DIFFUSION PARTICLE FILTERING ON THE SPECIAL ORTHOGONAL GROUP USING LIE ALGEBRA STATISTICS

Claudio J. Bordin Jr.[1], Caio G. de Figueredo[2], and Marcelo G. S. Bruno[3]
[1]claudio.bordin@ufabc.edu.br [2]caio.figueredo@ifce.edu.br [3]bruno@ita.br
[1]Universidade Federal do ABC, Brazil [2]Instituto Federal do Ceará, Brazil [3]Instituto Tecnológico de Aeronáutica, Brazil

1. Introduction

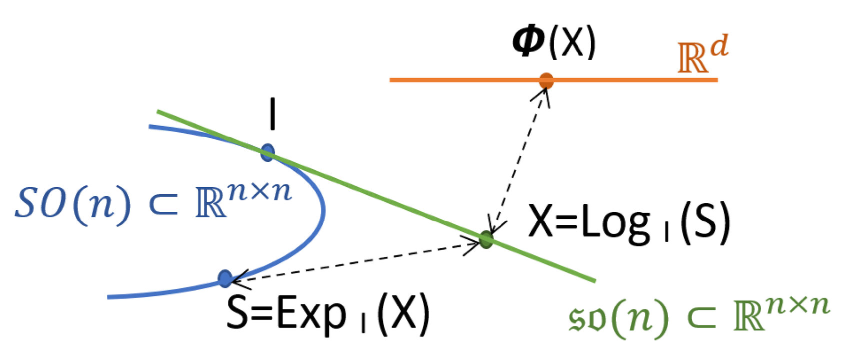
- Many applications in engineering require the estimation of variables with built-in nonlinear constraints, which leads them to be modeled as points on a manifold.
- Additionally it is often advantageous to distribute the computational burden of estimation methods among separate remote nodes.
- In previous works, we introduced diffusion particle filter (PF) algorithms for distributed estimation of variables constrained to the unit hypersphere and the Stiefel manifold.
- Random Exchange (RndEx) and the Adapt-then-Combine (ATC) diffusion techniques were used.
- Here, we consider a state tracking problem in which the state evolves as a random walk on the Special Orthogonal Group $SO(n)$.
- An element of $SO(3)$ can represent the rotational state of a rigid body.

2. Special Orthogonal Group $SO(n)$

- The Special Orthogonal Group is a matrix Lie group, closed with respect to matrix multiplications.
- $SO(n)$ is also a smooth manifold. The tangent space to point $\boldsymbol{M} \in SO(n)$, denoted $T_{\boldsymbol{M}}$, is the space of $n \times n$ real matrices $\boldsymbol{X}$ such that $\boldsymbol{M}^T\boldsymbol{X}$ is skew-symmetric.
- It is possible to move from a point $\boldsymbol{X} \in T_{\boldsymbol{M}_1}$ to another point in $T_{\boldsymbol{M}_2}$, with $\boldsymbol{M}_1, \boldsymbol{M}_2 \in SO(n)$, using the transport operator $\mathcal{T} : T_{\boldsymbol{M}_1} \to T_{\boldsymbol{M}_2}$ such that $\mathcal{T}(\boldsymbol{X}) = \boldsymbol{M}_2\boldsymbol{M}_1^{-1}\boldsymbol{X}$.
- It is also possible to map a point $\boldsymbol{S} \in SO(n)$ into a point $\boldsymbol{X} \in T_{\boldsymbol{M}}$ and vice-versa using the logarithmic and exponential maps, respectively:

$$\mathrm{Log}_{\boldsymbol{M}}(\boldsymbol{S}) = \boldsymbol{M}\,\mathrm{logm}(\boldsymbol{M}^T\boldsymbol{S}), \qquad \mathrm{Exp}_{\boldsymbol{M}}(\boldsymbol{X}) = \boldsymbol{M}\,\mathrm{expm}(\boldsymbol{M}^T\boldsymbol{X}),$$

where $\mathrm{logm}(\cdot)$ and $\mathrm{expm}(\cdot)$ are the matrix logarithm and matrix exponential functions.
- The tangent space $T_{\boldsymbol{I}}$ is by definition the Lie algebra $\mathfrak{so}(n)$ of the matrix Lie group $SO(n)$, the set of all real skew-symmetric matrices of dimension $n \times n$.



