A Distributed Recursive Maximum Likelihood Implementation for Sensor Registration

Nikolas Kantas Sumeetpal S. Singh
Signal Processing Group,
Department of Engineering
University of Cambridge, UK
{nk234,sss40}@cam.ac.uk

Arnaud Doucet
Department of Computer Science,
Department of Statistics
University of British Columbia, CA
arnaud@cs.ubc.ca

Abstract - Recursive Maximum Likelihood (RML) is a popular methodology for estimating unknown static parameters in state-space models. We describe how a completely decentralized version of RML can be implemented in dynamic graphical models through the propagation of suitable messages that are exchanged between neighbouring nodes of the graph. The resulting algorithm can be interpreted as a generalization of the celebrated belief propagation algorithm to compute likelihood gradients. This algorithm is applied to solve the sensor registration and localisation problem for sensor networks. An exact implementation is given for dynamic linear Gaussian models without loop. If loops are present, a loopy version of the algorithm is described. For non-linear non Gaussian scenarios, a Sequential Monte Carlo (SMC) or particle filter implementation is sketched.

Keywords: decentralised sensor networks, distributed parameter estimation, distributed sensor registration, dynamic graphical models.

1 Introduction

The standard approach in centralised sensor management is to transmit measurements from all sensor nodes to a fusion node which performs most of the computation [2]. This limits the number of expensive sensors used [5]. However, the spatial deployment of the network is limited to a restricted range relative to the central fusion node. Moreover, the (large) communication bandwidth requirements also limit the maximum number of potentially utilised sensors [1]. This motivates the adoption of decentralised or distributed approaches, such as ad-hoc sensor networks. In such networks neighbouring sensors-trackers exchange information between them in order to track optimally the target as in the centralised approach [3].

In such a context, it is important for all sensors to "learn" some parameters of its neighbouring nodes so as to be able to use their measurements for tracking. The network should also be able to adapt to potentially changing parameters and/or network topology. In particular, sensors' measurements are known to exhibit errors, which in some cases can be decomposed in random noise and systematic biases. The problem of identifying these biases is known as sensor registration [8, 9, 14]. In a centralised approach this can be solved at the fusion center by augmenting the state with this bias and maximising the resulting measurement likelihood. In a distributed context, the problem is somewhat more complex and each node needs to estimate/register the biases of its neighbours. We want to achieve this using only local messages between a node and its neighbours. Solving this problem is essential in many real-world systems as an heterogeneous collection of sensors is often used and their associated sensor biases can differ significantly [9].

Furthermore in many distributed sensor networks it is also important for each node to "learn" the position or coordinates of its neighbours relative to itself. This is known as sensor self-localisation or self-calibration [1]; this problem also often appears in the computer vision literature [6]. For example, let a sensor network consist of a collection of nodes: each node is a tracker and is at the origin of its own coordinate system. In a target tracking scenario it is important for each sensor-tracker node to be able to translate the relative frame of reference of its neighbour to its own in order to utilise and fuse the measurements. This is essentially equivalent to knowing with precision the coordinates of its neighbours w.r.t. itself. If these coordinates are biased or unknown, the sensor calibration problem can be put in the same framework as the sensor registration one.

In this article, we show that these problems -sensor registration and self-localisation for sensor networks- can be cast as recursive static parameter estimation for dynamic graphical models. We shall be adopting here a RML approach to solve this problem where the (average) log-likelihood of the unknown parameters is maximized on-line using a stochastic gradient approach. Belief propagation ideas have been widely used to perform statistical inference in undirected and directed graphs using message passing [10]. The novelty of our paper relies on a fully decentralized calculation of the log-likelihood gradient on graphs. In this respect, it can be interpreted as a generalization of belief propagation to directed graphs. Received messages at a node shall contain the sufficient statistics sent from the rest

of the network, in order for that node to infer the optimal parameters of its surrounding neighbourhood.

The rest of this paper is organized as: in Section 2 we present the recursive maximum likelihood (RML) parameter estimation approach [7]. In Section 3 we formulate precisely the sensor registration and self-localisation problems. In Sections 4 and 5 we solve these problems using an original distributed RML implementation. Finally, in Sections 6 and 7 we illustrate our ideas on an example and make some conclusive remarks.

2 RML Problem Formulation

Let $\{X_n\}_{n\geq 0}$ and $\{Y_n\}_{n\geq 1}$ be $\mathcal{X}$ and $\mathcal{Y}$-valued stochastic processes defined on a measurable space $(\Omega, \mathcal{F})$. For each $\theta \in \Theta$ where Θ is an open subset of $\mathbb{R}^k$, the process $\{X_n\}_{n\geq 0}$ is an unobserved (hidden) Markov process of initial density π_0 and Markov transition density $f_\theta(x'|\,x)$, i.e. $X_0 \sim \pi_0$ and

$$X_{n+1}|\,X_n = x \sim f_\theta(\cdot|\,x)\,. \tag{1}$$

The hidden process $\{X_n\}_{n\geq 0}$ is observed through the process $\{Y_n\}_{n\geq 1}$. It is assumed that the observations conditioned upon $\{X_n\}_{n\geq 0}$ are independent with marginal density $g_\theta(y|\,x)$, i.e.,

$$Y_n|\,X_n = x \sim g_\theta(\cdot|\,x)\,. \tag{2}$$

It is assumed that the true parameter θ^* governing the evolution of the processes $\{X_n\}_{n\geq 0}$ and $\{Y_n\}_{n\geq 1}$ is not known and it to be estimated. For a given sequence of T observations, the batch ML estimate of θ^* is the solution to the maximization of the log likelihood function,

$$\begin{aligned} L_\theta(Y_{1:T}) &= \log P_\theta(Y_1, \ldots, Y_T) \\ &= \sum_{n=1}^{T} \log P_\theta(Y_n|\,Y_1, \ldots, Y_{n-1}) \end{aligned}$$

where for $n = 1$ the conditional density is understood to be $P_\theta(Y_1)$.

