)
10. Geman, D., Reynolds, G.: Constrained restoration and the recovery of discontinuities. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(3), 367–383 (1992)
11. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2000)
12. Indelman, V., Roberts, R., Beall, C., Dellaert, F.: Incremental light bundle adjustment. In: Proc. BMVC (2012)
13. Ishikawa, H.: Exact optimization for Markov random fields with convex priors. IEEE Trans. Pattern Anal. Mach. Intell. 25(10), 1333–1336 (2003)
14. Jeong, Y., Nister, D., Steedly, D., Szeliski, R., Kweon, I.S.: Pushing the envelope of modern methods for bundle adjustment. IEEE Trans. Pattern Anal. Mach. Intell. 34(8), 1605–1617 (2012)
15. Jian, Y.D., Balcan, D.C., Dellaert, F.: Generalized subgraph preconditioners for large-scale bundle adjustment. In: Proc. ICCV (2011)
16. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Quarterly of Applied Mathematics 2(2), 164–168 (1944)
17. Li, H., Sumner, R.W., Pauly, M.: Global correspondence optimization for non-rigid registration of depth scans. In: Proc. SGP, pp. 1421–1430. Eurographics Association (2008)
18. Lourakis, M.I., Argyros, A.A.: Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment? In: ICCV, vol. 2, pp. 1526–1531. IEEE (2005)
19. Marquardt, D.: An algorithm for the least-squares estimation of nonlinear parameters. SIAM Journal of Applied Mathematics 11(2), 431–441 (1963)

20. Steffen, R., Frahm, J.-M., Förstner, W.: Relative bundle adjustment based on trifocal constraints. In: Kutulakos, K.N. (ed.) ECCV 2010 Workshops, Part II. LNCS, vol. 6554, pp. 282–295. Springer, Heidelberg (2012)
21. Sunderhauf, N., Protzel, P.: Switchable constraints for robust pose graph slam. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1879–1884. IEEE (2012)
22. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment – A modern synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, pp. 298–372. Springer, Heidelberg (2000)
23. Zhang, J., Boutin, M., Aliaga, D.: Robust bundle adjustment for structure from motion. In: Proc. ICIP (2006)
24. Zollhöfer, M., Nießner, M., Izadi, S., Rehmann, C., Zach, C., Fisher, M., Wu, C., Fitzgibbon, A., Loop, C., Theobalt, C., Stamminger, M.: Real-time non-rigid reconstruction using an rgb-d camera. ACM Transactions on Graphics, TOG (2014)