MODEL BASED LEARNING OF SIGMA POINTS IN UNSCENTED KALMAN FILTERING

Ryan Turner and Carl Edward Rasmussen

University of Cambridge
Department of Engineering
Trumpington Street, Cambridge CB2 1PZ, UK

ABSTRACT

The unscented Kalman filter (UKF) is a widely used method in control and time series applications. The UKF suffers from arbitrary parameters necessary for a step known as sigma point placement, causing it to perform poorly in nonlinear problems. We show how to treat sigma point placement in a UKF as a learning problem in a model based view. We demonstrate that learning to place the sigma points correctly from data can make sigma point collapse much less likely. Learning can result in a significant increase in predictive performance over default settings of the parameters in the UKF and other filters designed to avoid the problems of the UKF, such as the GP-ADF. At the same time, we maintain a lower computational complexity than the other methods. We call our method UKF-L.

1. INTRODUCTION

Filtering in linear dynamical systems (LDS) and nonlinear dynamical systems (NLDS) is frequently used in many areas, such as signal processing, state estimation, control, and finance/econometric models. Filtering (inference) aims to estimate the state of a system from a stream of noisy measurements. Imagine tracking the location of a car based on odometer and GPS sensors, both of which are noisy. Sequential measurements from both sensors are combined to overcome the noise in the system and to obtain an accurate estimate of the system state. Even when the full state is only partially measured, it can still be inferred; in the car example the engine temperature is unobserved, but can be inferred via the nonlinear relationship from acceleration. To exploit this relationship appropriately, inference techniques in nonlinear models are required; they play an important role in many practical applications.

LDS and NLDS belong to a class of models known as state-space models. A state-space model assumes that there exists a sequence of latent states $\mathbf{x}_t$ that evolve over time according to a Markovian process specified by a transition function f. The latent states are observed indirectly in $\mathbf{y}_t$ through a measurement function g. We consider state-space models given by

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}\,, \quad \mathbf{x}_t \in \mathbb{R}^M, \\ \mathbf{y}_t &= g(\mathbf{x}_t) + \boldsymbol{\nu}\,, \quad \mathbf{y}_t \in \mathbb{R}^D\,. \end{aligned} \tag{1}$$

Here, the system noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon)$ and the measurement noise $\boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\nu)$ are both Gaussian. In the LDS case, f and g are linear functions, whereas the NLDS covers the general nonlinear case.

Kalman filtering [1] corresponds to exact (and fast) inference in the LDS, however it can only model a limited set of phenomena. For the last few decades, there has been interest in NLDS for more general applicability. In the state-space formulation, the nonlinear systems do not generally yield analytically tractable algorithms.

The most widely used approximations for filtering in NLDS are the extended Kalman filter (EKF) [2] and the unscented Kalman filter (UKF) [3]. The EKF linearizes f and g at the current estimate of x_t and treats the system as a nonstationary linear system even though it is not. The UKF propagates several estimates of x_t through f and g and reconstructs a Gaussian distribution assuming the propagated values came from a linear system. The locations of the estimates of x_t are known as the *sigma points*. Many heuristics have been developed to help set the sigma point locations [4]. Unlike the EKF, the UKF has free parameters that determine where to put the sigma points. The key idea in this paper, is that the UKF and EKF are doing exact inference in a model that is somewhat perverted from the original model described in the state-space formulation. The interpretation of EKF and UKF as models, not just approximate methods, allows us to better identify their underlying assumptions. It also enables us to *learn* the free parameters in the UKF in a model based manner from training data. If the settings of the sigma point are a poor fit to the underlying dynamical system, the UKF can make horrendously poor predictions. This paper's contribution is a strategy for improving the UKF through a novel learning algorithm for appropriate sigma point placement: we call this method UKF-L.

