Pros and Cons of truncated Gaussian EP in the context of Approximate Inference Control

Marc Toussaint, TU Berlin
mtoussai@cs.tu-berlin.de
October 29, 2009

Approximate Inference Control (AICO, [1]) is a method for solving Stochastic Optimal Control (SOC) problems. A core step in this method is to approximate the belief in each time step, that is, compute a distribution over the system state given the coupling to the previous state, the next state, and the current constraints or goals. Since constraints and goals may be arbitrarily non-linear (specified in arbitrary task spaces) exact inference is infeasible. The core motivation for introducing AICO was to leverage the variety of existing *approximate* inference methods to suggest new efficient approximations in the context of SOC. In [1] we used Gaussian message approximations which can be computed from local LQG approximations of the system dynamics and costs. We also mentioned the possibility to use Expectation Propagation and truncated Gaussians for inference under hard constraints and limits. With this abstract we discuss in more detail the approach, its promises and limitations, and report on experiments with AICO and truncated Gaussians for robotics problems.

Truncated Gaussians vs. collision potentials. Efficient motion optimization algorithms are either based on sampling (explicitly testing whether a configuration or state is feasible, RRTs, road maps, etc) or based on some optimization principle which assumes a smooth cost function (gradient-based, CHOMP, DDP/iLQG, etc). In the latter case, to deal with hard (non-continuous) constraints one has to introduce local approximations to the hard constraint. A typical example is a repelling squared potential: Given a list of object pairs with distances d_i below a margin m one may define a squared cost function $C_{\text{col}} = \alpha_{\text{col}} \sum_i (1 - d_i/m)^2$. Using the Jacobian J_i of pair distances one can pull this back to a squared cost function in joint space. Efficient optimal control methods like DDP or AICO can be used with this cost approximation. However, this approach implies that collision constraints are now in trade-off with other cost terms. Choosing the constant α_{col} becomes a hand-tuning problem. Also, fixing the margin parameter m is often undesired: during slow and controlled object manipulation the fingers may get very close to objects, during a fast large-scale reaching motion the margin should much higher.

Using truncated Gaussians seems to provide a principled solution to hard constraints – here are the pros of truncated Gaussians:

a) it directly corresponds to minimizing the probability of collision (maximizing the likelihood $P(\text{non-collision})$),

b) its effect on a belief is variance dependent,

c) it is parameter free.

To point a): In the probabilistic setting, avoiding collisions corresponds to introducing a binary random variable which indicates collisions and conditioning this variable to zero. This evidence results in a discontinuous message over the state space, which can be approximated by a heavy-side function $\theta(x)$ along a hyperplane (normal to the distance Jacobian J_i). Given a Gaussian belief $b'(x)$ in state space which accounts for all other information (fwd & bwd messages, squared costs) we use expectation propagation (EP) to compute a message which approximates the effect of the truncation: We compute the moments of the truncated belief $\theta(x)b'(x)$ which defines the Gaussian belief $b(x) \approx \theta(x)b'(x)$. The constraint evidence is finally represented as the Gaussian message $\mu(x) \propto b(x)/b'(x)$. To summarize, this procedure directly approximates the effect of conditioning on non-collision. It does not rely on inventing a heuristic cost function that indirectly might lead to avoiding collisions.

To point b): The effect of the truncation of the belief in state space very much depends on the variance of the belief. For instance, a high-variance belief around a configuration close to collision is truncated much more than a low-variance belief. This effect is particularly interesting seems desirable in the context of the full SOC framework.

To point c): There are no parameters implicit in this approach. In place of the margin parameter, the typical mean distance to collision depends on the variance of the belief. In place of the α_{col} parameter, the effect of the truncation depends on the likelihood of collision.