- An $n \times n$ skew-symmetric matrix has only $d \triangleq n(n-1)/2$ free-varying entries. Thus, we can define a bijective mapping $\Phi(\cdot)$ that associates each skew-symmetric matrix to a vector in $\mathbb{R}^d$. The inverse mapping is denoted by $\Phi^{-1}(\cdot)$.

3. Problem Setup

- Let the states $\{\mathbf{S}_k\} \in SO(n)$ be a sequence of random matrices that evolve according to the random walk

$$\mathbf{S}_k = \mathrm{Exp}_{\mathbf{S}_{k-1}}(\Phi^{-1}(\mathbf{v}_k)),$$

were k is the time index, $\{\mathbf{v}_k\}$ are i.i.d.Gaussian random vectors with zero mean and covariance matrix $\lambda^2\boldsymbol{I}$, and $\lambda \in \mathbb{R}$ is a hyperparameter.
- Multiple nodes on a sensor network record the observations $\{\mathbf{Y}_{k,r}\} \in \mathbb{R}^{n\times n}$, such that

$$p(\boldsymbol{Y}_{k,r}|\boldsymbol{S}_k) = \mathcal{N}_{\mathbb{R}^{n\times n}}(\boldsymbol{Y}_{k,r}|\boldsymbol{\mathcal{H}}_{k,r}(\boldsymbol{S}_k), \boldsymbol{\Omega}_r, \boldsymbol{\Gamma}_r),$$

where $r \in \{1, ..., R\}$ denotes the r-th node in the network, $\boldsymbol{\mathcal{H}}_{k,r} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ is a (possibly nonlinear) function, and $\mathcal{N}_{\mathbb{R}^{n\times n}}$ is the matrix normal p.d.f. in $\mathbb{R}^{n\times n}$.
- Given a realization $\{\boldsymbol{Y}_{j,r}\}$, $1 \le j \le k$, $1 \le r \le R$, of the observations $\{\mathbf{Y}_{j,r}\}$, we want to recursively estimate $\mathbf{S}_k$ distributedly.

4. ATC Diffusion Particle Filter

- Assume that node r has a posterior p.d.f. $p_{k-1|k-1,r}(\boldsymbol{S}_{k-1})$ conditioned on all network measurements assimilated by node r up to instant $k-1$.
- The ATC diffusion filtering method is divided into two steps: Adapt Step and Combine Step.

Adapt Step: update $p_{k-1|k-1,r}$ to $\tilde{p}_{k|k,r}$, the posterior p.d.f., assimilating the measurements $\{Y_{k,u}\}$ available at all nodes u in $N(r)$, the neighborhood of node r.

- Assuming that the observation vectors $\mathbf{Y}_{k,u}$ are conditionally independent from node to node given the state

$$\tilde{p}_{k|k,r}(\boldsymbol{S}_k) \propto \left[\prod_{u\in N(r)} p(\boldsymbol{Y}_{k,u}|\boldsymbol{S}_k)\right] \times \int_{\boldsymbol{S}_k \in SO(n)} p(\boldsymbol{S}_k|\boldsymbol{S}_{k-1})p_{k-1|k-1,r}(\boldsymbol{S}_{k-1})\, d(SO(n)).$$

- If $p_{k-1|k-1,r}(\boldsymbol{S}_{k-1})$ is represented by a weighted particle set $\{w^{(q)}_{k-1,r}, \boldsymbol{S}^{(q)}_{k-1,r}\}$, $q \in \{1, ..., Q\}$, we can build a new weighted representation $\{\tilde{w}^{(q)}_{k,r}, \tilde{\boldsymbol{S}}^{(q)}_{k,r}\}$ for $\tilde{p}_{k|k,r}(\boldsymbol{S}_k)$ using a marginal particle filter:

