

# 1 Overview of Trust-region Methods

For nice figures, see [1].

We just deal here with a small subset of trust-region methods, specifically approximating the cost function as quadratic using Newton’s method, and using the Dogleg method and later to include Steihaug’s method.

The overall goal of a nonlinear optimization method is to iteratively find a local minimum of a nonlinear function

$$\hat{x} = \underset{x}{\operatorname{argmin}} f(x)$$

where  $f(x) \rightarrow \mathbb{R}$  is a scalar function. In GTSAM, the variables  $x$  could be manifold or Lie group elements, so in this document we only work with *increments*  $\delta x \in \mathbb{R}^n$  in the tangent space. In this document we specifically deal with *trust-region* methods, which at every iteration attempt to find a good increment  $\|\delta x\| \leq \Delta$  within the “trust radius”  $\Delta$ .

Further, most nonlinear optimization methods, including trust region methods, deal with an approximate problem at every iteration. Although there are other choices (such as quasi-Newton), the Newton’s method approximation is, given an estimate  $x^{(k)}$  of the variables  $x$ ,

$$f(x^{(k)} \oplus \delta x) \approx M^{(k)}(\delta x) = f^{(k)} + g^{(k)\top} \delta x + \frac{1}{2} \delta x^\top G^{(k)} \delta x, \quad (1)$$

where  $f^{(k)} = f(x^{(k)})$  is the function at  $x^{(k)}$ ,  $g^{(x)} = \left. \frac{\partial f}{\partial x} \right|_{x^{(k)}}$  is its gradient, and  $G^{(k)} = \left. \frac{\partial^2 f}{\partial x^2} \right|_{x^{(k)}}$  is its Hessian (or an approximation of the Hessian).

Trust-region methods adaptively adjust the trust radius  $\Delta$  so that within it,  $M$  is a good approximation of  $f$ , and then never step beyond the trust radius in each iteration. When the true minimum is within the trust region, they converge quadratically like Newton’s method. At each iteration  $k$ , they solve the *trust-region subproblem* to find a proposed update  $\delta x$  inside the trust radius  $\Delta$ , which decreases the approximate function  $M^{(k)}$  as much as possible. The proposed update is only accepted if the true function  $f$  decreases as well. If the decrease of  $M$  matches the decrease of  $f$  well, the size of the trust region is increased, while if the match is not close the trust region size is decreased.

Minimizing Eq. 1 is itself a nonlinear optimization problem, so there are various methods for approximating it, including Dogleg and Steihaug’s method.

## 2 Adapting the Trust Region Size

As mentioned in the previous section, we increase the trust region size if the decrease in the model function  $M$  matches the decrease in the true cost function  $S$  very closely, and decrease it if they do not match closely. The closeness of this match is measured with the *gain ratio*,

$$\rho = \frac{f(x) - f(x \oplus \delta x_d)}{M(0) - M(\delta x_d)},$$

where  $\delta x_d$  is the proposed dogleg step to be introduced next. The decrease in the model function is always non-negative, and as the decrease in  $f$  approaches it,  $\rho$  approaches 1. If the true cost function increases,  $\rho$  will be negative, and if the true cost function decreases even more than predicted by  $M$ , then  $\rho$  will be greater than 1. A typical update rule, as per Lec. 7-1.2 of [1] is:

$$\Delta_{k+1} \leftarrow \begin{cases} \Delta_k/4 & \rho < 0.25 \\ \min(2\Delta_k, \Delta_{max}), & \rho > 0.75 \\ \Delta_k & 0.75 > \rho > 0.25 \end{cases}$$

where  $\Delta_k \triangleq \|\delta x_d\|$ . Note that the rule is designed to ensure that  $\Delta_k$  never exceeds the maximum trust region size  $\Delta_{max}$ .

### 3 Dogleg

Dogleg minimizes an approximation of Eq. 1 by considering three possibilities using two points - the minimizer  $\delta x_u^{(k)}$  of  $M^{(k)}$  along the negative gradient direction  $-g^{(k)}$ , and the overall Newton's method minimizer  $\delta x_n^{(k)}$  of  $M^{(k)}$ . When the Hessian  $G^{(k)}$  is positive, the magnitude of  $\delta x_u^{(k)}$  is always less than that of  $\delta x_n^{(k)}$ , meaning that the Newton's method step is "more adventurous". How much we step towards the Newton's method point depends on the trust region size:

1. If the trust region is smaller than  $\delta x_u^{(k)}$ , we step in the negative gradient direction but only by the trust radius.
2. If the trust region boundary is between  $\delta x_u^{(k)}$  and  $\delta x_n^{(k)}$ , we step to the linearly-interpolated point between these two points that intersects the trust region boundary.
3. If the trust region boundary is larger than  $\delta x_n^{(k)}$ , we step to  $\delta x_n^{(k)}$ .