Our aim is to solve θ^* recursively using the sequence of observations $\{Y_n\}_{n\geq 1}$. Under regularity assumptions including the stationarity of the state space model, then

$$\begin{aligned} L(\theta) &= \lim_{T\to\infty} \frac{1}{T} L_\theta(Y_{1:T}) \\ &= \int_{\mathcal{Y}\times\mathcal{P}(\mathcal{X})} \log\left(\int g_\theta(y|x)\mu(x)dx\right) \lambda_{\theta,\theta^*}(dy, d\mu) \end{aligned}$$

where $\mathcal{P}(\mathcal{X})$ is the space of probability distributions on $\mathcal{X}$, and $\lambda_{\theta,\theta^*}(dy, d\mu)$ is the joint invariant distribution of the measurement and the prediction density, $(Y_n, p_\theta(x_n|Y_{1:n-1}))$.

Recursive Maximum Likelihood (RML) consists of identifying the static parameter $\theta^* = \arg\max L(\theta)$ by using a stochastic gradient algorithm where at time n the parameter estimate θ_n is given by

$$\begin{aligned} \theta_n &= \theta_{n-1} + \gamma_n \nabla \log(P(Y_n|Y_{1:n-1})) \\ &= \theta_{n-1} + \gamma_n \nabla \log(\int g_{\theta_{n-1}}(Y_n|x_n) \\ &\qquad \times p_{\theta_{1:n-1}}(x_n|Y_{1:n-1})dx_n), \end{aligned} \tag{3}$$

where $\{\gamma_n\}$ is a sequence of step-sizes such that $\sum_n \gamma_n = \infty$ and $\sum_n {\gamma_n}^2 < \infty$.

In the context of sensor networks, we will use the specific (graph) structure of the model (1)-(2) to perform those calculations in a fully decentralized way.

3 Sensor Registration and Localisation

3.1 General Framework for Sensor Registration

We consider a sensor network deployed for the tracking of targets. Let the set of nodes of the network be indexed by the finite set $\mathcal{V}$ while the connectivity of the network is specified by the set of edges $\mathcal{E}$. We will deal with an undirected graphical model $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Thus, two nodes i and j communicate provided the edge $e = (i, j)$ (or (j, i)) belongs to $\mathcal{E}$.

The state of a node v is a random variable X_v that represents the state of the target being tracked. The state of the target X_v would comprise of the target's location and velocities as measured with respect the local coordinate frame of node v. At time n let the state of the target at node v be $X_{v,n}$. The only assumption on the target dynamic model is

$$X_{v,n+1}|\,X_{v,n} = x_v \sim f_v(\cdot|\,x_v)$$

A local measurement $Y_{v,n}$ of the target's state is generated based on the sensing capabilities of the node. Let the measurement be generated according to the probability density function

$$Y_{v,n}|\,X_{v,n} = x_v \sim g_{\vartheta_v^*}(\cdot|\,x_v)$$

where ϑ_v^* is the unknown bias of sensor node v.

Since all nodes are engaged in tracking the same target, it is possible to utilise the measurement of all nodes to enhance the tracking performance. For instance, for a completely decoupled implementation, each node will maintain its local filtering distribution

$$\pi_{v,n}(x_v)dx_v = P_{\vartheta_v^*}(X_{v,n} \in dx_v|\,Y_{v,1}, \ldots, Y_{v,n}).$$

In a *fully coupled* implementation, each node will use the measurements of all other nodes too in its update of the Bayes recursion, i.e. it propagates the filtering distribution

$$\pi_{v,n}(x_v)dx_v = P_{\vartheta^*}(X_{v,n} \in dx_v|\,Y_1, \ldots, Y_n)$$

where Y_n denotes the vector of stacked observations $[Y_{v,n}]_{v\in\mathcal{V}}$ and ϑ^* denotes the vector of stacked biases $[\vartheta_v^*]_{v\in\mathcal{V}}$. Similarly the prediction distribution will be

$$\pi_{v,n|n-1}(x_v)dx_v = P_{\vartheta^*}(X_{v,n} \in dx_v|\,Y_1, \ldots, Y_{n-1})$$

It is clear that a fully coupled implementation is advantageous since nodes whose observations are poor, due to distance or sensing capabilities, can benefit from other sensors with better quality observations.

The prediction step can be directly done in a decentralized way at each node. However in the update step, each node receives an observation and all nodes in the network have to share their information by passing messages around the network in order to compute their filtering distribution.