1) Sample $\tilde{\mathbf{S}}^{(q)}_{k,r} \sim \sum_{q'=1}^{Q} w^{(q')}_{k-1,r}\, p(\boldsymbol{S}_k|\boldsymbol{S}^{(q')}_{k-1,r})$.
2) Evaluate the weights $\tilde{w}^{(q)}_{k,r} \propto \prod_{u\in N(r)} p(\boldsymbol{Y}_{k,u}|\tilde{\boldsymbol{S}}^{(q)}_{k,r})$.

- The optimal local state estimate $\tilde{\boldsymbol{S}}_{k|k,r}$ prior to sensor fusion at node r is the intrinsic mean $\boldsymbol{S} \in SO(n)$ that minimizes $\mathbb{E}_{\tilde{p}_{k|k,r}}[d^2_G(\boldsymbol{S}, \mathbf{S}_k)]$.
- The particle filter approximates the intrinsic mean by the Karcher mean of the weighted particle set $\{\tilde{w}^{(q)}_{k,r}, \tilde{\boldsymbol{S}}^{(q)}_{k,r}\}$ on $SO(n)$, i.e., $\tilde{\boldsymbol{S}}_{k|k,r} = \arg\min_{\mathbf{S}\in\mathcal{G}} \sum_{q=1}^{Q} \tilde{w}^{(q)}_{k,r} d^2_G(\boldsymbol{S}, \tilde{\boldsymbol{S}}^{(q)}_{k,r})$, and transmits it to nodes $u \in \{N(r) \setminus \{r\}\}$.

Combine Step: fuse the local state estimates $\tilde{\boldsymbol{S}}_{k|k,u}$, $u \in N(r)$ to obtain a new merged estimate at node r.

- Upon receiving the local state estimates $\tilde{\boldsymbol{S}}_{k|k,u}$ from each node $u \in \{N(r) \setminus \{r\}\}$, node r first computes the Karcher mean

$$\bar{\boldsymbol{S}}_{k|k,r} = \arg\min_{\boldsymbol{S}\in\mathcal{G}} \sum_{u\in N(r)} a_{r,u}\, d^2_G(\boldsymbol{S}, \tilde{\boldsymbol{S}}_{k|k,u}),$$

where $\{a_{r,u}\}$ is a set of positive real weights such that $\sum_u a_{r,u} = 1$ for all r.
- Next, node r executes the following steps:

1) Compute $\boldsymbol{X}^{(q)}_{k,r} = \mathrm{Log}_{\bar{\boldsymbol{S}}_{k|k,r}}(\tilde{\boldsymbol{S}}^{(q)}_{k,r}) \in T_{\bar{\boldsymbol{S}}_{k|k,r}}$, evaluate the skew-symmetric matrix $\boldsymbol{Z}^{(q)}_{k,r} = \bar{\boldsymbol{S}}^T_{k|k,r}\boldsymbol{X}^{(q)}_{k,r} \in \mathfrak{so}(n)$, and determine $\boldsymbol{z}^{(q)}_{k,r} = \Phi(\boldsymbol{Z}^{(q)}_{k,r}) \in \mathbb{R}^d$.
2) Fit a d-variate Gaussian approximation $\pi_{k|k,r}$ to the weighted particle set $\{\tilde{w}^{(q)}_{k,r}, \boldsymbol{z}^{(q)}_{k,r}\}$, $q \in \{1, \dots, Q\}$, computing the corresponding sample mean $\boldsymbol{m}_{k|k,r}$ and sample covariance matrix $\boldsymbol{P}_{k|k,r}$.
3) Transmit $\boldsymbol{m}_{k|k,r}$ and $\boldsymbol{P}_{k|k,r}$ to nodes $u \in \{N(r) \setminus \{r\}\}$.
4) Receive $\boldsymbol{m}_{k|k,u}$ and $\boldsymbol{P}_{k|k,u}$ from nodes $u \in \{N(r) \setminus \{r\}\}$.
5) Combine the local p.d.f.'s $\pi_{k|k,u}$, $u \in N(r)$, into a fused p.d.f. $\breve{\pi}_{k|k,r}$ using the Geometric Average fusion rule

