Self-tuning M-estimators

G. Agamennoni, P. Furgale and R. Siegwart [1, 2]

***Abstract*— M-estimators are the de-facto standard method of robust estimation in robotics. They are easily incorporated into iterative non-linear least-squares estimation and provide seamless and effective handling of outliers in data. However, every M-estimator's robust loss function has one or more tuning parameters that control the influence of different data. The choice of M-estimator and the manual tuning of these parameters is always a source of uncertainty when applying the technique to new data or a new problem.**

In this paper we develop the concept of self-tuning M-estimators. We first make the connection between many common M-estimators and elliptical probability distributions. This connection shows that the choice of M-estimator is an assumption that the residuals belong to a well-defined elliptical distribution. We exploit this implication in two ways. First, we develop an algorithm for tuning the M-estimator parameters during iterative optimization. Second, we show how to choose the correct M-estimator for your data by examining the likelihood of the data given the model. We fully derive these algorithms and show their behavior on a representative example of visual simultaneous localization and mapping.

I. Introduction

We can think of a typical SLAM system as being composed of a front-end and a back-end. The front-end is responsible for collecting sensor measurements, tracking features, performing data association and recognizing loop closures to produce pose constraints. These are then passed on to the back-end, which minimizes a least-squares cost function to produce a sequence of states and a map that are maximally consistent with the data. This separation is so fundamental that a number of standard packages for back-end nonlinear least-squares optimization have been developed, including g^{2}o [1], the Ceres solver [2], and GTSAM [3].

From a probabilistic point of view, minimizing a nonlinear least squares cost function in the back-end is equivalent to maximizing the likelihood of the data given the model *under the assumption that the residuals are Gaussian-distributed*. In practice, this assumption rarely holds. Although many unimodal distributions appear Gaussian, real data often exhibits heavier tails and the front-end may produce incorrect data associations that result in residuals that do not fit the Gaussian model. Least-squares is extremely sensitive to outliers, and even a single anomalous residual can disrupt the estimates to the point that a full recovery is impossible [4]. Hence, correctly modeling the residuals is essential for a robust SLAM system.

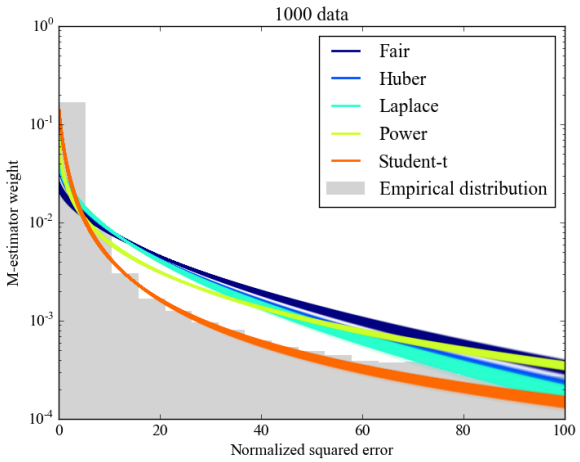

Fig. 1: Finding the right type of M-estimator. The plot shows a histogram of the normalized squared errors, and theoretical probability density function of the squared error distribution for five different M-estimators evaluated on a typical visual SLAM problem. The theoretical density is derived from each of the estimator's equivalent elliptical distribution. In this example, the Student-t distribution is the best fit to the empirical data.

Recently, there have been several attempts at strengthening the weak link between the front- and back-ends [5], [6], [7], [8], [9], [10], [11], [12]. These approaches seek to improve robustness by discarding and/or down-weighting erroneous residuals in the back-end, during non-linear least-squares minimization.

Most of these recent approaches are ad-hoc, e.g. based on M-estimators. Although this is perfectly acceptable, there are fundamental limitations:

1) It is not obvious what the cost function looks like, so it may be hard to see whether it is well behaved;
2) It is not obvious which M-estimator function one should choose when approaching a new problem; and
3) The user needs to manually select tuning parameters that are not necessarily intuitive.

In this paper we propose a novel approach for introducing robustness in data association and state estimation problems such as SLAM, with none of the limitations listed above. Our approach is based on defining a fully probabilistic model, and applying statistical inference techniques on the model.

[1]Autonomous Systems Lab (ASL), ETH Zürich, Switzerland. Contact details: {gabriel.agamennoni, paul.furgale} @mavt.ethz.ch, rsiwgwart@ethz.ch

[2]This work was supported in part by the European Commission (EC) under the grants FP7-600958-SHERPA, FP7-269916-V-Charge, and FP7-610603-EUROPA2

The authors would like to thank Dr Michael Bosse for many fruitful discussions on the subject of robust estimation. The authors would also like to thank the anonymous reviewers for their feedback, which helped us improve the quality of this manuscript.

We show how to choose the correct robust cost function for your data by examining the likelihood of the data given the model (see figure 1), and derive an algorithm to estimate the parameters of this cost function during iterative optimization. The end result is a sophisticated re-weighting mechanism that effectively replaces ad-hoc techniques such as M-estimators. This mechanism can be embedded within a wide class of non-linear least-squares optimization algorithms.

This paper is organized as follows. In section II we review recent approaches to robust SLAM, where M-estimators are ubiquitous. In section III we establish a connection between M-estimators and a general class of probability distributions. In section IV we exploit this connection to show how an M-estimator can be automatically tuned to the data within the framework of least-squares optimization. In section V we validate our approach via an experiment with genuine data from a vision-based SLAM problem. Finally, in section VI we draw conclusions and outline directions for future work.

II. Literature review

We briefly discuss four recent approaches to robustifying the SLAM back-end, namely: max-mixtures [5], [6], switchable constraints [7], [8], [9], dynamic covariance scaling [10], [11] and maximum a posteriori inference [12].