3.2 Sensor Localisation as a General Sensor Registration Problem

We will ignore here the sensor biases and concentrate on the sensor localisation problem formulated in a general sensor registration context as discussed earlier. For sake of clarity, we make the following assumption which we stress **is not** essential for the framework we propose.

Assumption 1 *All nodes maintain a 2D-cartesian coordinate system and maintain as the state of the target its position and velocity in the relevant directions. Also, each node regards itself as located at the origin of its own coordinate system.*

We refer to a particular node v and would like to use the measurements of the rest of the nodes to compute $\{\pi_{v,n}\}$. Obviously the measurement likelihood depends on the coordinate of each node w.r.t. node v. We denote then $\theta^*_{i\to j}$ the coordinate transformation from node i to j, i.e. $\theta^*_{i\to j}$ is the origin of node i in the coordinate frame of node j. We use then $\theta^* = [\theta^*_{i\to j}]_{(i,j)\in\mathcal{E}}$ as a static registration parameter instead of a measurement bias and we have $\theta^*_{i\to i} = 0$ because of Assumption 1.

We denote the observation likelihood of node v by $g_v(\cdot|\, x_v)$. The Bayes recursion applied independently to each node i would yield

$$\pi_{i,n+1}(x_{i,n+1}) \propto g_i(Y_{i,n+1}|\, x_{i,n+1}) \int f_i(x_{i,n+1}|x_{i,n})\pi_{i,n}(x_{i,n})dx_{i,n}.$$

In a coupled implementation of two nodes, node i incorporates the observation of adjacent node j (connected by an edge),

$$\pi_{i,n+1}(x_{i,n+1}) \propto g_i(Y_{i,n+1}|\, x_{i,n+1})\quad g_j(Y_{j,n+1}|\theta^*_{i\to j} + x_{i,n+1}) \times \int f_i(x_{i,n+1}|x_{i,n})\pi_{i,n}(x_{i,n})dx_{i,n}.$$

It is clear that sharing the observations is only possible with the coordinate transformation variable $\theta^*_{i\to j}$. In this paper, we will use the observations from the entire network to update each node's filtering density.

Assumption 2 *All nodes have a* consistent *transition density for the target being tracked and let this density for node i be denoted by $f_i(x'_i|\, x_i)$. By consistent we mean that for any two nodes i and j,*

$$f_i(x'_i|\, x_i) = f_j(x'_i + \theta^*_{i\to j}|\, x_i + \theta^*_{i\to j}).$$

This assumption is necessary for the problem to be formulated without any ambiguity when translating the coordinate system from one node to another. All nodes should maintain the same transition density or prediction model with respect to an arbitrary coordinate scheme. Moreover the transition density should in turn follow the physical coordinate transformation of the state from one node to another. This is important since we want all nodes to share the same filtering and prediction densities. Using Assumption 2 we can also show that the prediction and filtering density at each node can be propagated consistently:

Property 3 *Assume $\pi_{i,n}$ and $\pi_{j,n}$ satisfy*

$$\pi_{i,n}(x_i) = \pi_{j,n}(x_i + \theta^*_{i\to j})$$

for all x_i. (This implies $\pi_{i,n}(x_j + \theta^*_{j\to i}) = \pi_{j,n}(x_j)$ for all x_j.) Then one also has $\pi_{i,n+1|n}(x_i) = \pi_{j,n+1|n}\left(x_i + \theta^*_{i\to j}\right)$.

In the fully coupled implementation, each node i will incorporate the observations of all other nodes in the network too. In the filtering step implemented by a node i, after all nodes take a measurement, we have

$$\pi_{i,n+1}(x_i) \propto \prod_{v\in\mathcal{V}\setminus i} g_v(Y_{v,n+1}|\, x_i + \theta^*_{i\to v}) \times g_i(Y_{i,n+1}|\, x_i)\pi_{i,n+1|n}(x_i)$$

where $\theta^*_{i\to v}$ for non adjacent nodes is defined as follows: for any path that connects nodes i and v then

$$\theta^*_{i\to v} = \theta^*_{i\to j_1} + \theta^*_{j_1\to j_2} + \ldots + \theta^*_{j_{n-1}\to j_n} + \theta^*_{j_n\to v}. \quad (4)$$

Since we start with the same prior and all nodes have the same transition densities, it is obvious that the filtering density shared in the network will be the same for all nodes on it. The coordinate transformations are consistent at each filtering step and hence can be used when one node needs to pass a message to another.

Property 4 *After the update step, for any two pair of nodes i and j, it follows that*

$$\pi_{i,n+1}(x_i) = \pi_{j,n+1}(x_i + \theta^*_{i\to j})$$

for all x_i.

Note: For the localisation problem the coordinate transformation function $h_{\theta^*_{i\to j}}$ of the state from node i to node j is given simply given by $h_{\theta^*_{i\to j}} = x_i + \theta^*_{i\to j}$. In the general sensor registration case $h_{\theta^*_{i\to j}}$ can admit any nonlinear form but given Assumption 2 is true the results in this and later sections still hold.