2. UNSCENTED KALMAN FILTERING

We first review how filtering and the UKF works and then explain the UKF's generative assumptions. Filtering methods consist of three steps: time update, prediction step, and measurement update. They iterate in a predictor-corrector setup. In the time update we find $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\, d\mathbf{x}_{t-1}\,, \quad (2)$$

using $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$. In the prediction step we predict the observed space, $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ using $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$:

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})\, d\mathbf{x}_t\,. \quad (3)$$

Finally, in the measurement update we find $p(\mathbf{x}_t|\mathbf{y}_t)$ using information from how good (or bad) the prediction in the prediction step is:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})\,. \quad (4)$$

In the linear case all of these equations can be done analytically using matrix multiplications. The EKF explicitly linearizes f and g at the point $\mathbb{E}[\mathbf{x}_t]$ at each step. The UKF uses the whole distribution on $\mathbf{x}_t$, not just the mean, to place sigma points and implicitly linearize the dynamics, which we call the *unscented transform* (UT). In one dimension the sigma points roughly correspond to the mean and α-standard deviation points; the UKF generalizes this idea to higher dimensions. The exact placement of sigma points depends on the unitless parameters $\{\alpha, \beta, \kappa\} \in \mathbb{R}^+$ through

$$\mathcal{X}^0 := \mu,\ \mathcal{X}^i := \mu \pm (\sqrt{(D+\lambda)\Sigma})_i \quad (5)$$

$$\lambda := \alpha^2(D+\kappa) - D\,, \quad (6)$$

where $\sqrt{\cdot}_i$ refers to the ith row of the Cholesky factorization.[1] The sigma points have weights assigned by:

$$w_m^0 := \lambda/(D+\lambda),\ w_c^0 := \lambda/(D+\lambda) + (1-\alpha^2+\beta)$$
$$w_m^i := w_c^i := 1/2(D+\lambda)\,, \quad (7)$$

where w_m is used to reconstruct the predicted mean and w_c used for the predicted covariance. We can loosely interpret the unscented transform as approximating the input distribution by $2D+1$ point masses at $\mathcal{X}$ with weight w. Once the sigma points $\mathcal{X}$, have been calculated the filter accesses f and g as black boxes to find $\mathcal{Y}_t$, either $f(\mathcal{X}_t)$ or $g(\mathcal{X}_t)$ depending on the step. The UKF reconstructs the mean and variance of the propagated distribution from $\mathcal{Y}_t$ *had* the dynamics been linear. It does not guarantee the moments will match the moment of the true non-Gaussian distribution.

[1] If $\sqrt{P} = A \Rightarrow P = A^\top A$, then we use the rows in (5). If $P = AA^\top$, then we use the columns.

Algorithm 1 Sampling data from UKF's implicit model

1: $p(\mathbf{x}_1|\emptyset) \leftarrow (\mu_0, \Sigma_0)$
2: **for** $t = 1$ to T **do**
3: Prediction step: $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ using $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$
4: Sample $\mathbf{y}_t$ from prediction step distribution
5: Measurement update: $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ using $\mathbf{y}_t$
6: Time update: find $p(\mathbf{x}_{t+1}|\mathbf{y}_{1:t})$ using $p(\mathbf{x}_t|\mathbf{y}_{1:t})$
7: **end for**

Both the EKF and the UKF approximate the nonlinear state-space as a nonstationary linear system. The UKF defines its own *generative process* that linearizes the nonlinear function f and g wherever in $\mathbf{x}_t$ a UKF filtering the time series would expect $\mathbf{x}_t$ to be. Therefore, it is possible to sample synthetic data from the UKF by sampling from its one-step-ahead predictions as seen in Algorithm 1. The sampling procedure augments the filter: predict-sample-correct. If we use the UKF with the same $\{\alpha, \beta, \kappa\}$ used to generate synthetic data, then the one-step-ahead predictive distribution will be the exact same distribution the data point was sampled from.

2.1. Setting the parameters

We summarize all the parameters as $\theta := \{\alpha, \beta, \kappa\}$. For any setting of θ the UKF will give identical predictions to the Kalman filter if f and g are both linear. Many of the heuristics for setting θ assume f and g are linear (or close to it), which is not the problem the UKF solves. For example, one of the heuristics for setting θ is that $\beta = 2$ is optimal if the state distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is exactly Gaussian [5]. However, the state distribution will seldom be Gaussian unless the system is linear, in which case any setting of θ is exact! It is often recommended to set the parameters to $\alpha = 1$, $\beta = 0$, and $\kappa = 2$.

3. THE ACHILLES' HEEL OF THE UKF

The UKF can have very poor performance because its predictive variances can be far too small if the sigma points are placed in inconvenient locations. A too small predictive variance will cause observations to have too much weight in the measurement update, which causes the UKF to fit to noise. Meaning, the UKF will perform poorly even when evaluated on root-mean-square-error (RMSE), which only uses the predictive mean.