Examples. Figure 1 illustrates the effect of using truncated Gaussian EP in AICO for trajectory planning. Most interestingly: in the collision potential case we see that collisions have no effect when the belief mode (which is used to compute the local LQG approximation) is outside the proximity margin of $m = 0.1$, independent of the belief's variance. We also performed experiments on simulated 7DoF robotic configurations (reaching with

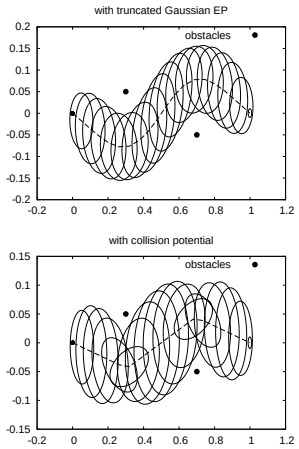

Figure 1: The result of AICO when using truncated Gaussian EP vs. squared collision potentials (margin $m = 0.1$) on a toy problem. Illustrated are the posterior beliefs of 20 time steps of a trajectory with random walk prior starting at $(0, 0)$ conditioned to reaching $(1, 0)$ and avoiding obstacles.

an arm), where the proximity distances were computed using a collision detection library (SWIFT++) and the truncating hyperplanes was transformed into joint space using the Jacobians of these distances. The additional computational cost of the truncation and EP are negligible in this specific setup. The resulting posterior trajectories successfully avoid collisions – typically by keeping rather large distances to the objects. We will report on these results on the poster.

Limitations in practice. However, when applying the method in such robotic scenarios we realized some limitations (which we will expand on more on the poster):

a) Every collision pair has to be handled by its own truncation: In the case of collision potentials, all (near) collisions are summarized by a single cost scalar C_{col}, thereby adding only a single task variable to the optimization problem. In the case of handling constraints with truncated Gaussians, every collision pair contributes a new constraint and thereby adds to the computational cost. In relevant scenarios (e.g., manipulation) there may be hundreds of such collision pairs which would lead to very high computational costs. We tried an alternative: Since we can in principle handle constraints in any task space we can impose a constraint on a single scalar that subsumes all collisions, e.g., C_{col}. However, we failed with this trick since the severe nonlinearity of this scalar prohibited convergence of EP and AICO. We were only successful with translating each near-collision to a constraint.

b) Typical collision detection thresholds lead to discontinuities in the algorithm that may prevent convergence of EP and AICO: Typically, in simulations one discards pairs of objects to be collision relevant when their distance is above a certain threshold ϵ. In effect, the collision library discontinuously switches between reporting and not reporting a critical distance. This can have negative effects in the case of truncated Gaussian: In our experiments, when we set ϵ to naturally small values, the discontinuous switching of active constraints prohibits EP and thereby AICO to converge. The method works nice, if we set ϵ rather large – but this increases inefficiency.

In conclusion, EP with truncated Gaussians in the context of motion control is theoretically very elegant and straight-forward in the context of AICO. We are not yet sure, however, if the practical limitations mentioned can be solved. For further details on computing moments of truncated Gaussians and corresponding software please see `http://user.cs.tu-berlin.de/~mtoussai/source-code/` .

[1] M. Toussaint. Robot trajectory optimization using approximate inference. In *Proc. of the 26rd Int. Conf. on Machine Learning (ICML 2009)*, 2009.

Technical Note: Computing moments of a truncated Gaussian for EP in high-dimensions

Marc Toussaint
Machine Learning & Robotics group, TU Berlin
Franklinstr. 28/29, FR 6-9, 10587 Berlin, Germany

October 16, 2009

In this technical note we derive an algorithm for computing the moments of a truncated Gaussian in high-dimensions. *In principle, all of this is well known and not novel.* Herbrich has already an (unpublished) technical note on EP with truncated Gaussians at `research.microsoft.com/pubs/74554/EP.pdf`. However, getting an *efficient* algorithm in high-dimension is not so trivial. We derive one in this note. The corresponding source code is available at `user.cs.tu-berlin.de/~mtoussai/source-code/`.
Our motivation is the application in the context of Aproximate Inference Control (**?**), where we use approximate inference to compute trajectories under hard constraints: Collision and joint avoidance implis messages of the form of heavyside functions; using Expectation Propagation with truncated Gaussians we can approximate the motion posterior.