4 Learning $\theta^* = \left[\theta^*_{i\to j}\right]_{i,j}$ by Distributed RML

We propose a distributed implementation of RML for estimating all the coordinate transformations $\theta^*_{i\to j}$. Since there is one parameter per edge, namely $\theta_{i\to j}$

for the edge $(i,j) \in \mathcal{E}$, a particular node, say i, will take ownership of the parameter and recursively update it as observations are received in the network. This node shall be referred as root node for that edge, since it shall be the node to collect messages from the rest of the network in order to iterate $\theta_{i\to j,n}$ as in (3). The messages received by each node i from its neighbours should be sufficient to iterate the parameter $\theta_{i\to j,n}$ and perform the update step of the Bayesian recursion yielding the filtering density $\pi_{i,n}$. Since the prediction step can be performed locally at each node, we shall be able to estimate $\theta^*_{i\to j}$ while propagating $\pi_{i,n}$.

We now formulate the RML problem for node 1 as the reference node. Note this is an arbitrary choice since any node in the network can become root node. For a fixed parameter $\theta = [\theta_{i\to j}]_{(i,j)\in\mathcal{E}}$, the recursive log likelihood $\log P_\theta(Y_n|Y_{1:n-1})$, where Y_n denotes the vector of stacked observations $[Y_{v,n}]_{v\in\mathcal{V}}$, is

$$J_1(Y_n;\theta) = \log \int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1$$

where $\theta_{1\to 1} = 0$, while superscript θ on $\pi^\theta_{1,n|n-1}$ emphasizes the dependency on the vector of coordinate transformation. We will now take the gradient of this quantity with respect to $\theta_{1\to 2}$, i.e., we are assuming that node $(1,2)$ is a valid edge and that node 1 has ownership of parameter $\theta_{1\to 2}$, i.e. node 1 is the root node in this case,

$$\begin{aligned}
&\nabla_{\theta_{1\to 2}} J_1(Y_n;\theta) = \\
&= \nabla_{\theta_{1\to 2}} \log \int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1 \\
&\{\int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1\}^{-1} \\
&\times\{\int[\sum_{v\in\mathcal{V}} \tfrac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}] \\
&\times \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1 \\
&+\int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1)dx_1\}
\end{aligned} \tag{5}$$

In the context of recursive distributed parameter estimation $\pi^\theta_{1,n|n-1}(x_1)$ and its gradient $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1)$ should be propagated locally at node 1. It also appears necessary to pass $\prod_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\sum_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}$ using appropriate messages from the network to node 1 in order to be able to update $\theta_{i\to j,n}$ as in (3). It can be shown that these messages are sufficient to propagate the filtering and prediction densities as well as their gradients.

4.1 Propagating the filtering and prediction densities and its derivatives

For node 1 we would like to implement a recursion for $\pi^\theta_{1,n}(x_1)$ and $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n}(x_1)$ given that $\pi^\theta_{1,n-1}(x'_1)$ and $\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n-1}(x'_1)$ are available locally from previous epoch $n-1$. As node 1 is chosen to be the reference or root node just for convenience and is a completely arbitrary choice, the derivation for any other node is identical. Consider the prediction and update stage of the filter given the evolution density of the target's model f_1 and the observation density g_1:

$$\pi^\theta_{1,n|n-1}(x_1) = \int f_1(x_1|x'_1)\pi^\theta_{1,n-1}(x'_1)dx'_1, \tag{6}$$

and

$$\pi^\theta_{1,n}(x_1) = \frac{\prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)}{\int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1} \tag{7}$$

In the meantime we aim to propagate the densities and their derivatives. For the derivatives we have:

$$\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1) = \int f_1(x_1|x'_1)\nabla_{\theta_{1\to 2}}\pi^\theta(x'_1)dx'_1, \tag{8}$$

and

$$\begin{aligned}
&\nabla_\theta \pi^\theta_{1,n}(x_1) = \{\int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1\}^{-1} \\
&\times\{\prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1) \\
&+(\tfrac{\sum_{v\in\mathcal{V}} \nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}) \\
&\times \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1) \\
&-\pi^\theta_{1,n}(x_1)[\int \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\nabla_{\theta_{1\to 2}}\pi^\theta_{1,n|n-1}(x_1)dx_1 \\
&+\int(\tfrac{\sum_{v\in\mathcal{V}} \nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}) \\
&\times \prod_{v\in\mathcal{V}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})\pi^\theta_{1,n|n-1}(x_1)dx_1]\}.
\end{aligned} \tag{9}$$

Note that if $\prod_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\sum_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}$ are available at the root node 1, then the filtering and predictions densities together with their derivatives can be computed locally. These quantities should become available to node 1 by messages received from its neighbours, which in turn receive messages from their neighbours. We aim to define an appropriate message passing scheme so that all possible nodes can act as roots and update one or more parameters associated with its adjacent edges.

4.2 Defining Message Passing in the Sensor Network

To derive a distributed implementation, consider first how for each $v \in \mathcal{V}$ $g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ can be communicated to node 1 via a sequence of messages. Assume the directed path from node v to 1 traverses the edges $\overrightarrow{(v,j_l)}$, $\overrightarrow{(j_l,j_{l-1})}$, $\ldots$, $\overrightarrow{(j_2,j_1)}$, $\overrightarrow{(j_1,1)}$.