Max-mixture (MM) models, proposed by Olson and Agarwal [5], [6], are an alternative to the single-Gaussian observation model. Unlike the single Gaussian, the MM model allows for multi-modal distributions and hence for multiple data association hypotheses. A max-mixture differs from a standard mixture in that the density function is not a weighted average of Gaussian densities but a point-wise maximum. Exchanging the order of the maximum and logarithm operators purportedly simplifies the negative log-likelihoods and leads to an efficient optimization algorithm. The approach is flawed, however, since it neglects the log-normalization factor. In MMs the log-normalization factor is a function of the states and thus it is non-constant, i.e. it changes after every iteration. Furthermore, the log-likelihood is non-smooth, which means that trust region methods are not directly applicable. There is no guarantee that the algorithm converges, even if the observation models are well behaved.

The method of switchable constraints (SC), introduced by Sünderhauf et al. [7], [8], [9], augments the optimization problem with a set of switch variables. The switch variables give the optimizer the freedom to down-weight individual pose constraints, i.e. to switch them on or off according to how well they agree with the bulk of the data. Experimental results with real data indicate that SC improves the robustness of the overall system. The SLAM front-end need not be perfect for the algorithm to accurately recover the pose graph. A disadvantage of SC is that the functional relationship between the switch and pose variables is ad-hoc. This necessitates adding regularization terms, and corresponding tuning parameters, to avoid nonsensical results. In addition, since the switch variables are part of the optimization, the search space increases with the number of constraints, which makes the algorithm prone to becoming stuck in local optima.

Hee Lee et al. [12] formulated SLAM as a maximum a posteriori (MaP) inference problem. The authors defined a log-likelihood function and a set of weight variables that play the same role as the auxiliary variables in SC. Then, they minimized the negative log-likelihood jointly with respect to the weights and the poses. This is akin to the expectation-maximization (EM) algorithm [13]. The difficulty with this approach stems from the absence of a prior distribution over weight variables, which precludes direct application of the EM algorithm. Once again, ad-hoc regularization terms are added to avoid trivial solutions. The authors justify MaP over EM by claiming that the problem grows exponentially with the number of weight variables, making EM intractable. This is not true. Intractability does not stem from an exponential increase in complexity, but from the fact that it is impossible to evaluate the posterior distribution over the weights without defining a prior. Given a fully factorized prior, weights are conditionally independent given the poses, and thus the posterior factorizes.

Dynamic covariance scaling (DCS), proposed by Agarwal et al. [10], [11], is essentially an M-estimator. Just like SC, it introduces additional variables that enable the optimizer to regulate the effect of each constraint individually. The key difference is that DCS optimizes these variables analytically, and hence achieves much faster convergence. DCS is more computationally efficient than SC. Analytically solving for the auxiliary variables eliminates a great deal of unnecessary computational burden and substantially improves convergence speed. Note, however, that DCS does not marginalize these variables, but minimizes with respect to them in closed form. Hence they are still part of the objective function. As with SC, the formulation is ad-hoc and bears no direct connection to a probabilistic model. Therefore, each auxiliary variable must have its regularization term. Hand-tuning the parameter that controls the strength of the regularization is non-intuitive. Mazuran et al. propose a method to automatically tune the parameter by searching for the parameter value that produces the best a map-consistency score [14]. However, this method is specific to 2D laser maps and is not motivated by the distribution of residuals, but rather by an external measure of map consistency.

There are other approaches which are not based on M-estimators. Namely, Casafranca et al. [15] replace the ℓ_2 norm in least-squares by the ℓ_1 norm, and solve the resulting convex-composite problem with a primal-dual algorithm. Carlone et al. [16] search for the largest subset of coherent measurements by formulating an ℓ_0 optimization problem and applying convex relaxation. However, since the objective function is non-differentiable, these approaches require specialized solvers. We focus on trust-region methods, e.g. Gauss-Newton and Levenberg-Marquardt, which are fairly standard in robotics.

The acknowledgement that the Gaussian distribution is not necessarily the best fit for our data in robotics is not new. Kerl et al. derived an M-estimator based on the t distribution

in order to provide a better fit to dense RGB-D residuals [17]. Rosen et al. [18] show how to incorporate non-Gaussian distributions into non-linear least-squares solvers. However, to the best of our knowledge, ours is the first work to (a) show the connection between M-estimators and elliptical distributions, (b) develop a principled method to select a robust cost function by examining the negative log-likelihood of the data, and (c) propose a strategy for automatically tuning its free parameters.

III. Elliptical distributions

Elliptical distributions [19], [20] are a generalization of the multi-variate Gaussian distribution. As such, they share many of its useful properties, e.g. closure under marginalization and linear combination. In this section we show that certain classes of M-estimators can be interpreted as arising from an elliptical distribution. This has important implications for parameter tuning, since it enables us to apply techniques such as maximum-likelihood estimation and Bayesian inference.

A. Definition

Intuitively, an elliptical random variable has an elliptically contoured probability density function, hence its name. The Gaussian, Laplace and Cauchy distributions are special cases of elliptical distributions. The following definition formalizes this notion.

Definition 1 (elliptical distribution)**.** *Let ψ be a real-valued function on the positive real line such that*

$$c\left(\psi\right) \triangleq \frac{1}{2}\int_0^\infty t^{\frac{d}{2}-1}\exp\left(-\frac{1}{2}\psi\left(t\right)\right)\mathrm{d}t < \infty. \tag{1}$$

Let $\boldsymbol{\mu} \in \mathbb{R}^d$ and let $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ be symmetric and positive-definite. A random variable $\mathbf{x}$ follows a d-dimensional elliptical distribution with location $\boldsymbol{\mu}$, scale $\boldsymbol{\Sigma}$, and generator ψ, noted

$$\mathbf{x} \sim \mathcal{E}\left(\psi, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right),$$

if its probability density function is

$$p\left(\mathbf{x}\right) = \frac{1}{2}\Gamma\left(d/2\right)\pi^{-\frac{d}{2}}c\left(\psi\right)^{-1}\det\left(\boldsymbol{\Sigma}\right)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\psi\left(\left(\mathbf{x}-\boldsymbol{\mu}\right)^\top\boldsymbol{\Sigma}^{-1}\left(\mathbf{x}-\boldsymbol{\mu}\right)\right)\right), \tag{2}$$

where Γ is the Gamma function [21].