1 1D case

Let us first address the simple 1D case. The problem is defined as follows: Let $x \in \mathbb{R}$ and $g(x) = e^{-x^2/2}$ and $\theta(x) = [[x \geq z]]$ (the heavyside function at z). We want to compute a Gaussian approximation of $g(x)\theta(x)$. For this we need to compute the moments of $g(x)\theta(x)$. For the norm (0th moment) we have:

$$\int_0^z e^{-t^2}\, dt = \frac{\sqrt{\pi}}{2}\, \text{erf}(z) \tag{1}$$

$$\begin{aligned} M_0 &:= \int_z^\infty e^{-x^2/2}\, dx = \sqrt{2}\int_{z/\sqrt{2}}^\infty e^{-t^2}\, dt \\ &= \sqrt{\pi/2}\, [1 - \text{erf}(z/\sqrt{2})] \end{aligned} \tag{2}$$

For the 1st moment:

$$\begin{aligned} M_1 &:= \int_z^\infty e^{-x^2/2}\, x\, dx = -\int_{-\frac{1}{2}z^2}^{-\infty} e^t\, dt \\ &= -\Big[e^t\Big]_{-z^2/2}^{-\infty} = -\Big[0 - e^{-z^2/2}\Big] = e^{-z^2/2} \end{aligned} \tag{3}$$

For the n-th moment:

$$I_n(z,a) := \int_z^\infty e^{-ax^2}\, x^n\, dx \tag{4}$$

$$\frac{\partial}{\partial a} I_n(z,a) = \int_z^\infty e^{-ax^2}\, (-x^2)\, x^n\, dx = -I_{n+2}(z,a) \tag{5}$$

$$\frac{\partial}{\partial a}\text{erf}(\sqrt{a}z) = \frac{a^{-1/2}z}{2}\, \frac{2}{\sqrt{\pi}}\, e^{-az^2} \tag{6}$$

And hence for 2nd moment:

$$I_2(z,a) = -\frac{\partial}{\partial a} N(z,a) \tag{7}$$

$$= a^{-3/2}\frac{\sqrt{\pi}}{4}\, [1 - \text{erf}(\sqrt{a}z)] + a^{-1/2}\, \frac{\sqrt{\pi}}{2}\, [\frac{a^{-1/2}z}{\sqrt{\pi}}\, e^{-az^2}] \tag{8}$$

$$= a^{-3/2}\frac{\sqrt{\pi}}{4}\, [1 - \text{erf}(\sqrt{a}z)] + \frac{z}{2a}\, e^{-az^2} \tag{9}$$

$$\begin{aligned} M_2 &:= \int_z^\infty e^{-x^2/2}\, x^2\, dx = I_2(z, \frac{1}{2}) \\ &= \sqrt{\pi/2}\, [1 - \text{erf}(z/\sqrt{2})] + z\, e^{-z^2/2} \end{aligned} \tag{10}$$

$$= M_0 + zM_1 \tag{11}$$

In summary, we have

$$\text{norm} \quad n := M_0 = \sqrt{\pi/2}\, [1 - \text{erf}(z/\sqrt{2})] \tag{12}$$

$$\text{mean} \quad m := M_1/M_0 = e^{-z^2/2}/n \tag{13}$$

$$\text{variance} \quad v := M_2/M_0 - m^2 = 1 + zm - m^2 \tag{14}$$

2 General case

We now have a n-dim Gaussian $f(y)$ and heavyside function $\theta(y)$ along a hyperplane with normal c and offset d,