In order to pass $g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ from node v to 1, the incoming message to node j_k from node j_{k+1}

should be $g_v(Y_{v,n}|\, x_{j_k}+\theta_{j_k\to v})$. Then node j_k should forward to node j_{k-1} the message

$$g_v(Y_{v,n}|\, x_{j_k}+\theta_{j_k\to v})|_{x_{j_k}=x_{j_{k-1}}+\theta_{j_{k-1}\to j_k}} = g_v(Y_{v,n}|\, x_{j_{k-1}}+\theta_{j_{k-1}\to v})$$

So starting from $g_v(Y_{v,n}|\, x_v)$ at node v, after using repetitive coordinate transformations from each node j_{k-1} to node j_k and taking advantage of (4), $g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ can reach root node 1.

In order to show how $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ could be passed from node v to 1 in a similar fashion, we should note from (4), that for any path connecting node 1 and v, which does not include the edge $(1,2)$, then $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})=0$. If $\theta_{1\to v}$ is defined over a path that includes edge $(1,2)$ then

$$\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v}) = \nabla_z g_v(Y_{v,n}|\, z)|_{z=x_1+\theta_{1\to v}}$$

since $\nabla_{\theta_{1\to 2}}[x_1+\theta_{1\to v}]=1$. As before then each node j_k should forward to node j_{k-1} the message

$$\nabla_{x_{j_k}+\theta_{j_k\to v}} g_v(Y_{v,n}|\, x_{j_k}+\theta_{j_k\to v})|_{x_{j_k}=x_{j_{k-1}}+\theta_{j_{k-1}\to j_k}} = \nabla_{x_{j_{k-1}}+\theta_{j_{k-1}\to v}} g_v(Y_{v,n}|\, x_{j_{k-1}}+\theta_{j_{k-1}\to v})$$

so that $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ travels along the path $\overrightarrow{(v,j_l)}, \overrightarrow{(j_l,j_{l-1})}, \ldots, \overrightarrow{(j_2,j_1)}, \overrightarrow{(j_1,1)}$.

Previously we have shown that the sufficient statistics that root node 1 needs to compute $\theta_{1\to 2,n}$ as well as propagating the densities in (6)-(9) are $\prod_{v\in\mathcal{V}\setminus\{1\}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $\sum_{v\in\mathcal{V}} \frac{\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|x_1+\theta_{1\to v})}{g_v(Y_{v,n}|x_1+\theta_{1\to v})}$. Therefore we could define messages from each node in a way that it is not necessary to transmit each $\nabla_{\theta_{1\to 2}} g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ and $g_v(Y_{v,n}|\, x_1+\theta_{1\to v})$ separately in all different possible paths $\overrightarrow{(v,j_l)}, \overrightarrow{(j_l,j_{l-1})}, \ldots, \overrightarrow{(j_2,j_1)}, \overrightarrow{(j_1,1)}$.

For this reason we inherit a generalized version of belief propagation (BP) approach as described in [10] where messages are defined $m_{1,i\to j}$ and $m_{2,i\to j}$ from node i to j for all $(i,j)\in\mathcal{E}$

$$m_{1,i\to j}(x_j) = g_i(Y_{i,n}|\, x_j+\theta_{j\to i}) \times \prod_{p\in \mathrm{ne}(i)\setminus\{j\}} m_{1,p\to i}(x_j+\theta_{j\to i}) \qquad (10)$$

$$m_{2,i\to j}(x_j) = \frac{\nabla_{\theta_{1\to 2}} g_i(Y_{i,n}|\, x_j+\theta_{j\to i})}{g_i(Y_{i,n}|\, x_j+\theta_{j\to i})} + \sum_{p\in \mathrm{ne}(i)\setminus\{j\}} m_{2,p\to i}(x_j+\theta_{j\to i}) \qquad (11)$$

where $\mathrm{ne}(i)$ is the set of neighbouring nodes of i in $\mathcal{V}$.

Equations (5)-(9) can be reproduced for any other root node r and parameter $\theta_{r\to q}$, $(q,r)\in\mathcal{E}$, by obvious substitutions to the subscript indices. In addition we can define for each root node r an appropriate message scheduling, where we choose to use only messages $m_{1,i\to j}(x_j)$ and $m_{1,i\to j}(x_j)$ only for (i,j) directed towards that root node, i.e. start from the outer branches of the graphical model and leading each time to node r. So after the root node receives its messages from its neighbours it will have available: $\prod_{p\in\mathrm{ne}(r)} m_{1,p\to r}(x_1+\theta_{r\to p}) = \prod_{v\in\mathcal{V}\setminus\{r\}} g_v(Y_{v,n}|\, x_r+\theta_{r\to v})$ and $\sum_{p\in\mathrm{ne}(r)} m_{2,p\to r}(x_r+\theta_{r\to p}) = \sum_{v\in\mathcal{V}\setminus\{r\}} \frac{\nabla_{\theta_{r\to q}} g_v(Y_{v,n}|x_r+\theta_{r\to v})}{g_v(Y_{v,n}|x_r+\theta_{r\to v})}$, as desired.

5 Distributed RML Ideal Algorithm

In this section we demonstrate how the general distributed parameter estimation problem for sensor localisation can be solved using RML. Using the results of Section 4 we estimate $\theta_{r\to q}$ for any edge (r,q) using now an arbitrary root node r. At each iteration n all edges should be updated in a cyclic fashion using a valid root node and hence $\theta=\theta_{i\to j_{ij}}$ will be updated for all $(i,j)\in\mathcal{E}$.