B. Mixture expansion

Any elliptically distributed random variable can be expanded as a scale mixture of Gaussians [22]. Specifically, if $\mathbf{x} \sim \mathcal{E}\left(\psi, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$, then there exists a positive random variable w such that, for all $w > 0$,

$$\mathbf{x}|w \sim \mathcal{N}\left(\boldsymbol{\mu}, w^{-1}\boldsymbol{\Sigma}\right) \tag{3}$$

where $\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ denotes a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. In other words, there exists a probability density $p\left(w\right)$, $w > 0$, such that the convolution of $p\left(w\right)$ with (3) yields (2) (refer to Chu [22], page 500, for a proof).

The mixing weight w is an auxiliary variable that originates from the expansion. This is convenient for parameter estimation since it renders $\mathbf{x}$ conditionally Gaussian. In non-linear least-squares, if we somehow knew the exact value of w, then the negative log-likelihood would be a quadratic function of the residuals.

C. Robust non-linear least squares

Consider the following non-linear least-squares problem. Given a set $\{\mathbf{x}_k\}$ of data, we wish to minimize the robust loss function

$$\ell\left(\boldsymbol{\theta}\right) = \frac{1}{2}\sum_{k=1}^{n}\psi\left(\left\|\mathbf{x}_k - \mathbf{f}_k(\boldsymbol{\theta})\right\|^2\right) \tag{4}$$

with respect to $\boldsymbol{\theta}$. Functions $\mathbf{f}_k$ and ψ are differentiable, and ψ satisfies (1).

For instance, in a visual-based SLAM problem the $\mathbf{x}_k$ are image keypoints, and $\boldsymbol{\theta}$ comprises robot poses and landmark positions in Euclidean space.

Minimizing the loss function directly is generally difficult, since $\boldsymbol{\theta}$ may be high-dimensional and ψ may be non-convex. In these cases we can turn to the probabilistic interpretation of the $\mathbf{x}_k$. Specifically, we assume that

$$\mathbf{x}_k \sim \mathcal{E}\left(\psi, \mathbf{f}_k(\boldsymbol{\theta}), \mathbf{I}\right), \tag{5}$$

where $\mathbf{I}$ is the identity matrix. From (2) we know that, up to an additive constant, $\ell\left(\boldsymbol{\theta}\right)$ is equal to $-\sum_k \ln p\left(\mathbf{x}_k|\boldsymbol{\theta}\right)$. Hence minimizing (4) is equivalent to minimizing the negative log-likelihood of the data, assuming they are generated by an elliptical distribution.

Robust least squares is equivalent to maximum-likelihood estimation with elliptically contoured data. One of the advantages of this interpretation is that it enables us to transform the original, non-convex problem into a series of iteratively re-weighted convex sub-problems. This is accomplished via the EM algorithm.

D. Expectation-maximization

The EM algorithm is a procedure for iteratively estimating a set of parameters from data according to the maximum-likelihood criterion. EM is well-suited for problems involving latent variables. Each iteration consists of two steps: the E-step and the M-step. The E-step imputes the latent variables by evaluating expectations with respect to their posterior distribution, while the M-step updates the parameters in order to increase the likelihood of the data.

In the context of robust least squares, we can apply EM to find $\boldsymbol{\theta}$. If we invoke the mixture expansion of elliptical random variables we can express (5) as

$$\mathbf{x}_k|w_k \sim \mathcal{N}\left(\mathbf{f}_k(\boldsymbol{\theta}), w_k^{-1}\mathbf{I}\right).$$

The w_k are latent variables. The E-step of EM computes the

expected value of the negative log-likelihood,

$$-\sum_{k=1}^{n}\mathrm{E}\left[\ln p\left(\mathbf{x}_k, w_k\right)\right] = \frac{1}{2}\sum_{k=1}^{n}\mathrm{E}\left[w_k|\mathbf{x}_k\right]\left\|\mathbf{x}_k-\mathbf{f}_k(\boldsymbol{\theta})\right\|^2+\ldots, \quad (6)$$

where the expectations are taken with respect to the w_k, and the dots denote terms independent of $\boldsymbol{\theta}$. The M-step updates $\boldsymbol{\theta}$ in order to minimize (6). If we alternate between the E- and M-steps, we will eventually converge to a local minimum of the loss function ℓ.

The following proposition provides an analytic expression for computing the expectations in (6).

Proposition 1 (expected weight)**.** *Let* $\mathbf{x}\sim\mathcal{E}\left(\psi,\boldsymbol{\mu},\boldsymbol{\Sigma}\right)$ *such that* ψ *is differentiable at* $t=(\mathbf{x})=(\mathbf{x}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$*,* $\mathbf{x}\neq\boldsymbol{\mu}$*, and* $\psi'(t)\geq 0$*. Then,*

$$\mathrm{E}\left[w|\mathbf{x}\right]=\psi'(t) \quad (7)$$

Proof. See appendix I for details. □

Note how we have transformed the original robust problem into an iteratively re-weighted least-squares (IRLS) problem. In each iteration we compute the weights as

$$\mathrm{E}\left[w_k|\mathbf{x}_k\right]=\psi'\Big(\left\|\mathbf{x}_k-\mathbf{f}_k(\boldsymbol{\theta})\right\|^2\Big) \quad (8)$$

where ψ' is the derivative of ψ. Then, we apply a standard optimization method, such as Gauss-Newton or Levenberg-Marquardt, to approximately minimize the weighted sum-of-squares error. Since this is a convex-composite function, we don't need to worry about the Hessian becoming negative-definite. Convergence is guaranteed since the E- and M-steps monotonically decrease the negative log-likelihood.