$$f(y) \propto \exp\{-\frac{1}{2}(y-a)^\top A^{-1}\, (y-a)\} \tag{15}$$

$$\theta(y) = [[c^\top y - d \geq 0]] \tag{16}$$

where $[[\cdot]]$ is the indicator function. We transform this problem such that the Gaussian becomes a standard Gaussian and the constraint is aligned with the x-axis. We need two transformations for this: first a linear transform to standardize the Gaussian, then a rotation to align with the x-axis. Let $A = M^\top M$ be the Cholesky decomposition ($A^{-1} = M^{-1}M^{-\top}$) and we define $x = M^{-\top}(y-a)$. We have

$$f(x) = \exp\{-\frac{1}{2}x^\top x\} \tag{17}$$

Algorithm 1 Truncated Standard Gaussian

Input: z
Output: norm n, mean m, variance v
$n = \sqrt{\pi/2}[1 - \text{erf}(z\sqrt{2})]$
$m = \exp(-z^2/2)/n$
$v = 1 + zm - m^2$

Algorithm 2 Truncate Gaussian

Input: mean a, covariance A, constraint coeffs c, d
Output: mean b, covariance B
$M^T M = A$ *// Cholesky decomposition*
$z = (c^\top a + d)/|Mc|$
$v = Mc/|Mc|$
$R =$ rotation onto v *// as in equation (**??**)*
(m, v) = Truncated Standard Gaussian(z)
$b = M^\top R(m, 0, .., 0) + a$
$B = M^\top R\text{diag}(v, 1, .., 1)R^\top M$

$$\theta(x) = [[c^\top(M^\top x + a - d) \geq 0]] = [[v^\top x + z \geq 0]] \,, \tag{18}$$

$$v := Mc/|Mc| \,, \quad z := [c^\top(a - d)]/|Mc| \tag{19}$$

Note that we defined v to be normalized. (If $|Mc|$ is zero the truncation has no effect or zero likelihood, depending on whether $c^\top a - d > 0$ or $c^\top a - d < 0$, respectively.) We compute a rotation matrix that rotates the unit vector $\boldsymbol{e} = (1, 0, .., 0)$ onto v (implemented in array.cpp). We define $x' = R^{-1}x$. We have $v = R\boldsymbol{e}$ and

$$f(x') = \exp\{-\frac{1}{2}x'^\top x'\} \tag{20}$$

$$\begin{aligned}\theta(x') &= [[v^\top R x' + z \geq 0]] \\ &= [[(R^{-1}v)^\top x' + z \geq 0]] = [[x'_1 + z \geq 0]]\end{aligned} \tag{21}$$

That is, $\theta(x')$ truncates along the first axis in the x' coordinate system. Given the mean m and variance v of the $(-z)$-truncated standard Gaussian, we have

$$f(x')\,\theta(x') \approx \mathcal{N}(x'|b', B') \tag{22}$$

$$b' = (m, 0, .., 0) \tag{23}$$

$$B' = \text{diag}(v, 1, .., 1) \tag{24}$$

We undo the transformation $x' = R^{-1}M^{-\top}(y - a)$ and get the result

$$f(y)\,\theta(y) \approx \mathcal{N}(y|b, B) \tag{25}$$

$$b = M^\top R b' + a \tag{26}$$

$$B = M^\top R B' R^\top M \tag{27}$$

which gives the mean and covariance of the truncated Gaussian. The explicit algorithms are given below. The figure illustrates the result of truncating a Gaussian in 2D with the constraint $[[x > 1]]$.

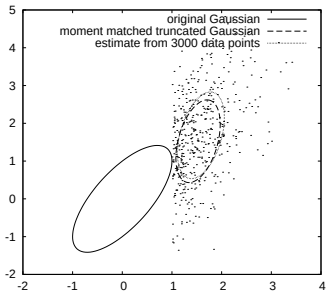

Figure 1: Truncation of a Gaussian at the constraint $[[x > 1]]$ in 2D.