To find the intersection of the line between  $\delta x_u^{(k)}$  and  $\delta x_n^{(k)}$  with the trust region boundary, we solve a quadratic roots problem,

$$\begin{aligned} \Delta &= \|(1 - \tau) \delta x_u + \tau \delta x_n\| \\ \Delta^2 &= (1 - \tau)^2 \delta x_u^T \delta x_u + 2\tau(1 - \tau) \delta x_u^T \delta x_n + \tau^2 \delta x_n^T \delta x_n \\ 0 &= \delta x_u^T \delta x_u - 2\tau \delta x_u^T \delta x_u + \tau^2 \delta x_u^T \delta x_u + 2\tau \delta x_u^T \delta x_n - 2\tau^2 \delta x_u^T \delta x_n + \tau^2 \delta x_n^T \delta x_n - \Delta^2 \\ 0 &= (\delta x_u^T \delta x_u - 2\delta x_u^T \delta x_n + \delta x_n^T \delta x_n) \tau^2 + (2\delta x_u^T \delta x_n - 2\delta x_u^T \delta x_u) \tau - \Delta^2 + \delta x_u^T \delta x_u \\ \tau &= \frac{-(2\delta x_u^T \delta x_n - 2\delta x_u^T \delta x_u) \pm \sqrt{(2\delta x_u^T \delta x_n - 2\delta x_u^T \delta x_u)^2 - 4(\delta x_u^T \delta x_u - 2\delta x_u^T \delta x_n + \delta x_n^T \delta x_n)(\delta x_u^T \delta x_u - \Delta^2)}}{2(\delta x_u^T \delta x_u - \delta x_u^T \delta x_n + \delta x_n^T \delta x_n)} \end{aligned}$$

From this we take whichever possibility for  $\tau$  such that  $0 < \tau < 1$ .

To find the steepest-descent minimizer  $\delta x_u^{(k)}$ , we perform line search in the gradient direction on the approximate function  $M$ ,

$$\delta x_u^{(k)} = \frac{-g^{(k)\top} g^{(k)}}{g^{(k)\top} G^{(k)} g^{(k)}} g^{(k)} \quad (2)$$

Thus, mathematically, we can write the dogleg update  $\delta x_d^{(k)}$  as

$$\delta x_d^{(k)} = \begin{cases} -\frac{\Delta}{\|\delta x_u^{(k)}\|} \delta x_u^{(k)}, & \Delta < \|\delta x_u^{(k)}\| \\ (1 - \tau^{(k)}) \delta x_u^{(k)} + \tau^{(k)} \delta x_n^{(k)}, & \|\delta x_u^{(k)}\| < \Delta < \|\delta x_n^{(k)}\| \\ \delta x_n^{(k)}, & \|\delta x_n^{(k)}\| < \Delta \end{cases}$$

## 4 Working with $M$ as a Bayes' Net

When we have already eliminated a factor graph into a Bayes' Net, we have the same quadratic error function  $M^{(k)}(\delta x)$ , but it is in a different form:

$$M^{(k)}(\delta x) = \frac{1}{2} \|Rx - d\|^2,$$

where  $R$  is an upper-triangular matrix (stored as a set of sparse block Gaussian conditionals in GTSAM), and  $d$  is the r.h.s. vector. The gradient and Hessian of  $M$  are then

$$\begin{aligned} g^{(k)} &= R^\top (Rx - d) \\ G^{(k)} &= R^\top R \end{aligned}$$

In GTSAM, because the Bayes' Net is not dense, we evaluate Eq. 2 in an efficient way. Rewriting the denominator (leaving out the  $(k)$  superscript) as

$$g^\top G g = \sum_i (R_i g)^\top (R_i g),$$

where  $i$  indexes over the conditionals in the Bayes' Net (corresponding to blocks of rows of  $R$ ) exploits the sparse structure of the Bayes' Net, because it is easy to only include the variables involved in each  $i^{\text{th}}$  conditional when multiplying them by the corresponding elements of  $g$ .

## References

- [1] Raphael Hauser. Lecture notes on unconstrained optimization. [link](#), 2006.