Algorithm 1:
For each edge $(q,r)\in\mathcal{E}$ assign a valid root node, say r, so as to update parameter $\theta_{r\to q}$
At time n,

Prediction Update for all nodes: For all nodes $v\in\mathcal{V}$ derive the prediction densities $\pi^{\theta}_{v,n|n-1}(x_{v,n}) = \int f_v(x_{v,n}|x_{v,n-1})\pi^{\theta}_{v,n-1}(x_{v,n-1})dx_{v,n-1}$ and their derivatives $\nabla_{\theta_{r\to q}}\pi^{\theta}_{v,n|n-1}(x_{v,n}) = \int f_v(x_{v,n}|x_{v,n-1})\nabla_{\theta_{r\to q}}\pi^{\theta}_{v,n-1}(x_{v,n-1})dx_{v,n-1}$.

Propagate messages: After $Y_{v,n}$ is received from all nodes $v\in\mathcal{V}$ propagate all possible messages $m_{1,i\to j}$, $m_{2,i\to j}$ given by (10) and (11) for all edges $(i,j)\in\mathcal{E}$ in the network.

Use messages to compute sufficient statistics: At each node v compute $\prod_{p\in\mathrm{ne}(v)} m_{1,p\to v}(x_v+\theta_{v\to p})$ and, for each root node r, $\sum_{p\in\mathrm{ne}(r)} m_{2,p\to r}(x_r+\theta_{r\to p})$.

Update the parameter $\theta_{r\to q}$: At each root node r set

$$\theta_{r\to q,n+1} = \theta_{r\to q,n} + \gamma_n \nabla_{\theta_{r\to q}} J_r(Y_n;\tilde{\theta}_n),$$

where $\tilde{\theta}_n$ is defined so that $\tilde{\theta}_{r\to q}=\theta_{r\to q,n}, \tilde{\theta}_{l\to e}=\theta_{l\to e,n}$, for all $(l,e)\in\mathcal{E}$ and $\nabla_{\theta_{r\to q}} J(Y_n;\theta)$ is given by an expression similar to (5).

Update Filtering density and derivative: Using incoming messages, update at all nodes the current state belief $\pi^{\theta}_{v,n}(x_{v,n})$ as in (7) and its derivative $\nabla_{\theta_{r\to q}}\pi^{\theta}_{v,n}(x_{v,n})$ as in (9).

Typically, the step-sizes are selected as $\gamma_n = n^{-\gamma}$, where $\gamma > 0.5$, so that $\sum_n \gamma_n = \infty$ and $\sum_n {\gamma_n}^2 < \infty$.

As for standard belief propagation [10], this algorithm will only be exact when the graph/network admits a tree structure. Applying belief propagation algorithms to non-tree topologies is generally referred as Loopy Belief Propagation (LBP). In some cases LBP can lead to very good approximations given [15]. Similarly, we can apply this algorithm to general topologies and we demonstrate in simulations that this loopy version can exhibit excellent performance.

Using Particle Filters:

This algorithm relies on the evaluation of complex multi-dimensional integrals given in equations (5)-(9). In the general non linear non Gaussian case this is impossible and one must rely on approximation and simulation-based methods. It is important to emphasise that we have **not** made any linearity or gaussianity assumption in our framework. Hence in the nonlinear non-gaussian case, Sequential Monte Carlo or in other words Particle Filters can be used.

In [11, 12] a centralised implementation of particle filtering for RML has been derived. This implementation can be extended to our distributed framework. In the above algorithm the state's belief at each node $\pi^{\theta}_{i,n}$ as well as the prediction density $\pi^{\theta}_{i,n|n-1}$ will have to be replaced by their particle filter approximations $\hat{\pi}^{\theta}_{i,n}$ and $\hat{\pi}^{\theta}_{i,n|n-1}$, which will be weighted sums of Dirac delta-mass functions [4]. It can be shown that the messages defined above evaluated at the particle's states are adequate to recur $\hat{\pi}^{\theta}_{i,n}$ and $\hat{\pi}^{\theta}_{i,n|n-1}$ as well as their derivatives.

6 Numerical Example

In this section we shall demonstrate how to solve the sensor self-localisation problem using our framework. We will be using the ideal algorithm of Section 5 to solve the sensor localisation problem for the linear Gaussian case.

We consider an M node sensor network. At each node v, the target being tracked yields observation $Y_{v,n}$ and obeys the following dynamics

$$\begin{aligned} X_{v,n} &= AX_{v,n-1} + BV_{v,n}, \\ Y_{v,n} &= CX_{v,n} + DW_{v,n} \end{aligned}$$

with $V_{v,n} \overset{i.i.d.}{\sim} \mathcal{N}(0, Q)$ and $W_{v,n} \overset{i.i.d.}{\sim} \mathcal{N}(0, R)$.