In the most extreme case, the UKF can give a delta spike predictive distribution. We call this *sigma point collapse*. As seen in Fig. 1, when the sigma points are arranged together horizontally the UKF has no way to know the function varies anywhere. We aim to learn the parameters θ in such a way that collapse becomes unlikely. Anytime col-

lapse happens in training the marginal likelihood will be very low. Hence, the learned parameters will avoid anywhere this delta spike occurred in training. Maximizing the marginal likelihood is tricky since it is not well behaved for settings of θ that cause sigma point collapse.

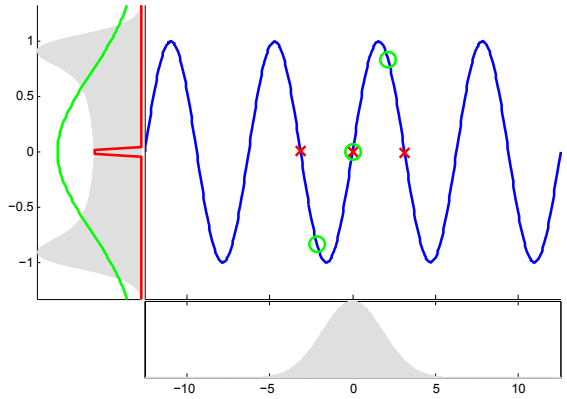


Fig. 1. An illustration of a good and bad assignment of sigma points. The lower panel shows the true input distribution. The center panel shows the sinusoidal system function f (blue) and the sigma points for $\alpha = 1$ (red crosses) and $\alpha = 0.68$ (green rings). The left panel shows the true output distribution (shaded), the output distribution under $\alpha = 1$ (red spike) and $\alpha = 0.68$ (green). Using a different set of sigma points we can get either a completely degenerate solution (a delta spike) or a near optimal approximation within the class of Gaussian approximations.

4. MODEL BASED LEARNING

A common approach to estimating model parameters θ in general is to maximize the log marginal likelihood

$$\ell(\theta) := \log p(\mathbf{y}_{1:T}|\theta) = \sum_{t=1}^{T} \log p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \theta)\,. \qquad (8)$$

Hence we can equivalently maximize the sum from the one-step-ahead predictions. One might be tempted to apply a gradient based optimizer on (8), but as seen in Fig. 2 the marginal likelihood can be very noisy. The noise, or instability in the likelihood, is likely the result of the phenomenon explained in Section 3, where a slight change in parameterization can avoid problematic sigma point placement. This makes the application of a gradient-based optimizer hopeless.

It is also possible to apply Markov chain Monte Carlo (MCMC) and integrate out the parameters. However, this is usually overkill as the posterior on θ is usually highly peaked unless T is very small. Tempering must be used as mixing will be difficult if the chain is not initialized inside the posterior peak. Even in the case when T is small enough to spread the posterior out, we would still like a single point estimate for computational speed on the test set.[2]

We will focus on learning using a Gaussian process (GP) based optimizer [6]. Since the marginal likelihood surface has an underlying smooth function but contains what amounts to additive noise, a probabilistic regression method seems a natural fit for finding the maximum.

5. GAUSSIAN PROCESS OPTIMIZERS

Gaussian processes form a prior over functions. Estimating the parameters amounts to finding the maximum of a structured function: the log marginal likelihood. Therefore, it seems natural to use a prior over functions to guide our search. The same principle has been applied to integration in [7].

GP optimization (GPO) allows for effective *derivative free* optimization. We consider the maximization of a likelihood function $\ell(\theta)$. GPs allow for derivative information $\partial_\theta \ell$ to be included as well, but in our case that will not be very useful due to the function's instability.

GPO treats optimization as a sequential decision problem in a probabilistic setting, receiving reward r when using the right input θ to get a large function value output $\ell(\theta)$. At each step GPO uses its posterior over the objective function $p(\ell(\theta))$ to look for θ it believes have large function value $\ell(\theta)$. A maximization strategy that is *greedy* will always evaluate the function $p(\ell(\theta))$ where the mean function $\mathbb{E}[\ell(\theta)]$ is the largest. A strategy that trades-off *exploration* with *exploitation* will take into account the posterior variance $\mathrm{Var}[\ell(\theta)]$. Areas of θ with high variance carry a possibility of having a large function value or high reward r. The optimizer is programmed to evaluate at the maxima of

$$J(\theta) := \mathbb{E}[\ell(\theta)] + K\sqrt{\mathrm{Var}[\ell(\theta)]}\,, \qquad (9)$$

where K is a constant to control the exploration exploitation trade-off. The optimizer must also find the maximum of J, but since it is a combination of the GP mean and variance functions it is easy to optimize with gradient methods.