$$\breve{\pi}_{k|k,r}(\boldsymbol{z}_k) \propto \prod_{u\in N(r)} [\pi_{k|k,u}(\boldsymbol{z}_k)]^{a_{r,u}},$$

which minimizes $\sum_u a_{r,u} D_{KL}(\breve{\pi}\|\pi_{k|k,u})$, where D_{KL} is the Kullback-Leibler divergence.
6) Resample $\breve{\boldsymbol{z}}^{(q)}_{k,r} \sim \breve{\pi}_{k|k,r}(\boldsymbol{z}_k) \in \mathbb{R}^d$, evaluate $\breve{\boldsymbol{X}}^{(q)}_{k,r} = \bar{\boldsymbol{S}}_{k|k,r}\breve{\boldsymbol{Z}}^{(q)}_{k,r} \triangleq \bar{\boldsymbol{S}}_{k|k,r}\Phi^{-1}(\breve{\boldsymbol{z}}^{(q)}_{k,r}) \in T_{\bar{\boldsymbol{S}}_{k|k,r}}$, then recover $\boldsymbol{S}^{(q)}_{k,r} = \mathrm{Exp}_{\bar{\boldsymbol{S}}_{k|k,r}}(\breve{\boldsymbol{X}}^{(q)}_{k,r}) \in SO(n)$.
7) Reset the particle weights $w^{(q)}_{k,r} = 1/Q$, $q \in \{1, \dots, Q\}$.
8) Compute the fused state estimate $\hat{\boldsymbol{S}}_{k|k,r}$ at node r as the Karcher mean of $\{w^{(q)}_{k,r}, \boldsymbol{S}^{(q)}_{k,r}\}$, which represents the final posterior $p_{k|k,r}$ propagated to the next time step.

5. RndEx Diffusion PF

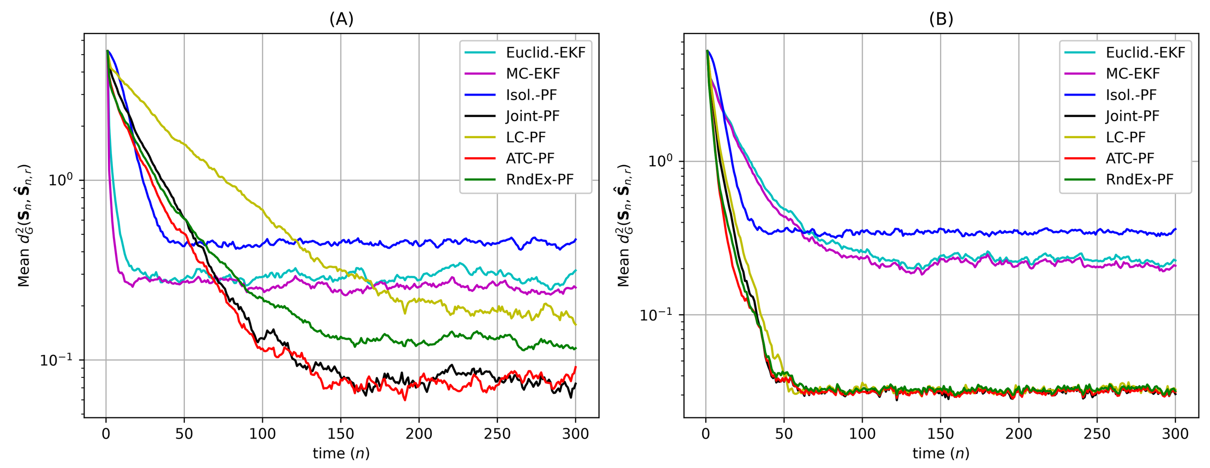
- The RndEx Algorithm has two steps: Random Exchange and Data Assimilation.
- In the Random Exchange step, a node l exchanges with another randomly chosen node r a compressed representation of its posterior p.d.f. $p(\boldsymbol{S}_{k-1}|\bar{\boldsymbol{Y}}_{1:k-1,l})$, in which $\bar{\boldsymbol{Y}}_{1:k-1,l}$ denotes all observations assimilated up to instant k.
- At the end of the Random Exchange Step, node r receives the compressed representation and rebuilds the particle set.
- In the Data Assimilation step, node r samples new particles and updates the particles' weights as