Thanks to the linear and Gaussian assumptions, we have at time n

$$\begin{aligned} \pi_{v,n|n-1}(x_{v,n}) = &\quad \mathcal{N}_{x_n}(\mu_{v,n|n-1}, \Sigma_{v,n|n-1}) \\ \pi_{v,n}(x_{v,n}) = &\quad \mathcal{N}_{x_n}(\mu_{v,n|n}, \Sigma_{v,n|n}) \end{aligned}$$

whose parameters can be computed using a distributed

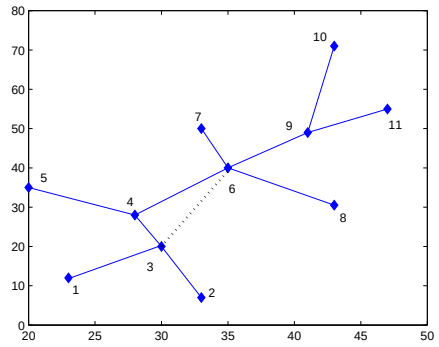

Figure 1: Sensor Network used for target tracking; in the numerical example its localisation parameter θ is estimated.

Kalman filter recursion for each node v

$$\begin{aligned} \mu_{v,n|n-1} &= A\,\mu_{v,n-1|n-1} \\ \Sigma_{v,n|n-1} &= A\Sigma_{v,n-1|n-1}A^T + BQB^T \\ m_{v,n} &= C\mu_{v,n|n-1} \\ S_{v,n} &= C\Sigma_{v,n|n-1}C^T + M^{-1}DRD^T \\ K_{v,n} &= \Sigma_{v,n|n-1}C^T S^{-1}_{v,n} \\ \mu_{v,n|n} &= \mu_{v,n|n-1} \\ &\quad + K_{v,n}(M^{-1}\sum_{i\in\mathcal{V}}(Y_{i,n} - C\theta_{v\to i}) - m_{v,n}) \\ \Sigma_{v,n|n} &= \Sigma_{v,n|n-1} - K_{v,n}C\Sigma_{v,n|n-1} \end{aligned}$$

So it is only necessary to propagate the mean and covariance of these densities. Moreover since $\nabla_{\theta_{v\to k}}\mathcal{N}_x(m_\theta, \Sigma) = (\nabla_{\theta_{v\to k}} m_\theta)^T\mathcal{N}_x(m_\theta, \Sigma)\Sigma^{-1}(x - m_\theta)$, in order to propagate the derivatives of the densities we only need to propagate $\nabla_{\theta_{v\to k}}\mu_{v,n|n-1}$ and $\nabla_{\theta_{v\to k}}\mu_{v,n|n}$. Also to update each edge parameter $\theta_{r\to q}$, we can use the analytical expression of the predictive distribution $P_\theta(Y_n|Y_{0:n-1})$ when taking the derivative of $\log P_\theta(Y_n|Y_{0:n-1})$.

For the sensor network in Figure 1 ignore first the dotted line between nodes 3 and 6. In this case, the graph has a tree structure and we can solve for any $\theta_{i\to j}$ exactly. We choose nodes $\{3, 4, 6, 9\}$ as root nodes and update at each iteration their adjacent edges. For practical implementation reasons we choose to use a constant step size $\gamma_n = 10^{-3}$. For stochastic approximation in general, decreasing step-sizes are essential conditions of convergence. If fixed step-sizes are used, then we may still have convergence, but now the iterates "oscillate" about their limiting values with variance proportional to the step-size. We also initialise $\theta_{i\to j} = 0$ for all $(i, j) \in \mathcal{E}$. In Figures 2, 3 we illustrate the convergence to the correct values of all the localisation parameters relative to node 6, $\theta_{6\to v}$, for all $v \in \mathcal{V}$.

Next we solve for the sensor network in Figure 1 but with the dotted line connected. Now the sensor network has a loop in its topology defined by $\mathcal{E}$ at $\{3, 4, 6\}$.

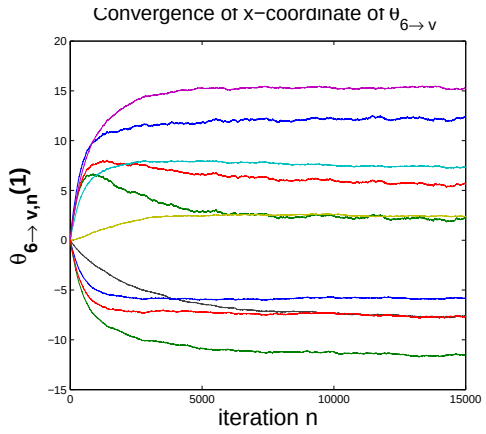

Figure 2: Convergence of x-coordinate of $\theta_{6\rightarrow v,n}$ for the sensor net with dotted line not connected

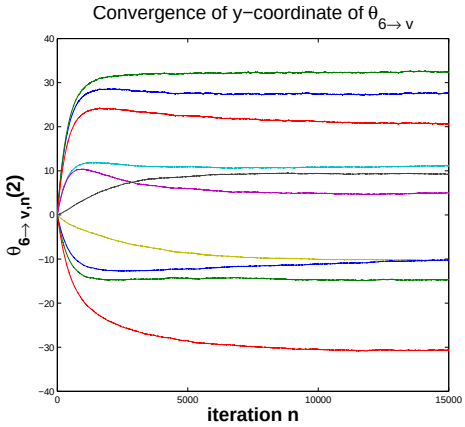

Figure 3: Convergence of y-coordinate of $\theta_{6\rightarrow v,n}$ for the sensor net with dotted line not connected

We choose again nodes $\{3, 4, 6, 9\}$ as root nodes and update at each iteration their adjacent edges. As before, we choose to use a constant step size $\gamma_n = 10^{-3}$ and initialise $\theta_{i\rightarrow j} = 0$. In Figures 4, 5 we illustrate the convergence of all the localisation parameters relative to node 6, $\theta_{6\rightarrow v}$ when LBP is used. Note that the errors of the solution for $\theta_{6\rightarrow v}$ are very small, but the convergence rate is now slower.