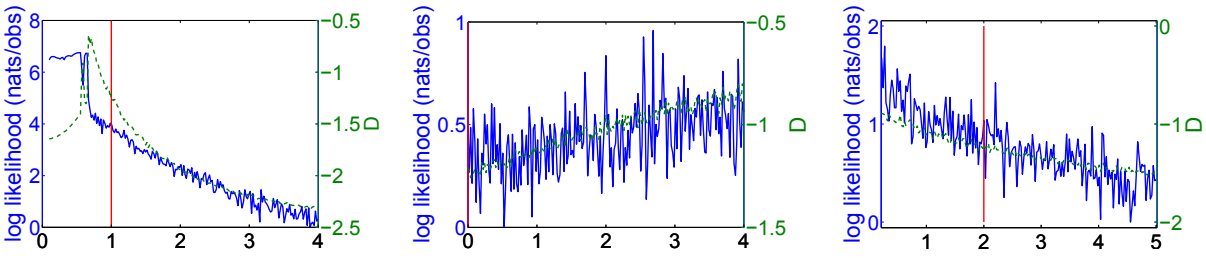
6. EXPERIMENTS AND RESULTS

We test our method on filtering in three dynamical systems: the sinusoidal dynamics used in [8], the Kitagawa dynamics used in [9, 10], and pendulum dynamics used in [9]. The sinusoidal dynamics are described by

$$x_{t+1} = 3\sin(x_t) + w,\ w \sim \mathcal{N}(0, 0.1^2)\,, \qquad (10)$$

$$y_t = \sigma(x_t/3) + v,\ v \sim \mathcal{N}(0, 0.1^2)\,. \qquad (11)$$

[2] If we want to integrate the parameters out we must run the UKF with each sample of $\theta|\mathbf{y}_{1:T}$ during test and average. To get the optimal point estimate of the posterior we would like to compute the *Bayes' point*.



(a) $\alpha \in (0.1, 4)$ (b) $\beta \in (0, 4)$ (c) $\kappa \in (0.1, 5)$

Fig. 2. Illustration of the UKF when applied to a pendulum system. Cross section of the marginal likelihood (blue line) varying the parameters one at a time from the defaults (red vertical line). We shift the marginal likelihood, in nats/observation, to make the lowest value zero. The dashed green line is the total variance diagnostic $D := \mathbb{E}\left[\log(|\Sigma|/|\Sigma_0|)\right]$, where Σ is the predictive variance in one-step-ahead prediction. We divide out the variance Σ_0 of the time series when treating it as iid to make D unitless. Values of θ with small predictive variances closely track the θ with low marginal likelihood.

where $\sigma(\cdot)$ represents a sigmoid. The Kitagawa model is described by

$$x_{t+1} = 0.5x_t + \frac{25x_t}{1+x_t^2} + w,\ w \sim \mathcal{N}(0, 0.2^2)\,, \tag{12}$$

$$y_t = 5\sin(2x_t) + v,\ v \sim \mathcal{N}(0, 0.01^2)\,. \tag{13}$$

The Kitagawa model was presented as filtering problem in [10]. The pendulum dynamics is described by a discretized ordinary differential equation (ODE) at $\Delta_t = 400\,\text{ms}$. The pendulum possesses a mass $m = 1\,\text{kg}$ and a length $l = 1\,\text{m}$. The pendulum angle φ is measured anti-clockwise from hanging down. The state $\mathbf{x} = [\varphi, \dot{\varphi}]^\top$ of the pendulum is given by the angle φ and the angular velocity $\dot{\varphi}$. The ODE is

$$\frac{d}{dt}\begin{bmatrix}\dot{\varphi}\\ \varphi\end{bmatrix} = \begin{bmatrix}\frac{-mlg\sin\varphi}{ml^2}\\ \dot{\varphi}\end{bmatrix}\,, \tag{14}$$