E. M-estimators and elliptical distribution

There is a clear connection between M-estimators and elliptical distributions. Given an M-estimator, we can associate it with a generator ψ, assuming that (1) converges, and derive its corresponding elliptical distribution from 1.

The converse is not necessarily true. Some M-estimators have no corresponding elliptical distribution. In order for (1) to converge it is a necessary condition that

$$\lim_{t\to\infty}\left[t^{\frac{d}{2}-1}\exp\left(-\frac{1}{2}\psi\left(t\right)\right)\right]=0 \quad (9)$$

This condition does not hold for the Geman-McLure, Tukey and Welsch M-estimators (Zhang [23], page 70), hence they have no corresponding elliptical distribution.

DCS [10], [11] is amongst those without an elliptical equivalent. Its weight function is given by

$$\psi'(t)=\min\left\{1,\left(\frac{2\phi}{\phi+t}\right)^2\right\}$$

where $\phi>0$ is a tuning constant that controls its shape (see [10], equation (15)). The corresponding generator is

$$\psi(t)=\begin{cases}t & t\leq\phi\\ \phi+2\phi\left(1-\frac{2\phi}{\phi+t}\right) & t>\phi\end{cases}$$

However, since ψ is asymptotic, it violates (9), and thus the integral in (1) diverges.

Table I contains examples of commonly used M-estimators which do have an elliptical equivalent. In all examples $\phi>0$ is a tuning constant. By virtue of this equivalence, we could potentially tune ϕ automatically by minimizing the negative log-likelihood of the data

$$\min_{\boldsymbol{\theta},\phi}\left[n\ln c\left(\psi\right)+\frac{1}{2}\sum_{k=1}^{n}\psi\left(\left\|\mathbf{x}_k-\mathbf{f}_k(\boldsymbol{\theta})\right\|^2\right)\right] \quad (10)$$

where $\psi=\psi\left(\cdot,\phi\right)$. Unlike the original loss function (4), the regularized loss in (10) is well-behaved. The minimization problem does not yield a trivial solution since the $n\ln c\left(\psi\right)$ term prevents ψ from becoming identically zero.

TABLE I: Common M-estimators and their generators

	ψ	ψ'
Laplace	$2\phi\sqrt{t}$	$\phi/\sqrt{t}$
Power exp.	t^{ϕ}/ϕ	$t^{\phi-1}$
Cauchy	$\phi\ln\left(1+t/\phi\right)$	$\left(1+t/\phi\right)^{-1}$
Student-t	$\left(\phi+d\right)\ln\left(1+t/\phi\right)$	$\frac{\phi+d}{\phi+t}$
"Fair"	$2\phi\left[\sqrt{t/\phi}-\ln\left(1+\sqrt{t/\phi}\right)\right]$	$\left(1+\sqrt{t/\phi}\right)^{-1}$
Huber	$\begin{cases}t & t\leq\phi\\ 2\sqrt{\phi t}-\phi & t>\phi\end{cases}$	$\begin{cases}1 & t\leq\phi\\ \sqrt{\phi/t} & t>\phi\end{cases}$

IV. Automatic tuning of M-estimators

Most common M-estimators have at least one tuning constant —all of the examples in III-E have one. In general, selecting values for these constants is non-trivial and non-intuitive. In this section we apply the results from section III to show how an M-estimator can be tuned automatically within the Bayesian framework. Our tuning mechanism is computationally efficient, simple to use, and is easily integrated into modern optimization algorithms.

A. Direct and gradient-based minimization

Note that minimizing (10) is feasible either via direct search [24] or gradient-based methods. From (1), the derivative of the first term in (10) with respect to ϕ is

$$n\frac{\partial}{\partial\phi}\ln c\left(\psi\right) = -\frac{n}{4\,c\left(\psi\right)}\int_0^{\infty}t^{\frac{d}{2}-1}\exp\left(-\frac{1}{2}\psi\left(t\right)\right)\frac{\partial\psi}{\partial\phi}\left(t\right)\mathrm{d}t,$$

which can be evaluated via numerical integration. However, we prefer to make no assumptions about ψ —not even that $\partial\psi/\partial\phi$ exists.

B. Adaptive importance sampling

The generator function is usually a highly non-linear function of the tuning constants. Often, ψ it is not continuously differentiable with respect to ϕ, e.g. the Huber M-estimator in table I. Rather than adding them to the optimization problem and treating them as unknowns, we approximate the distribution over tuning constants via adaptive importance sampling.

Let $\boldsymbol{\phi}_j$ be a vector of real-valued tuning constants. Suppose that, a priori, $\boldsymbol{\phi}_j$ follows a distribution with probability density function $p\left(\boldsymbol{\phi}\right)$. Let $\Phi = \left\{\boldsymbol{\phi}_j\right\}$ be a finite, i.i.d. sample generated according to a proposal density $q\left(\boldsymbol{\phi}\right)$.

From the law of total expectation, and the law of large numbers, we can approximate the weights as

$$\mathrm{E}\left[w_k|\mathbf{x}_k\right] \simeq \sum_{j=1}^{|\Phi|} \rho_j \, \mathrm{E}\left[w_k\middle|\mathbf{x}_k, \boldsymbol{\phi}_j\right] \tag{11}$$

where

$$\mathrm{E}\left[w_k\middle|\mathbf{x}_k, \boldsymbol{\phi}_j\right] = \psi_j'\Big(\left\|\mathbf{x}_k - \mathbf{f}_k(\boldsymbol{\theta})\right\|^2\Big)$$

similarly to (8), and

$$\rho_j \propto p\left(\boldsymbol{\phi}_j\right) q\left(\boldsymbol{\phi}_j\right)^{-1} \prod_{k=1}^{n} p\left(\mathbf{x}_k\middle|\boldsymbol{\theta}, \boldsymbol{\phi}_j\right) \tag{12}$$

The likelihood terms are

$$p\left(\mathbf{x}_k\middle|\boldsymbol{\theta}, \boldsymbol{\phi}_j\right) = \frac{1}{2}\Gamma\left(d/2\right)\pi^{-\frac{d}{2}} c\left(\psi_j\right)^{-1} \exp\left(-\frac{1}{2}\psi_j\Big(\left\|\mathbf{x}_k - \mathbf{f}_k(\boldsymbol{\theta})\right\|^2\Big)\right) \tag{13}$$

where $\psi_j = \psi\left(\cdot, \boldsymbol{\phi}_j\right)$ denotes the generator function with tuning constants $\boldsymbol{\phi}_j$. Note the proportionality sign in (12); a normalization constant ensures that $\sum_j \rho_j = 1$.