$$\boldsymbol{S}^{(q)}_{k,r} \sim p(\boldsymbol{S}_k|\boldsymbol{S}^{(q)}_{k-1,l}), \quad w^{(q)}_{k,r} \propto w^{(q)}_{k-1,l}\left[\prod_{u\in N(r)} p(\boldsymbol{Y}_{k,u}|\boldsymbol{S}^{(q)}_{k,r})\right].$$

- The updated set $\{w^{(q)}_{k,r}, \boldsymbol{S}^{(q)}_{k,r}\}$ is then a Monte Carlo representation of the posterior p.d.f. $p(\boldsymbol{S}_k|\bar{\boldsymbol{Y}}_{k,r}, \bar{\boldsymbol{Y}}_{1:k-1,l}) \triangleq p(\boldsymbol{S}_k|\bar{\boldsymbol{Y}}_{1:k,r})$.

6. Simulation Results

- We ran Monte Carlo simulations with 1,000 independent runs.
- The network has five nodes: nodes 1 to 4 are on the vertices of a square and node 5 is at its center and is connected to all other nodes.
- The noise covariance matrices were set to $\boldsymbol{\Omega}_r = \boldsymbol{I}$ and $\boldsymbol{\Gamma}_r = \boldsymbol{I} \cdot 10^{-\alpha_r/10}$, with α_r equal to 3, 6, 10, 13 and 20 dB for $r = 1, \dots, 5$, respectively.
- The weights $\{a_{r,u}\}$ are determined by the Metropolis rule. We assumed that $n = 3$, $Q = 200$, and $\lambda = 0.15$.
- $\boldsymbol{\mathcal{H}}_{k,r}$ was defined as $[\boldsymbol{\mathcal{H}}_{k,r}(\boldsymbol{S}_k)]_{i,j} = h\left([\boldsymbol{S}_k]_{i,j}\right)$ and $h(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ is scalar.
- We employed two formulations for $h(\cdot)$, namely, A) $h(x) = x^3 - 1/2$, and B) $h(x) = \mathrm{sat}(x; 1/2)$, where $\mathrm{sat}(x;\beta) = \begin{cases} x, & |x| < \beta, \\ \beta \cdot x/|x|, & |x| \ge \beta. \end{cases}$
- For comparison, we ran in the same setup i) a centralized EKF (Euclid.-EKF), ii) a centralized manifold-constrained EKF (MC-EKF), iii) Particle filters (Isol-PF) that operate in isolation at each node, iv) a centralized particle filter (Joint-PF), and v) a modified likelihood consensus distributed PF (LC-PF).



7. Conclusions

- We introduced new distributed diffusion PFs to track a sequence of matrices that evolve on the Special Orthogonal Group.
- The algorithms employ a Gaussian approximation of the weighted particle set defined in an isomorphism of the Lie algebra $\mathfrak{so}(n)$.
- Simulation results show that the performance of the proposed algorithms surpasses or equals that of an alternative distributed PF at a lower communication cost, and outperform a manifold-constrained centralized EKF.