where g the acceleration of gravity. This model is commonly used in stochastic control for the inverted pendulum problem [11]. The measurement function is

$$\mathbf{y}_t = \begin{bmatrix}\arctan\left(\frac{p_1 - l\sin(\varphi_t)}{p_1 - l\cos(\varphi_t)}\right)\\ \arctan\left(\frac{p_2 - l\sin(\varphi_t)}{p_2 - l\cos(\varphi_t)}\right)\end{bmatrix}\,,\ \begin{bmatrix}p_1\\ p_2\end{bmatrix} = \begin{bmatrix}1\\ -2\end{bmatrix}\,, \tag{15}$$

which corresponds to *bearings only measurement* since we do not directly observe the velocity. We use system noise $\Sigma_w = \text{diag}([0.1^2\ 0.3^2])$ and $\Sigma_v = \text{diag}([0.2^2\ 0.2^2])$ as observation noise.

For all the problems we compare to UKF-D, EKF, the GP-UKF, and GP-ADF, and the time independent model (TIM); we use UKF-D to denote a UKF with default parameter settings, and UKF-L for learned parameters. The TIM treats the data as iid normal and is inserted as a reference point. The GP-UKF and GP-ADF use GPs to approximate f and g and exploit the properties of GPs to make tractable predictions. The Kitagawa and pendulum dynamics were used by [9] to illustrate the performance of the GP-ADF and the very poor performance of the UKF. [9] used the default settings of $\alpha = 1$, $\beta = 0$, $\kappa = 2$ for all of the experiments. We used exploration trade off $K = 2$ for the GPO in all the experiments. Additionally, GPO used the squared-exponential with automatic relevance determination (SE-ARD) covariance function plus a noise term of 0.01 nats/observation. We set the GPO to have a maximum number of function evaluations of 100, even better results can be obtained by letting the optimizer run longer to hone the parameter estimate. We show that by *learning* appropriate values for θ we can match, if not exceed, the performance of the GP-ADF and other methods.

The models were evaluated on one-step-ahead prediction. The evaluation metrics were the negative log-predictive likelihood (NLL), the mean squared error (MSE), and the mean absolute error (MAE) between the mean of the prediction and the true value. Note that unlike the NLL, the MSE and MAE do not account for uncertainty. The MAE will be more difficult for approximate methods than MSE. For MSE, the optimal action is to predict the mean of the predictive distribution, while for the MAE it is the median. Most approximate methods attempt to moment match to a Gaussian and preserve the mean; the median of the true predictive distribution is implicitly assumed to be the same as mean. Quantitative results are shown in Table 1.

6.1. Sinusoidal dynamics

The models were trained on $T = 1000$ observations from the sinusoidal dynamics, and tested on $R = 10$ restarts with $T = 500$ points each. The initial state was sampled from a standard normal $x_1 \sim \mathcal{N}(0, 1)$. The UKF optimizer found the optimal values $\alpha = 2.0216$, $\beta = 0.2434$, and $\kappa = 0.4871$.

Table 1. Comparison of the methods on the sinusoidal, Kitagawa, and pendulum dynamics. The measures are supplied with 95% confidence intervals and a p-value from a one-sided t-test under the null hypothesis UKF-L is the same or worse as the other methods. NLL is reported in nats/observation, while MSE and MAE are in the units of $\mathbf{y}^2$ and $\mathbf{y}$, respectively. Since the observations in the pendulum data are angles we projected the means and the data to the complex plane before computing MSE and MAE.