C. Bayesian model comparison

The proportionality sign in (12) implies the presence of a normalization constant. This constant is a sample approximation of the marginal likelihood of the data,

$$p\left(\mathbf{x}_1, \ldots, \mathbf{x}_n|\boldsymbol{\theta}\right) \simeq \sum_{j=1}^{|\Psi|} p\left(\boldsymbol{\phi}_j\right) q\left(\boldsymbol{\phi}_j\right)^{-1} \prod_{k=1}^{n} p\left(\mathbf{x}_k\middle|\boldsymbol{\theta}, \boldsymbol{\phi}_j\right) \tag{14}$$

also known as the model evidence.

Equation (14) enables Bayesian model selection. By definition, the model evidence is the likelihood of the data for a given type of estimator, without assuming any particular tuning constants. Having marginalized out the tuning constants, the remaining variables are $\boldsymbol{\theta}$, and the identity of the estimator itself.

Suppose we wish to compare two types of M-estimators, M_a and M_b. Assuming that both types are equally likely, the Bayes factor for M_a against M_b is

$$\frac{p\left(M_a|\mathbf{x}, \boldsymbol{\theta}\right)}{p\left(M_b|\mathbf{x}, \boldsymbol{\theta}\right)} = \frac{p_a(\mathbf{x}|\boldsymbol{\theta})}{p_b(\mathbf{x}|\boldsymbol{\theta})}$$

where p_a and p_b are given by (14) with the corresponding estimator type. We select M_a if the Bayes factor is greater than one, and M_b otherwise.

D. Proposal and prior distributions

Adaptive importance sampling requires a proposal density function q in order to generate the $\boldsymbol{\phi}_j$. We choose a Gaussian proposal with mean and covariance given by the weighted sample $\left\{\left(\rho_j, \boldsymbol{\phi}_j\right)\right\}$. That is,

$$q\left(\boldsymbol{\phi}\right) \propto \exp\left(-\frac{1}{2}\left(\boldsymbol{\phi} - \boldsymbol{\mu}\right)^\top \boldsymbol{\Sigma}^{-1}\left(\boldsymbol{\phi} - \boldsymbol{\mu}\right)\right) \tag{15}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are given by

$$\boldsymbol{\mu} = \sum_j \rho_j \boldsymbol{\phi}_j \tag{16a}$$

$$\boldsymbol{\Sigma} = \sum_j \rho_j \left(\boldsymbol{\phi}_j - \boldsymbol{\mu}\right)\left(\boldsymbol{\phi}_j - \boldsymbol{\mu}\right)^\top + r\,\mathbf{I} \tag{16b}$$

and $r > 0$ is a regularization constant. Sampling is repeated m times between each optimization step to allow $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ to settle. In our implementation $r = 10^{-3}$ and $m = 3$.

Importance sampling also requires a prior p to compute the ρ_j in (12). We choose a Gaussian prior with zero mean and spherical covariance,

$$p\left(\boldsymbol{\phi}\right) \propto \exp\left(-\frac{1}{2}\boldsymbol{\phi}^\top\boldsymbol{\phi}\right) \tag{17}$$

E. Algorithm and pseudo-code

Algorithm 1 contains pseudo-code showing how to augment a non-linear least-squares solver with a self-tuning M-estimator. The algorithm iteratively optimizes the robust least-squares objective function with respect to $\boldsymbol{\theta}$, adaptively tuning the M-estimator and re-weighting the error terms in between iterations.

repeat
  `// sampling`
1. Initialize $q = p$, with p given by (17)
   for $i = 1, \ldots, m$ **do**
     for $j = 1, \ldots, |\Phi|$ **do**
2. Generate $\boldsymbol{\phi}_j$ with density q
3. $\psi_j \leftarrow \psi\left(\cdot, \boldsymbol{\phi}_j\right)$
4. Compute ρ_j as in (12)
     end
5. Normalize the ρ_j so that $\sum_j \rho_j = 1$
6. Build q from $\left\{\left(\rho_j, \boldsymbol{\phi}_j\right)\right\}$ according to (15)
   end
   `// re-weighting`
   for $k = 1, \ldots, n$ **do**
7. $\omega_k \leftarrow \sum_j \rho_j \, \psi_j'\Big(\left\|\mathbf{x}_k - \mathbf{f}_k(\boldsymbol{\theta})\right\|^2\Big)$
   end
   `// optimization`
8. Compute the search direction $\Delta\boldsymbol{\theta}$ by solving the linearized sub-problem (18) with weights $\{\omega_k\}$
9. $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$
until convergence

Algorithm 1. Self-tuning M-estimation

The least-squares solver comprises lines 8 and 9. In line 8 we compute the search direction by solving a linearized sub-problem, and in line 9 we update the linearization point. The sub-problem is of the form

$$\min_{\Delta\boldsymbol{\theta}} \left[\frac{1}{2} \sum_{k=1}^{n} \omega_k \left\| \mathbf{x}_k - \mathbf{f}_k(\boldsymbol{\theta}) - \frac{\partial \mathbf{f}_k}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta})\, \Delta\boldsymbol{\theta} \right\|^2 + \mu \mathbf{I} \right] \quad (18)$$

where $\mu = 0$ yields the Gauss-Newton direction, and $\mu > 0$ produces a Levenberg-Marquardt direction. For all but the smallest of problems, these two lines are by far the most computationally expensive part of the algorithm.