7 Conclusion

In this paper, we have presented a fully decentralized algorithm to perform recursive static parameter estimation in dynamic graphical models. We have used this approach to formulate the sensor registration problem and proposed an algorithm to solve the sensor localisation problem. For linear Gaussian graphs, our algorithm can be implemented exactly using a distributed version of the Kalman filter and its derivative. In the general non linear and non Gaussian case, Sequential Monte Carlo can be used.

Acknowledgement: This work has been funded by the DIF-DTC.

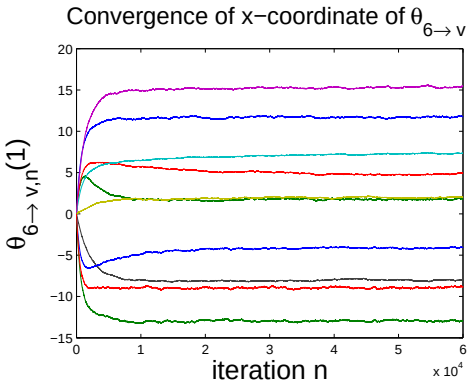

Figure 4: Convergence of x-coordinate of $\theta_{6\rightarrow v,n}$ when LBP is applied to the sensor net with the dotted line connected

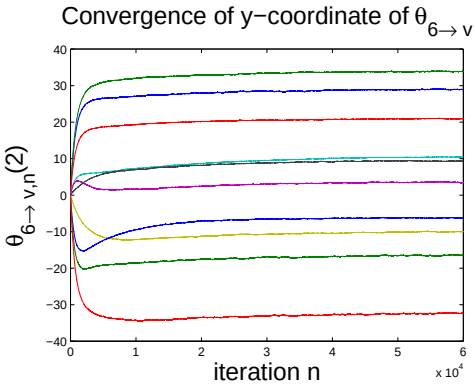

Figure 5: Convergence of y-coordinate of $\theta_{6\rightarrow v,n}$ when LBP is applied to the sensor net with the dotted line connected

References

[1] I.F. Akyildiz, W. Su , Y. Sankarasubramaniam, E. Cayirci,"A Survey on Sensor Networks ", *IEEE Communications Magazine*, Vol.40(8), pp:102–114, August, 2002

[2] Y. Bar-Shalom, and X. R. Li, *Estimation and Tracking: Principles, Techniques, and Software,* Artech House Inc.,1993.

[3] T. Brehard, and J.-P. Le Cadre, "Distributed Target Tracking for Nonlinear Systems: Application to Bearings-only Tracking, " *In Proceedings Conference Information Fusion (FUSION'05)*, Vol. 40(2), Philadelphia, USA, 2005.

[4] A. Doucet, J.F.G. de Freitas and N.J. Gordon, *Sequential Monte Carlo Methods in Practice.* New York: Springer, 2001.

[5] M.L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom, "Multi-sensor Resource Deployment using Posterior Cramer-Rao Bounds, " *IEEE Transactions on Aerospace and Electronic Engineering*, Vol. 40(2), pp:339-416, April, 2004.

[6] A. T. Ihler and J. W. Fisher and R. L. Moses and A. S. Willsky, "Nonparametric Belief Propagation for Self-Localisation in Sensor Networks, " *IPSN'04: Proceedings of 3rd International Symposium on Information processing in sensor networks*, pp: 225–233, Berkeley, California, 2004

[7] F. LeGland, and L. Mevel, "Recursive Identification in Hidden Markov Models" *Proceedings of 36th IEEE Conference Decision and Control*, pp:3468-3473, 1997.

[8] N. N. Okello, and S. Challa, "Joint Sensor Registration for Distributed Trackers, " *IEEE Transactions on Aerospace and Electronic Systems*, March, 2003.

[9] N. N. Okello, and B. Ristic, "Maximum Likelihood Registration for Dissimilar Sensors, " *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 39(3), pp:1074–1083, July 2003.

[10] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference,* Morgan-Kauffman, 1988.

[11] G. Poyiadjis, A. Doucet and S.S. Singh, "Particle Methods for Optimal Filter Derivative: Application to Parameter Estimation," *Proceedings IEEE ICASSP*, 2005.

[12] G. Poyadjis, A. Doucet and S.S. Singh, "Maximum Likelihood Parameter Estimation using Particle Methods", Proceedings of the American Statistical Association, 2005.

[13] G.Ch. Pflug, *Optimization of stochastic models: the interface between simulation and optimization.* Boston: Kluwer, 1996.

[14] J. Vermaak, S. Maskell, and M. Briers, "Online Sensor Registration, " *IEEE Aerospace Conference*, Big Sky, MT, 2006 (to appear).

[15] Weiss Y., "Correctness of Local Probability Propagation in Graphical Models with Loops, " *Neural Computation*, Vol. 12 pp:1–41, 2000.

[16] Weiss Y. and Freeman W.T., "Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology, " *Neural Computation*, Vol. 13 pp:2173–2200, 2001.