Method	NLL	p-value	MSE	p-value	MAE	p-value
Sinusoid ($T = 500$ and $R = 10$)						
UKF-D	$10^{-1}\times$ -4.58±0.168	<0.0001	$10^{-2}\times$ 2.32±0.0901	<0.0001	$10^{-1}\times$ 1.22±0.0253	<0.0001
UKF-L ⋆	$\mathbf{-5.53 \pm 0.243}$	N/A	$\mathbf{1.92 \pm 0.0799}$	N/A	$\mathbf{1.09 \pm 0.0236}$	N/A
EKF	-1.94±0.355	<0.0001	3.03±0.127	<0.0001	1.37±0.0299	<0.0001
GP-ADF	-4.13±0.154	<0.0001	2.57±0.0940	<0.0001	1.30±0.0261	<0.0001
GP-UKF	-3.84±0.175	<0.0001	2.65±0.0985	<0.0001	1.32±0.0266	<0.0001
TIM	-0.779±0.238	<0.0001	4.52±0.141	<0.0001	1.78±0.0323	<0.0001
Kitagawa ($T = 10$ and $R = 200$)						
UKF-D	$10^{0}\times$ 3.78±0.662	<0.0001	$10^{0}\times$ 5.42±0.607	<0.0001	$10^{0}\times$ 1.32±0.0841	<0.0001
UKF-L ⋆	$\mathbf{2.24 \pm 0.369}$	N/A	$\mathbf{3.60 \pm 0.477}$	N/A	$\mathbf{1.05 \pm 0.0692}$	N/A
EKF	617±554	0.0149	9.69±0.977	<0.0001	1.75±0.113	<0.0001
GP-ADF	2.93±0.0143	0.0001	18.2±0.332	<0.0001	4.10±0.0522	<0.0001
GP-UKF	2.93±0.0142	0.0001	18.1±0.330	<0.0001	4.09±0.0521	<0.0001
TIM	48.8±2.25	<0.0001	37.2±1.73	<0.0001	4.54±0.179	<0.0001
Pendulum ($T = 200 = 80\,\mathrm{s}$ and $R = 100$)						
UKF-D	$10^{0}\times$ 3.17±0.0808	<0.0001	$10^{-1}\times$ 5.74±0.0815	<0.0001	$10^{-1}\times$ 11.5±0.0988	<0.0001
UKF-L ⋆	$\mathbf{0.392 \pm 0.0277}$	N/A	$\mathbf{1.93 \pm 0.0378}$	N/A	6.14±0.0577	N/A
EKF	0.660±0.0429	<0.0001	1.98±0.0429	0.0401	$\mathbf{6.11 \pm 0.0611}$	0.779
GP-ADF	1.18±0.00681	<0.0001	4.34±0.0449	<0.0001	10.3±0.0589	<0.0001
GP-UKF	1.77±0.0313	<0.0001	5.67±0.0714	<0.0001	11.6±0.0857	<0.0001
TIM	0.896±0.0115	<0.0001	4.13±0.0426	<0.0001	10.2±0.0589	<0.0001

6.2. Kitagawa

The Kitagawa model has a tendency to stabilize around $x = \pm 7$ where it is linear. The challenging portion for filtering is away from the stable portions where the dynamics are highly nonlinear. [9] evaluated the model using $R = 200$ independent starts of the time series allowed to run only $T = 1$ time steps, which we find somewhat unrealistic. Therefore, we allow for $T = 10$ time steps with $R = 200$ independent starts. In this example, $x_1 \sim \mathcal{N}(0, 0.5^2)$.

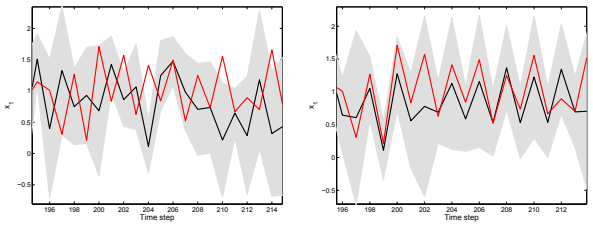
The learned value of the parameters where, $\alpha = 0.3846$, $\beta = 1.2766$, $\kappa = 2.5830$.

6.3. Pendulum

The models were tested on $R = 100$ runs of length $T = 200$ each, with $\mathbf{x}_1 \sim \mathcal{N}([-\pi\, 0], [0.1^2\, 0.2^2])$. The initial state mean of $[-\pi\, 0]$ corresponds to the pendulum being in the downward position. The models were trained on $R = 5$ runs of length $T = 200$. We found that in order to perform well on NLL, but not on MSE and MAE, multiple runs of the time series were needed during training; otherwise, TIM had the best NLL. This is because if the time series is initialized in one state the model will not have a chance to learn the needed parameter settings to avoid rare, but still present, sigma point collapse in other parts of the state-space. A short period single sigma point collapse in a long time series can give the models a worse NLL than even TIM due to incredibly small likelihoods. The MSE and MAE losses are more bounded so a short period of poor performance will be hidden by good performance periods. Even when $R = 1$ during training, sigma point collapse is much rarer than in UKF-L than UKF-D. The UKF found optimal values of the parameters to be $\alpha = 0.5933$, $\beta = 0.1630$, $\kappa = 0.6391$. It is further evidence that the correct θ are hard proscribe a priori and must be learned empirically. We compare the predictions of the default and learned settings in Fig. 3.