Line 7 is the re-weighting step. We iterate over the error terms and update their weights according to (11). Note that this step can be parallelized.

Lines 1 to 9 implement the adaptive importance sampler described in IV-B. In line 2 we generate candidate tuning constants from the proposal distribution q, and in line 4 we compute their importance weights as given by (12). In line 6 we rebuild the proposal based on the weighted sample.

The most expensive part of adaptive importance sampling is computing the importance weights. Specifically, evaluating $c(\psi_j)$ for every ψ_j. This involves numerically evaluating the integral in (1). In our implementation we use the adaptive version of Simpson's method. For large enough problems, the cost incurred by one-dimensional numerical integration is negligible compared to the cost of solving (18) to obtain the search direction.

V. Experimental results

In this section we demonstrate M-estimator self-tuning on a visual SLAM problem. In visual SLAM the robot is equipped with an array of cameras. The images collected by the cameras are processed to extract keypoints, which are then matched to one another and triangulated to estimate the position of the landmarks. The robot estimates its pose and the positions of the landmarks simultaneously by solving a robust least-squares problem. The $\mathbf{x}_k$ are the pixel coordinates of the keypoints, and $\boldsymbol{\theta}$ contains the sequence of robot poses and the landmark positions.

A. Test platform and dataset

The data was collected by a modified Volkswagen Golf VI, a test platform in the European project V-Charge [25]. The modifications include the integration of four monocular fish-eye cameras for perception and self-localization. The cameras provide 360° imagery of the vehicle's surroundings.

For our experiment we consider a dataset collected while driving the vehicle around an indoor parking lot. The cameras are triggered synchronously at a rate of 12.5 Hz. Each of them has a nominal field of view of 185° and a resolution of 1.3 Mp. From the images we extract keypoints using the BRISK detector [26].

The keypoint observations are generated by the visual localization system described in [27]. The tBRISK keypoints are tracked within each image stream and triangulated to create landmarks. Optimization consists of standard multi-camera bundle adjustment to estimate the vehicle poses and landmark locations given all of the keypoint measurements.

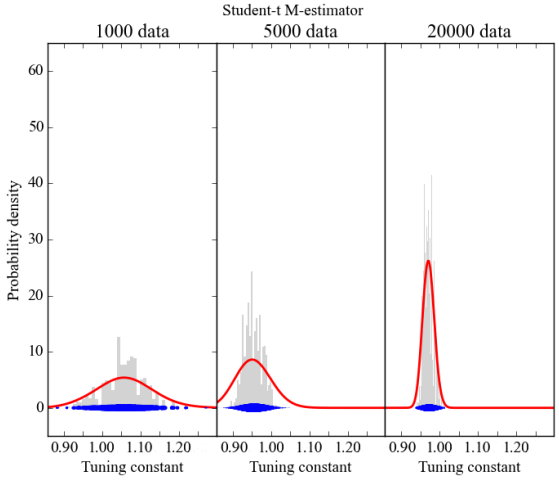

Fig. 2: Tuning the Student-t M-estimator with increasing number of data. Each of the panels shows the density of the proposal distribution for the tuning constant ϕ, and a weighted sample drawn from the proposal. From left to right, the number of data are 500, 5000 and 50000.

B. Estimator self-tuning

We evaluated 5 different types of M-estimators: the "fair," Huber, Laplace, power exponential and Student-t (see table I).[1] For each type, we ran the sampling module of algorithm 1 —namely, lines 1 to 7—on 3 subsets of the data. The subsets contained 1000, 5000 and 20000 keypoint observations. We re-parameterized ϕ by $\ln \phi$ so that it is not constrained to be positive.

Figure 2 shows the result of tuning the Student-t estimator with increasingly larger subsets of the data. In each panel the proposal distribution $q(\phi)$ is plotted in red, the sample distribution $\{(\rho_j, \phi_j)\}$ in blue and the weighted histogram in gray. The size of the blue dots and the height of the bars are both proportional to the importance weights.

As the number of data increase, so does the information gain. This causes the distribution to become more and more peaked around its mode. In all cases the proposal aligns well with the sample. The effective sample size is 172 on average, indicating that the adaptive importance sampler is efficient.

Figure 3 shows the weight functions for each type of estimator. For a given type, the weight function $\psi'(\cdot)$ is plotted for each ϕ_j in the sample as a colored line. Transparency values are set according to the weight ρ_j. The functions are aligned so that $\psi'(1) \simeq 1$ for clarity. The number of data is 5000 in this case.

C. Implied error distribution

Let $\mathbf{x} \sim \mathcal{E}(\psi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$. and define the normalized squared error statistic $t = (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})$. t follows a distri-

[1] We left out the Cauchy M-estimator as it is very similar to the Student-t.

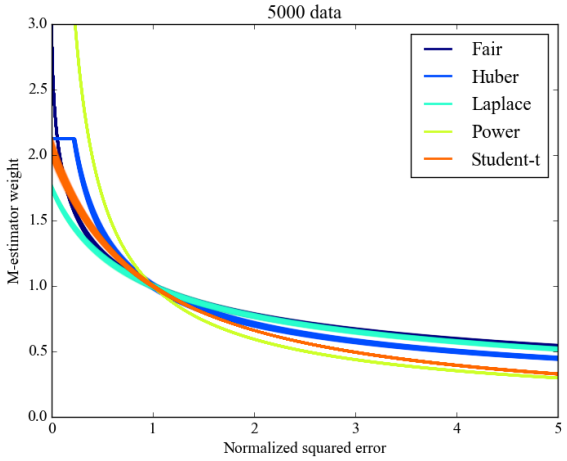

Fig. 3: Weight functions for different M-estimators, tuned with 5000 data. For a given M-estimator, the plot shows the weight function $\psi'(\cdot)$ for each ϕ_j in the sample, with alpha values proportional to its weight. All curves are scaled so that they pass through $(1, 1)$.

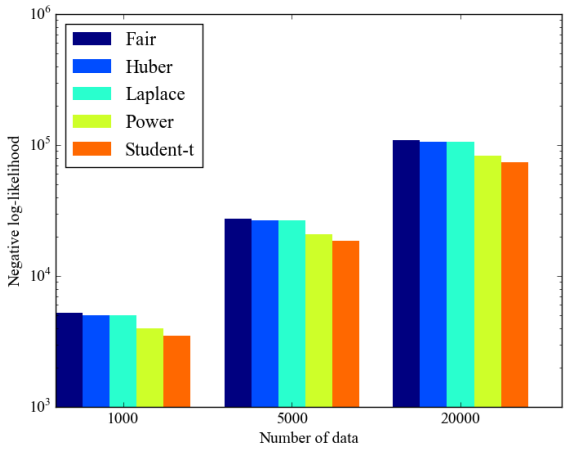

Fig. 4: A test to find the right M-estimator. The plot shows the negative log-likelihood of the data attained by each of the M-estimators for a given dataset. Estimators with a lower negative log-likelihood are better suited for the data.

bution with probability density function

$$p(t) \propto t^{\frac{d}{2}-1} \exp\left(-\frac{1}{2}\psi(t)\right)$$

Refer to the proof of theorem 2.2.5 in [28] for a derivation.

The density $p(t)$ tells us how well a given M-estimator fits the data. In theory, if $\mathbf{x}$ truly follows an elliptical distribution, we expect that $p(t)$ agrees with the empirical distribution of normalized squared errors. For example, if $\mathbf{x}$ is Gaussian $p(t)$ is a chi-square density function.

Figure 1 shows the theoretical density of the distribution over normalized squared errors for each type of estimator. For each ϕ_j in the sample, $p(t)$ is plotted as colored line, with its color given by the type of estimator and its alpha value proportional to ρ_j. The histogram of the normalized squared errors is plotted in gray. The number of data is 1000 in this case.

The Student-t achieves a much better fit than the rest of the estimators. This is due to the tail behavior of their equivalent elliptical distributions. The tails of the t decay as

$$(1 + t/\phi)^{-\frac{\phi+d}{2}}$$

(see table I). In contrast, the tails of the "fair," Laplace and Huber decay at rate that is asymptotically $O\left(\exp\left(-\sqrt{t}\right)\right)$. This is fast compared the the t.

The power exponential estimator can tune its rate of decay by adjusting ϕ. However, doing so involves a compromise between fitting small errors and large errors, a compromise that the Student-t is able to negotiate better.

D. Finding the right estimator

In IV-C we discussed Bayesian model comparison, and more specifically, how the marginal likelihood of the data indicates the goodness-of-fit for a given type of M-estimator. Figure 4 plots the negative log-likelihood of the data for each type. The height of each bar is equal to the negative logarithm of (14). Thus a lower bar indicates a better fit.

From the figure we observe that the Student-t consistently produces a better fit than all the other estimators, for all datasets. This agrees with what we observed previously from figure 1. The difference is that we now have a numerical value to confirm this. Note that there is no extra cost involved in computing this value; it comes for free as a byproduct of algorithm 1.

VI. Conclusion and Future Work

In this paper we presented a probabilistic interpretation of M-estimators. An M-estimator is understood as arising from an elliptical distribution. This connection enables us to automatically tune an M-estimator by minimizing the negative log-likelihood implied by the elliptical density function. The problem is well-posed thanks to the regularization effect of the log-normalization constant.

In our approach we went one step further and derived an approximate Bayesian estimation algorithm based on adaptive importance sampling. Albeit approximate, our algorithm is general and computationally cheap. Experimental results with genuine data showed that we can effectively select an M-estimator and tune it while avoiding over-fitting.

As part of our derivation, we established a key property related to the scale mixture representation of elliptical random variables. Namely, that the expected conditional weight is equal to the derivative of the generator function. This property allowed us to apply the EM algorithm and transform the robust non-linear least-squares problem into an iteratively re-weighted problem.

In upcoming work we will perform an extensive experimental evaluation in order to determine when and to what extent M-estimator tuning is effective. Standard datasets [29] will provide the benchmark.

APPENDIX I
PROOF OF PROPOSITION 1

Without loss of generality, we assume $\boldsymbol{\mu} = \mathbf{0}$ and $\boldsymbol{\Sigma} = \mathbf{I}$. Let $p(\mathbf{x}|w)$ be the probability density function implied by (3), and let $p(w)$ be the density of the prior over w. Bayes' rule states that

$$\int_0^\infty p(w)\, p(\mathbf{x}|w)\, \mathrm{d}w = p(\mathbf{x})$$

Replacing $p(\mathbf{x})$ and $p(\mathbf{x}|w)$ gives

$$(2\pi)^{-\frac{d}{2}} \int_0^\infty p(w)\, w^{\frac{d}{2}} \exp\left(-\frac{w}{2}\mathbf{x}^\top\mathbf{x}\right) \mathrm{d}w = \Gamma(d/2)\, \pi^{-\frac{d}{2}}\, c(\psi)^{-1} \exp\left(-\frac{1}{2}\psi\left(\mathbf{x}^\top\mathbf{x}\right)\right)$$

Differentiating both sides with respect to $\mathbf{x}$ leads to

$$-(2\pi)^{-\frac{d}{2}} \int_0^\infty w\, p(w)\, w^{\frac{d}{2}} \exp\left(-\frac{w}{2}\mathbf{x}^\top\mathbf{x}\right) \mathrm{d}w\, \mathbf{x} = -\Gamma(d/2)\, \pi^{-\frac{d}{2}}\, c(\psi)^{-1} \exp\left(-\frac{1}{2}\psi\left(\mathbf{x}^\top\mathbf{x}\right)\right) \psi'\left(\mathbf{x}^\top\mathbf{x}\right)\mathbf{x}$$