6.4. Analysis of sigma point collapse

We find that the marginal likelihood is extremely unstable in regions of θ that experience sigma point collapse. When sigma point collapse occurs, the predictive variances become far too small making the marginal likelihood much more susceptible to noise. Hence, the marginal likelihood is smooth near the optima, as seen in Fig. 2. As a diagnostic



(a) Default θ (b) Learned θ

Fig. 3. Comparison of default and learned for one-step-ahead prediction for first element of $\mathbf{y}_t$ in the Pendulum model. The red line is the truth, while the black line and shaded area represent the mean and 95% confidence interval of the predictive distribution.

D for sigma point collapse we look at the mean $|\Sigma|$ of the predictive distribution.

6.5. Computational Complexity

The UKF-L, UKF, and EKF have test set computational time $\mathcal{O}(DT(D^2 + M))$. The GP-UKF and GP-ADF have complexity $\mathcal{O}((D^3 + D^2MN^2)T)$, where N is the number of points used in training to learn f and g. If a large number of training points N is needed to approximate f and g well the GP-ADF and GP-UKF can become much slower than the UKF.

6.6. Discussion

The learned parameters of the UKF performed significantly better than the default UKF for all error measures and data sets. Likewise, it performed significantly better than all other methods except against the EKF on the pendulum data on MAE, where the two methods are essentially tied. We found that results could be improved further by averaging the predictions of the UKF-L and the EKF.

7. CONCLUSIONS

We have presented an automatic and model based mechanism to learn the parameters of a UKF, $\{\alpha.\beta, \kappa\}$, in a principled way. The UKF can be reinterpreted as a generative process that performs inference on a slightly different NLDS than desired through specification of f and g. We demonstrate how the UKF can fail arbitrarily badly in very nonlinear problems through sigma point collapse. Learning the parameters can make sigma point collapse less likely to occur. When the UKF learns the correct parameters from data it can outperform other filters designed to avoid sigma point collapse, such as the GP-ADF, on common benchmark dynamical systems problems.

Acknowledgements

We thank Steven Bottone, Zoubin Ghahramani, and Andrew Wilson for advice and feedback on this work.

8. REFERENCES

[1] Rudolf E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME — Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[2] Peter S. Maybeck, *Stochastic Models, Estimation, and Control*, vol. 141 of *Mathematics in Science and Engineering*, Academic Press, Inc., 1979.

[3] Simon J. Julier and Jeffrey K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proceedings of AeroSense: 11th Symposium on Aerospace/Defense Sensing, Simulation and Controls*, Orlando, FL, 1997, pp. 182–193.

[4] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[5] Eric A. Wan and Rudolph van der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control, IEEE*, Lake Louise, AB, 2000, pp. 153–158.

[6] Michael A. Osborne, Roman Garnett, and Stephen J. Roberts, "Gaussian processes for global optimization," in *3rd International Conference on Learning and Intelligent Optimization (LION3)*, Trento, Italy, January 2009.

[7] Carl E. Rasmussen and Zoubin Ghahramani, "Bayesian Monte Carlo," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds., pp. 489–496. The MIT Press, Cambridge, MA, USA, 2003.

[8] Ryan Turner, Marc Peter Deisenroth, and Carl Edward Rasmussen, "State-space inference and learning with Gaussian processes," in *the 13th International Conference on Artificial Intelligence and Statistics*, Sardinia, Italy, 2010, vol. 9.

[9] Marc P. Deisenroth, Marco F. Huber, and Uwe D. Hanebeck, "Analytic moment-based Gaussian process filtering," in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, QC, 2009, pp. 225–232, Omnipress.

[10] Genshiro Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[11] Marc P. Deisenroth, Carl E. Rasmussen, and Jan Peters, "Model-based reinforcement learning with continuous states and actions," in *Proceedings of the 16th European Symposium on Artificial Neural Networks (ESANN 2008)*, Bruges, Belgium, April 2008, pp. 19–24.