Substituting back $p(w)$, $p(\mathbf{x}|w)$ and $p(\mathbf{x})$ produces

$$-\int_0^\infty w\, p(w)\, p(\mathbf{x}|w)\, \mathrm{d}w\, \mathbf{x} = -p(\mathbf{x})\, \psi'\left(\mathbf{x}^\top\mathbf{x}\right)\mathbf{x}$$

Since ψ is differentiable at $t = \mathbf{x}^\top\mathbf{x}$, $\psi'(t) < \infty$ and hence $p(\mathbf{x}) > 0$. Dividing both sides by $p(\mathbf{x})$ and applying Bayes' rule once again yields

$$\underbrace{-\int_0^\infty w\, \frac{\overbrace{p(\mathbf{x}|w)\, p(w)}^{p(w|\mathbf{x})}}{p(\mathbf{x})}\, \mathrm{d}w}_{\mathrm{E}[w|\mathbf{x}]}\, \mathbf{x} = -\psi'(t)\, \mathbf{x}$$

Since $\mathbf{x} \neq \mathbf{0}$, there is at least one non-zero element in $\mathbf{x}$. Dividing the by the non-zero element in the corresponding equation, we arrive at (7).

REFERENCES

[1] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 3607–3613.

[2] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[3] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Center for Robotics and Intelligent Machines, Georgia Institute of Technology, Tech. Rep. GT-RIM-CP&R-2012-002, Sept 2012. [Online]. Available: https://research.cc.gatech.edu/borg/sites/edu.borg/files/downloads/gtsam.pdf

[4] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 108–117, September 2006.

[5] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," in *Robotics: Science and Systems*, 2012.

[6] ——, "Inference on networks of mixtures for robust robot mapping," University of Michigan & Universität Freiburg, Tech. Rep., 2013.

[7] N. Sünderhauf, "Robuts optimization for simultaneous localization and mapping," Ph.D. dissertation, Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität Chemnitz, 2012.

[8] N. Sünderhauf and P. Protzel, "Towards a robust back-end for pose graph SLAM," in *International Conference on Robotics and Automation*, 2012.

[9] ——, "Switchable constraints for robust pose graph SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[10] P. Agarwal, G. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, "Robust map optimization using dynamic covariance scaling," in *International Conference on Robotics and Automation*, 2013.

[11] ——, "Dynamic covariance scaling for robust map optimization," in *ICRA Workshop on Robust and Multimodal Inference in Factor Graphs*, 2013.

[12] G. Hee Lee, F. Fraundorfer, and M. Pollefeys, "Robust pos graph loop closures with expectation-maximization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[13] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[14] M. Mazuran, G. Diego Tipaldi, L. Spinello, W. Burgard, and C. Stachniss, "A statistical measure for map consistency in slam," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 3650–3655.

[15] J. Casafranca, L. Paz, and P. Piniés, "A back-end ℓ_1-norm-based solution for factor graph SLAM," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, November 2013, pp. 17–23.

[16] L. Carlone, A. Censi, and F. Dellaert, "Selecting good measurements via ℓ_1 relaxation: A convex approach for robust estimation over graphs," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 2667–2674.

[17] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, pp. 3748–3754.

[18] D. Rosen, M. Kaess, and J. Leonard, "Robust incremental online inference over sparse factor graphs: Beyond the gaussian case," in *International Conference on Robotics and Automation*, 2013.

[19] K.-T. Fang, S. Kotz, and K. Ng, *Symmetric Multivariate and Related Distributions*. Chapman & Hall, 1987.

[20] E. Gómez, M. Gómez-Villegas, and J. Miguel Marín, "A survey on continuous elliptical vector distributions," *Revista Matemática Complutense*, vol. 16, no. 1, pp. 345–361, 2003.

[21] M. Abramowitz and I. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, 1972.

[22] K.-C. Chu, "Estimation and decision for linear systems with elliptical random processes," *IEEE Transactions on Automatic Control*, vol. 18, no. 5, pp. 499–505, October 1973.

[23] Z. Zhang, "Parameter estimation techniques: A tutorial with applications to conic fitting," *Image and Vision Computing*, vol. 15, no. 1, pp. 59–76, January 1997.

[24] J. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[25] P. Furgale, U. Schwesinger, M. Rufli, W. Derendarz, H. Grimmett, P. Mühlfellner, S. Wonneberger, J. T. S. Rottmann, B. Li, B. Schmidt, T. N. Nguyen, E. Cardarelli, S. Cattani, S. Brüning, S. Horstmann, M. Stellmacher, H. Mielenz, K. Köser, M. Beermann, C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, R. Iser, R. Triebel, I. Posner, P. Newman, L. Wolf, M. Pollefeys, S. Brosig, J. Effertz, C. Pradalier, and R. Siegwart, "Toward automated driving in cities using close-to-market sensors, an overview of the v-charge project," in *IEEE Intelligent Vehicles Symposium*, June 2013, pp. 809–816.

[26] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *roceedings of the IEEE International Conference on Computer Vision*, November 2011, pp. 2548–2555.

[27] P. Muehlfellner, P. Furgale, W. Derendarz, and R. Philippsen, "Evaluation of fisheye camera-based visual multi-session localization in a real-world scenario," in *IEEE Intelligent Vehicles Symposium (IV)*, June 2013, pp. 56–62. [Online]. Available: bib/muehlfellner_iv13.pdf

[28] K. Fang and Y. Zhang, *Generalized Multi-variate Analysis*. Springer-Verlag, 1990.

[29] A. Howard and N. Roy, "Radish: The robotics dataset repository," 2003. [Online]. Available: http://radish.sourceforge.net/